Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL), Philadelphia, July 2002, pp. 481-488.

Teaching a Weaker Classifier: Named Entity Recognition on Upper Case Text

Hai Leong Chieu
DSO National Laboratories
20 Science Park Drive
Singapore 118230
chaileon@dso.org.sg

Hwee Tou Ng
Department of Computer Science
School of Computing
National University of Singapore
3 Science Drive 2
Singapore 117543
nght@comp.nus.edu.sg

Abstract

This paper describes how a machine-learning named entity recognizer (NER) on upper case text can be improved by using a mixed case NER and some unlabeled text. The mixed case NER can be used to tag some unlabeled mixed case text, which are then used as additional training material for the upper case NER. We show that this approach reduces the performance gap between the mixed case NER and the upper case NER substantially, by 39% for MUC-6 and 22% for MUC-7 named entity test data. Our method is thus useful in improving the accuracy of NERs on upper case text, such as transcribed text from automatic speech recognizers where case information is missing.

1 Introduction

In this paper, we propose using a mixed case named entity recognizer (NER) that is trained on labeled text, to further train an upper case NER. In the Sixth and Seventh Message Understanding Conferences (MUC-6, 1995; MUC-7, 1998), the named entity task consists of labeling named entities with the classes PERSON, ORGANIZATION, LOCATION, DATE, TIME, MONEY, and PERCENT. We conducted experiments on upper case named entity recognition, and showed how unlabeled mixed case text can be used to improve the results of an upper case NER on the official MUC-6 and MUC-7 test data. Besides upper case text, this approach can also be applied on transcribed text from automatic speech recognizers in Speech Normalized Orthographic Representation (SNOR) format, or from optical character recognition (OCR) output. For the English language, a word starting with a capital letter often designates a named entity. Upper case NERs do not have case information to help them to distinguish named entities from non-named entities. When data is sparse, many named entities in the test data would be unknown words. This makes upper case named entity recognition more difficult than mixed case. Even a human would experience greater difficulty in annotating upper case text than mixed case text (Figure 1).

Mixed Case: Consuela Washington, a longtime House staffer and an expert in securities laws, is a leading candidate to be chairwoman of the Securities and Exchange Commission in the Clinton administration.
Upper Case: CONSUELA WASHINGTON, A LONGTIME HOUSE STAFFER AND AN EXPERT IN SECURITIES LAWS, IS A LEADING CANDIDATE TO BE CHAIRWOMAN OF THE SECURITIES AND EXCHANGE COMMISSION IN THE CLINTON ADMINISTRATION.

Figure 1: Examples of mixed and upper case text

We propose using a mixed case NER to "teach" an upper case NER, by making use of unlabeled mixed case text. With the abundance of mixed case un-

labeled texts available in so many corpora and on the Internet, it will be easy to apply our approach to improve the performance of NER on upper case text. Our approach does not satisfy the usual assumptions of co-training (Blum and Mitchell, 1998). Intuitively, however, one would expect some information to be gained from mixed case unlabeled text, where case information is helpful in pointing out new words that could be named entities. We show empirically that such an approach can indeed improve the performance of an upper case NER.

In Section 5, we show that for MUC-6, this way of using unlabeled text can bring a relative reduction in errors of 38.68% between the upper case and mixed case NERs. For MUC-7 the relative reduction in errors is 22.49%.

2 Related Work

Considerable amount of work has been done in recent years on NERs, partly due to the Message Understanding Conferences (MUC-6, 1995; MUC-7, 1998). Machine learning methods such as BBN's IdentiFinder (Bikel, Schwartz, and Weischedel, 1999) and Borthwick's MENE (Borthwick, 1999) have shown that machine learning NERs can achieve comparable performance with systems using hand-coded rules. Bikel, Schwartz, and Weischedel (1999) have also shown how mixed case text can be automatically converted to upper case SNOR or OCR format to train NERs to work on such formats. There is also some work on unsupervised learning for mixed case named entity recognition (Collins and Singer, 1999; Cucerzan and Yarowsky, 1999). Collins and Singer (1999) investigated named entity classification using Adaboost, CoBoost, and the EM algorithm. However, features were extracted using a parser, and performance was evaluated differently (the classes were person, organization, location, and noise). Cucerzan and Yarowsky (1999) built a cross language NER, and the performance on English was low compared to supervised single-language NER such as IdentiFinder. We suspect that it will be hard for purely unsupervised methods to perform as well as supervised ones.

Seeger (2001) gave a comprehensive summary of recent work in learning with labeled and unlabeled data. There is much recent research on co-training, such as (Blum and Mitchell, 1998; Collins and Singer, 1999; Pierce and Cardie, 2001). Most co-training methods involve using two classifiers built on different sets of features. Instead of using distinct sets of features, Goldman and Zhou (2000) used different classification algorithms to do co-training.

Blum and Mitchell (1998) showed that in order for PAC-like guarantees to hold for co-training, features should be divided into two disjoint sets satisfying: (1) each set is sufficient for a classifier to learn a concept correctly; and (2) the two sets are conditionally independent of each other. Each set of features can be used to build a classifier, resulting in two independent classifiers, A and B. Classifications by A on unlabeled data can then be used to further train classifier B, and vice versa. Intuitively, the independence assumption is there so that the classifications of A would be informative to B. When the independence assumption is violated, the decisions of A may not be informative to B. In this case, the positive effect of having more data may be offset by the negative effect of introducing noise into the data (classifier A might not be always correct).

Nigam and Ghani (2000) investigated the difference in performance with and without a feature split, and showed that co-training with a feature split gives better performance. However, the comparison they made is between co-training and self-training. In self-training, only one classifier is used to tag unlabeled data, after which the more confidently tagged data is reused to train the same classifier.

Many natural language processing problems do not show the natural feature split displayed by the web page classification task studied in previous co-training work. Our work does not really fall under the paradigm of co-training. Instead of co-operation between two classifiers, we used a stronger classifier to teach a weaker one. In addition, it exhibits the following differences: (1) the features are not at all independent (upper case features can be seen as a subset of the mixed case features); and (2) The additional features available to the mixed case system will never be available to the upper case system. Co-training often involves combining the two different sets of features to obtain a final system that outperforms either system alone. In our context, however, the upper case system will never have access

to some of the case-based features available to the mixed case system.

Due to the above reason, it is unreasonable to expect the performance of the upper case NER to match that of the mixed case NER. However, we still manage to achieve a considerable reduction of errors between the two NERs when they are tested on the official MUC-6 and MUC-7 test data.

3 System Description

We use the maximum entropy framework to build two classifiers: an upper case NER and a mixed case NER. The upper case NER does not have access to case information of the training and test data, and hence cannot make use of all the features used by the mixed case NER. We will first describe how the mixed case NER is built. More details of this mixed case NER and its performance are given in (Chieu and Ng, 2002). Our approach is similar to the MENE system of (Borthwick, 1999). Each word is assigned a name class based on its features. Each name class N is subdivided into 4 classes, i.e., *N_begin, N_continue, N_end,* and *N_unique*. Hence, there is a total of 29 classes (7 name classes $\times$ 4 sub-classes $+$ 1 not-a-name class).

3.1 Maximum Entropy

The maximum entropy framework estimates probabilities based on the principle of making as few assumptions as possible, other than the constraints imposed. Such constraints are derived from training data, expressing some relationship between features and outcome. The probability distribution that satisfies the above property is the one with the highest entropy. It is unique, agrees with the maximum-likelihood distribution, and has the exponential form (Della Pietra, Della Pietra, and Lafferty, 1997):

$$p(o|h) = \frac{1}{Z(h)} \prod_{j=1}^{k} \alpha_j^{f_j(h,o)},$$

where o refers to the outcome, h the history (or context), and $Z(h)$ is a normalization function. In addition, each feature function $f_j(h, o)$is a binary function. For example, in predicting if a word belongs to a word class, o is either true or false, and h refers to the surrounding context:

$$f_j(h,o) = \begin{cases} 1 & \text{if } o = \textit{true}, \text{ previous word} = \textit{the} \\ 0 & \text{otherwise} \end{cases}$$

The parameters α_j are estimated by a procedure called Generalized Iterative Scaling (GIS) (Darroch and Ratcliff, 1972). This is an iterative method that improves the estimation of the parameters at each iteration.

3.2 Features for Mixed Case NER

The features we used can be divided into 2 classes: local and global. Local features are features that are based on neighboring tokens, as well as the token itself. Global features are extracted from other occurrences of the same token in the whole document.

Features in the maximum entropy framework are binary. Feature selection is implemented using a feature cutoff: features seen less than a small count during training will not be used. We group the features used into feature groups. Each group can be made up of many binary features. For each token w, zero, one, or more of the features in each group are set to 1.

The local feature groups are:

Non-Contextual Feature: This feature is set to 1 for all tokens. This feature imposes constraints that are based on the probability of each name class during training.

Zone: MUC data contains SGML tags, and a document is divided into zones (e.g., headlines and text zones). The zone to which a token belongs is used as a feature. For example, in MUC-6, there are four zones (*TXT, HL, DATELINE, DD*). Hence, for each token, one of the four features *zone-TXT, zone-HL, zone-DATELINE,* or *zone-DD* is set to 1, and the other 3 are set to 0.

Case and Zone: If the token w starts with a capital letter (*initCaps*), then an additional feature *(initCaps, zone)* is set to 1. If it is made up of all capital letters, then *(allCaps, zone)* is set to 1. If it contains both upper and lower case letters, then *(mixedCaps, zone)* is set to 1. A token that is *allCaps* will also be *initCaps*. This group consists of (3 $\times$ *total number of possible zones*) features.

Case and Zone of w_{+1} and w_{-1}: Similarly, if w_{+1} (or w_{-1}) is *initCaps*, a feature *(initCaps,*

Token satisfies	Example	Feature
Starts with a capital letter, ends with a period	*Mr.*	*InitCap-Period*
Contains only one capital letter	*A*	*OneCap*
All capital letters and period	*CORP.*	*AllCaps-Period*
Contains a digit	*AB3, 747*	*Contain-Digit*
Made up of 2 digits	*99*	*TwoD*
Made up of 4 digits	*1999*	*FourD*
Made up of digits and slash	*01/01*	*Digit-slash*
Contains a dollar sign	*US$20*	*Dollar*
Contains a percent sign	*20%*	*Percent*
Contains digit and period	*$US3.20*	*Digit-Period*

Table 1: Features based on the token string

zone)$_{NEXT}$ (or *(initCaps, zone)*$_{PREV}$) is set to 1, etc.

Token Information: This group consists of 10 features based on the string w, as listed in Table 1. For example, if a token starts with a capital letter and ends with a period (such as *Mr.*), then the feature *InitCapPeriod* is set to 1, etc.

First Word: This feature group contains only one feature *firstword*. If the token is the first word of a sentence, then this feature is set to 1. Otherwise, it is set to 0.

Lexicon Feature: The string of the token w is used as a feature. This group contains a large number of features (one for each token string present in the training data). At most one feature in this group will be set to 1. If w is seen infrequently during training (less than a small count), then w will not selected as a feature and all features in this group are set to 0.

Lexicon Feature of Previous and Next Token: The string of the previous token w_{-1} and the next token w_{+1} is used with the *initCaps* information of w. If w has *initCaps*, then a feature *(initCaps,* w_{+1}*)*$_{NEXT}$ is set to 1. If w is not *initCaps*, then *(not-initCaps,* w_{+1}*)*$_{NEXT}$ is set to 1. Same for w_{-1}. In the case where the next token w_{+1} is a hyphen, then w_{+2} is also used as a feature: *(initCaps,* w_{+2}*)*$_{NEXT}$ is set to 1. This is because in many cases, the use of hyphens can be considered to be optional (e.g., "third-quarter" or "third quarter").

Out-of-Vocabulary: We derived a lexicon list from WordNet 1.6, and words that are not found in this list have a feature *out-of-vocabulary* set to 1.

Dictionaries: Due to the limited amount of training material, name dictionaries have been found to be useful in the named entity task. The sources of our dictionaries are listed in Table 2. A token w is tested against the words in each of the four lists of location names, corporate names, person first names, and person last names. If w is found in a list, the corresponding feature for that list will be set to 1. For example, if *Barry* is found in the list of person first names, then the feature *PersonFirstName* will be set to 1. Similarly, the tokens w_{+1} and w_{-1} are tested against each list, and if found, a corresponding feature will be set to 1. For example, if w_{+1} is found in the list of person first names, the feature *PersonFirstName*$_{NEXT}$ is set to 1.

Month Names, Days of the Week, and Numbers: If w is one of *January*, *February*, ..., *December*, then the feature *MonthName* is set to 1. If w is one of *Monday*, *Tuesday*, ..., *Sunday*, then the feature *DayOfTheWeek* is set to 1. If w is a number string (such as *one*, *two*, etc), then the feature *NumberString* is set to 1.

Suffixes and Prefixes: This group contains only two features: *Corporate-Suffix* and *Person-Prefix*. Two lists, *Corporate-Suffix-List* (for corporate suffixes) and *Person-Prefix-List* (for person prefixes), are collected from the training data. For a token w that is in a consecutive sequence of *initCaps* tokens $(w_{-m}, \ldots, w, \ldots, w_{+n})$, if any of the tokens from w_{+1} to w_{+n} is in *Corporate-Suffix-List*, then a feature *Corporate-Suffix* is set to 1. If any of the tokens from w_{-m-1} to w_{-1} is in *Person-Prefix-List*, then another feature *Person-Prefix* is set to 1. Note that we check for w_{-m-1}, the word preceding the consecutive sequence of *initCaps* tokens, since person prefixes like *Mr.*, *Dr.* etc are not part of person names, whereas corporate suffixes like *Corp.*, *Inc.* etc are part of corporate names.

The global feature groups are:

InitCaps of Other Occurrences: There are 2 features in this group, checking for whether the first occurrence of the same word in an unambiguous posi-

Description	Source
Location Names	http://www.timeanddate.com http://www.cityguide.travel-guides.com http://www.worldtravelguide.net
Corporate Names	http://www.fmlx.com
Person First Names	http://www.census.gov/genealogy/names
Person Last Names	

Table 2: Sources of Dictionaries

tion (non first-words in the *TXT* or *TEXT* zones) in the same document is *initCaps* or *not-initCaps*. For a word whose *initCaps* might be due to its position rather than its meaning (in headlines, first word of a sentence, etc), the case information of other occurrences might be more accurate than its own.

Corporate Suffixes and Person Prefixes of Other Occurrences: With the same *Corporate-Suffix-List* and *Person-Prefix-List* used in local features, for a token w seen elsewhere in the same document with one of these suffixes (or prefixes), another feature *Other-CS* (or *Other-PP*) is set to 1.

Acronyms: Words made up of all capitalized letters in the text zone will be stored as acronyms (e.g., *IBM*). The system will then look for sequences of initial capitalized words that match the acronyms found in the whole document. Such sequences are given additional features of *A_begin*, *A_continue*, or *A_end*, and the acronym is given a feature *A_unique*. For example, if "FCC" and "Federal Communications Commission" are both found in a document, then "Federal" has *A_begin* set to 1, "Communications" has *A_continue* set to 1, "Commission" has *A_end* set to 1, and "FCC" has *A_unique* set to 1.

Sequence of Initial Caps: In the sentence "Even News Broadcasting Corp., noted for its accurate reporting, made the erroneous announcement.", a NER may mistake "Even News Broadcasting Corp." as an organization name. However, it is unlikely that other occurrences of "News Broadcasting Corp." in the same document also co-occur with "Even". This group of features attempts to capture such information. For every sequence of initial capitalized words, its longest substring that occurs in the same document is identified. For this example, since the sequence "Even News Broadcasting Corp." only appears once in the document, its longest substring that occurs in the same document is "News Broadcasting Corp.". In this case, "News" has an additional feature of *I_begin* set to 1,"Broadcasting" has an additional feature of *I_continue* set to 1, and "Corp." has an additional feature of *I_end* set to 1.

Unique Occurrences and Zone: This group of features indicates whether the word w is unique in the whole document. w needs to be in *initCaps* to be considered for this feature. If w is unique, then a feature *(Unique, Zone)* is set to 1, where *Zone* is the document zone where w appears.

3.3 Features for Upper Case NER

All features used for the mixed case NER are used by the upper case NER, except those that require case information.

Among local features, *Case and Zone*, *InitCap-Period*, and *OneCap* are not used by the upper case NER. Among global features, only *Other-CS* and *Other-PP* are used for the upper case NER, since the other global features require case information. For *Corporate-Suffix* and *Person-Prefix*, as the sequence of *initCaps* is not available in upper case text, only the next word (previous word) is tested for *Corporate-Suffix* (*Person-Prefix*).

3.4 Testing

During testing, it is possible that the classifier produces a sequence of inadmissible classes (e.g., *person_begin* followed by *location_unique*). To eliminate such sequences, we define a transition probability between word classes $P(c_i|c_j)$ to be equal to 1 if the sequence is admissible, and 0 otherwise. The probability of the classes $c_1, \ldots, c_n$ assigned to the words in a sentence s in a document D is defined as follows:

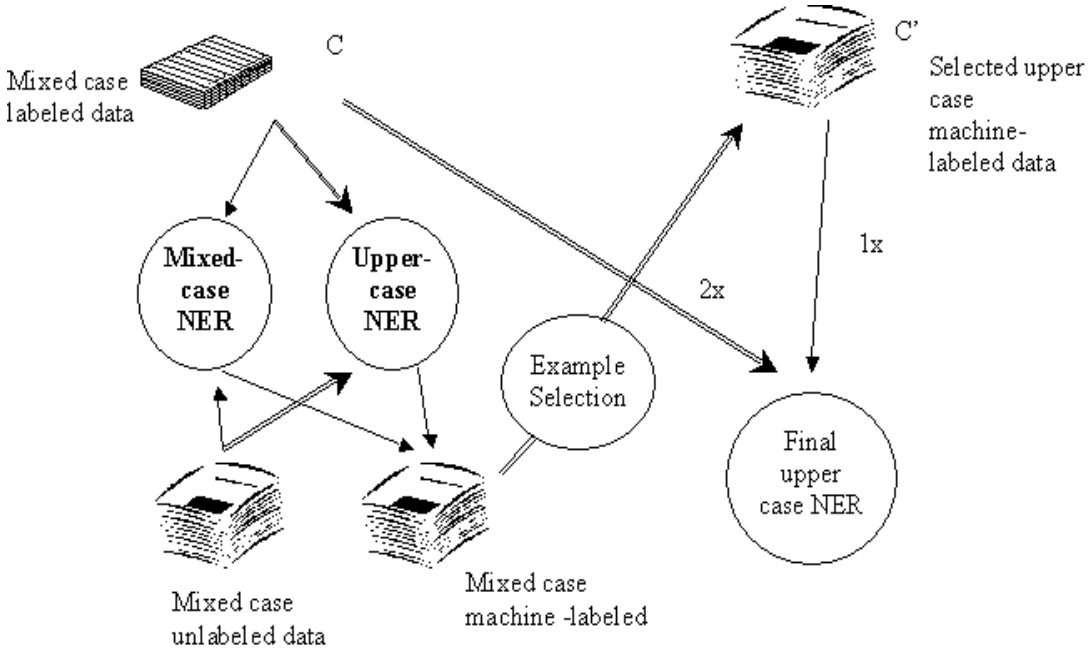

Figure 2: The whole process of re-training the upper case NER. ⇒ signifies that the text is converted to upper case before processing.

$$P(c_1, \ldots, c_n|s, D) = \prod_{i=1}^{n} P(c_i|s, D) * P(c_i|c_{i-1}),$$

where $P(c_i|s, D)$ is determined by the maximum entropy classifier. A dynamic programming algorithm is then used to select the sequence of word classes with the highest probability.

4 Teaching Process

The teaching process is illustrated in Figure 2. This process can be divided into the following steps:

Training NERs. First, a mixed case NER (MNER) is trained from some initial corpus C, manually tagged with named entities. This corpus is also converted to upper case in order to train another upper case NER (UNER). UNER is required by our method of example selection.

Baseline Test on Unlabeled Data. Apply the trained MNER on some unlabeled mixed case texts to produce mixed case texts that are machine-tagged with named entities (*text-mner-tagged*). Convert the original unlabeled mixed case texts to upper case, and similarly apply the trained UNER on these texts to obtain upper case texts machine-tagged with named entities (*text-uner-tagged*).

Example Selection. Compare *text-mner-tagged* and *text-uner-tagged* and select tokens in which the classification by MNER differs from that of UNER. The class assigned by MNER is considered to be correct, and will be used as new training data. These tokens are collected into a set C'.

Retraining for Final Upper Case NER. Both C and C' are used to retrain an upper case NER. However, tokens from C are given a weight of 2 (i.e., each token is used twice in the training data), and tokens from C' a weight of 1, since C is more reliable than C' (human-tagged versus machine-tagged).

5 Experimental Results

For manually labeled data (corpus C), we used only the official training data provided by the MUC-6 and MUC-7 conferences, i.e., using MUC-6 training data and testing on MUC-6 test data, and using MUC-7 training data and testing on MUC-7 test data.[1] The task definitions for MUC-6 and MUC-7 are not exactly identical, so we could not combine the training data. The original MUC-6 training data has a total of approximately 160,000 tokens and

[1]MUC data can be obtained from the Linguistic Data Consortium: http://www.ldc.upenn.edu

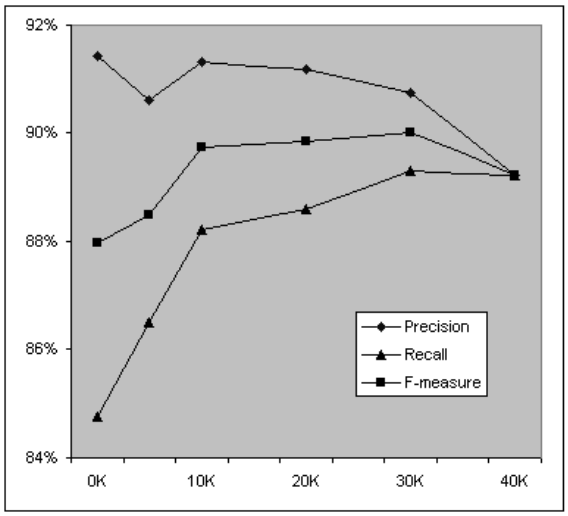

Figure 3: Improvements in F-measure on MUC-6 plotted against amount of selected unlabeled data used

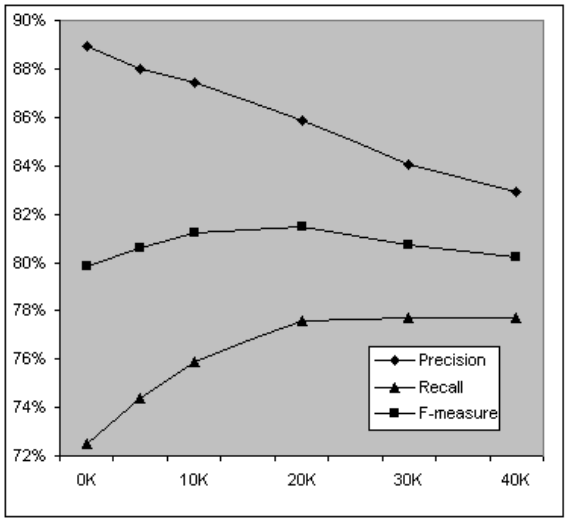

Figure 4: Improvements in F-measure on MUC-7 plotted against amount of selected unlabeled data used

MUC-7 a total of approximately 180,000 tokens.

The unlabeled text is drawn from the TREC (Text REtrieval Conference) corpus, 1992 Wall Street Journal section. We have used a total of 4,893 articles with a total of approximately 2,161,000 tokens. After example selection, this reduces the number of tokens to approximately 46,000 for MUC-6 and 67,000 for MUC-7.

Figure 3 and Figure 4 show the results for MUC-6 and MUC-7 obtained, plotted against the number of unlabeled instances used. As expected, it increases the recall in each domain, as more names or their contexts are learned from unlabeled data. However, as more unlabeled data is used, precision drops due to the noise introduced in the machine tagged data. For MUC-6, F-measure performance peaked at the point where 30,000 tokens of machine labeled data are added to the original manually tagged 160,000 tokens. For MUC-7, performance peaked at 20,000 tokens of machine labeled data, added to the original manually tagged 180,000 tokens.

The improvements achieved are summarized in Table 3. It is clear from the table that this method of using unlabeled data brings considerable improvement for both MUC-6 and MUC-7 named entity task.

The result of the teaching process for MUC-6 is a lot better than that of MUC-7. We think that this is

Systems	MUC-6	MUC-7
Baseline Upper Case NER	87.97%	79.86%
Best Taught Upper Case NER	90.02%	81.52%
Mixed case NER	93.27%	87.24%
Reduction in relative error	38.68%	22.49%

Table 3: F-measure on MUC-6 and MUC-7 test data

due to the following reasons:

Better Mixed Case NER for MUC-6 than MUC-7. The mixed case NER trained on the MUC-6 officially released training data achieved an F-measure of 93.27% on the official MUC-6 test data, while that of MUC-7 (also trained on only the official MUC-7 training data) achieved an F-measure of only 87.24%. As the mixed case NER is used as the teacher, a bad teacher does not help as much.

Domain Shift in MUC-7. Another possible cause is that there is a domain shift in MUC-7 for the formal test (training articles are aviation disasters articles and test articles are missile/rocket launch articles). The domain of the MUC-7 test data is also very specific, and hence it might exhibit different properties from the training and the unlabeled data.

The Source of Unlabeled Data. The unlabeled data used is from the same source as MUC-6, but different for MUC-7 (MUC-6 articles and the unlabeled articles are all Wall Street Journal articles,

whereas MUC-7 articles are New York Times articles).

6 Conclusion

In this paper, we have shown that the performance of NERs on upper case text can be improved by using a mixed case NER with unlabeled text. Named entity recognition on mixed case text is easier than on upper case text, where case information is unavailable. By using the teaching process, we can reduce the performance gap between mixed and upper case NER by as much as 39% for MUC-6 and 22% for MUC-7. This approach can be used to improve the performance of NERs on speech recognition output, or even for other tasks such as part-of-speech tagging, where case information is helpful. With the abundance of unlabeled text available, such an approach requires no additional annotation effort, and hence is easily applicable.

This way of teaching a weaker classifier can also be used in other domains, where the task is to infer $X \rightarrow Y$, and an abundance of unlabeled data $D = \{X, Z\}$ is available. If one possesses a second classifier $(X, Z) \rightarrow Y$ such that Z provides additional "useful" information that can be utilized by this second classifier, then one can use this second classifier to automatically tag the unlabeled data D, and select from D examples that can be used to supplement the training data for training $X \rightarrow Y$.

References

Daniel M. Bikel, Richard Schwartz, and Ralph M. Weischedel. 1999. An Algorithm that Learns What's in a Name. *Machine Learning*, 34(1/2/3):211-231.

Avrim Blum and Tom Mitchell. 1998. Combining Labeled and Unlabeled Data with Co-Training. In *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*, 92-100.

Andrew Borthwick. 1999. A Maximum Entropy Approach to Named Entity Recognition. Ph.D. dissertation. Computer Science Department. New York University.

Hai Leong Chieu and Hwee Tou Ng. 2002. Named Entity Recognition: A Maximum Entropy Approach Using Global Information. To appear in *Proceedings of the Nineteenth International Conference on Computational Linguistics*.

Michael Collins and Yoram Singer. 1999. Unsupervised Models for Named Entity Classification. In *Proceedings of the 1999 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, 100-110.

Silviu Cucerzan and David Yarowsky. 1999. Language Independent Named Entity Recognition Combining Morphological and Contextual Evidence. In *Proceedings of the 1999 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, 90-99.

J. N. Darroch and D. Ratcliff. 1972. Generalized Iterative Scaling for Log-Linear Models. *The Annals of Mathematical Statistics*, 43(5):1470-1480.

Stephen Della Pietra, Vincent Della Pietra, and John Lafferty. 1997. Inducing Features of Random Fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(4):380-393.

Sally Goldman and Yan Zhou. 2000. Enhancing Supervised Learning with Unlabeled Data. In *Proceedings of the Seventeenth International Conference on Machine Learning*, 327-334.

MUC-6. 1995. *Proceedings of the Sixth Message Understanding Conference (MUC-6)*.

MUC-7. 1998. *Proceedings of the Seventh Message Understanding Conference (MUC-7)*.

Kamal Nigam and Rayid Ghani. 2000. Analyzing the Effectiveness and Applicability of Co-training. In *Proceedings of the Ninth International Conference on Information and Knowledge Management*, 86-93.

David Pierce and Claire Cardie. 2001. Limitations of Co-Training for Natural Language Learning from Large Datasets. In *Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing*, 1-9.

Matthias Seeger. 2001. Learning with Labeled and Unlabeled Data. Technical Report, University of Edinburgh.