Bondec – A Sentence Boundary Detector

Haoyi Wang
Stanford Engineering Informatics
Stanford University
Stanford, CA 94305
haoyiw@stanford.edu

Yang Huang
Stanford Medical Informatics
Stanford University
Stanford, CA 94305
huangy@stanford.edu

ABSTRACT
The Bondec system is a sentence boundary detection system. It has three independent applications (Rule-based, HMM, and Maximum Entropy). Maximum Entropy Model is the central part of this system, which achieved an error rate less than 2% on part of the Wall Street Journal (WSJ) Corpus with only eight binary features. The performance of the three applications is illustrated and discussed.

Keywords:
Sentence boundary disambiguation, Maximum Entropy Model, Features, Generalized Iterative Scaling, Hidden Markov Model.

1 INTRODUCTION
Sentence boundary disambiguation is the task of identifying the sentence elements within a paragraph or an article. Because the sentence is the basic textual unit immediately above the word and phrase, Sentence Boundary Disambiguation (SBD) is one of the essential problems for many applications of Natural Language Processing – Parsing, Information Extraction, Machine Translation, and Document Summarizations. The accuracy of the SBD system will directly affect the performance of these applications. However, the past research work in this field has already achieved very high performance, and it is not very active now. The problem seems too simple to attract the attention of the researchers.

In fact, the problem itself is not as simple as it appears to be. We all know that a sentence is a sequence of words ending with a terminal punctuation, such as a '.', '?', or '!'. Most sentences use a period at the end. However, we should notice that sometimes a period can be associated with an abbreviation, such as "Mr." or represent a decimal point in a number like $12.58. In these cases, it is a part of an abbreviation or a number; we cannot delimit a sentence because the period has a different meaning here. On the other hand, the trailing period of an abbreviation can also represent the end of a sentence at the same time. In most such cases, the word following this period is a capitalized common word (e.g., The President lives in Washington D.C. He likes that place.). Moreover, if the following word is a proper noun or part of a proper phrase, which is always capitalized, the SBD system usually should not label the period as the start of the next sentence but as a part of the same sentence (e.g., P. R. China). Disambiguating a proper name from a common word is a challenging problem and makes the sentence boundary ambiguity problem even more complicated.

The original SBD systems were built from manually generated rules in the form of regular expressions for grammar, which is augmented by a list of abbreviations, common words, proper names, etc. For example, the Almebic system (Aderdenn et al., 1995) deploys over 100 regular-expression rules written in Flex. Such a system may work well on the language or corpus for which they were designed. Nevertheless, developing and maintaining an accurate rule-based system require substantial hand coding effort and domain knowledge, which are very time-consuming. Another drawback of this kind of systems is that it is difficult to port an existing system to other domains or corpora of other languages. Such a switch is equal to building a new system beginning from scratch.

The current research activity in SBD focuses on employing machine learning techniques, such as Decision Tree, Neural Network, Maximum Entropy, and Hidden Markov Model, which treat the SBD task as a standard classification problem. The general principles of these systems are: training the

system on a training set (usually annotated) to make the system "remember" the features of the local context around the sentence-breaking punctuation or global information on the list of abbreviations and proper names, and then recognize the real text sentences using this trained system.

Because the system we developed only implements machine-learning techniques, we will limit our discussion to the scope of this category. For this project report, Section One describes the problem we want to solve; Section Two summarizes the related research on the machine-learning systems; Section Three illustrates the approach we chose for this topic, including an introduction to the mathematic background; Section Four discusses the principal algorithms and explains the architecture of Bondec system; Section Five evaluates the performance of our system and compares it among three applications; Section Six demonstrates the experiences and lessons we derived from our work.

2. RELATED RESEARCH

Palmer and Hearst (1997) developed a system - SATZ - to use the local syntactic context to classify the potential sentence boundary. To obtain the syntactic information for local context, SATZ needs the words in the context to be tagged with part-of-speech (POS). "However, requiring a single POS tagging part-of-speech assignment for each word introduces a processing circularity: because most part-of-speech taggers require predetermined sentence boundaries, the boundary disambiguation must be done before tagging. But if the disambiguation must be done before tagging, no part-of-speech assignments are available for the boundary determination system."

To bypass this problem, SATZ redefines Penn Treebank POS tags into 18 generic POS categories, such as noun, article, proper noun, preposition, etc. These 18 categories are combined with two other non-POS categories – capitalization and following a punctuation mark – to compose a syntactic category set for each word. Thus, the sets for three tokens before and three tokens after the boundary candidate constitute the local syntactic context, which is the input for two types of classifiers – decision trees and neutral networks. They reported a performance of around 1.0% error rate on *Wall Street Journal* (WSJ) data.

In order to solve the loop problem encountered by the SANZ system, Mikheev (2000) segments a sentence into smaller sections. His POS predictions are deducted from the ambiguity tag of the current token and the partially disambiguated tags of the two previous word-tokens. This tri-gram POS tagger can be built from a mixture of Hidden Markov Models (HMM) and Maximum Entropy (ME) techniques. He claimed only a 0.25% error rate on the Brown corpus and a 0.39% error rate on the WSJ corpus.

An additional research study by Mikheev (2000) attempted to identify a list of abbreviations and proper names in the corpus. He defined several heuristic rules to globally detect potential abbreviations and proper nouns, such as "If a word has been used capitalized in an unambiguous context, this increases the chances for this word to act as a proper name in mandatory position in the same document.". With this enhanced feature set, his POS tagger can achieve a 0.07% smaller error rate than the original one.

Instead of tagging the context, Reynar and Ratnaparkhi (1997) presented a solution based on a Maximum Entropy (ME) model for the SBD problem. The main advantages of their approach are: convincing mathematic background (Section Three will disclose the nature of a ME model in detail.) and a little essential information for disambiguation work. The ME model they created does not require POS tags, heuristic rules, or domain-specific information, such as the list of abbreviations and proper names. Instead, the system can use any diverse features extracted from the local context. The model can attain an accuracy of 98.8% on the WSJ data, which is very powerful, considering how simple the model is and how flexibly it can select the features.

3. OUR APPROACH

Among many potential machine-learning methods for SBD, such as Naïve Bayes, Decision Tree, Neutral Network, Hidden Markov Model, and Maximum Entropy, we decided to choose ME as our main method of solving this problem. Making this decision is due to: first, the ME model has a solid

mathematical foundation; second, the features in ME model could be from heterogeneous sources and be implemented easily; third, ME is relatively new to us and we want to gain some knowledge on it from this project. In this section, we will explain the basic mathematical theories within the ME model.

3.1 Entropy

Entropy is an essential terminology in the field of information theory (Manning, 2002). It was originally used to estimate how much of the data can be compressed before they are transmitted over a communication channel (Shannon 1948). The entropy H itself measures the average uncertainty of a single random variable *X*:

$$H(p) = H(X) = \sum_{x \in X} p(x) \log_2 p(x) \quad (1)$$

In Equation 1, *p(x)* is the probability mass function of the random variable *X*. And the above equation tells us the average bits we need to transfer all the information in *X*. For example, if we toss a coin and make *X* be the count of head; *X* will be a binary random variable. We can assume *p(X=1) = p* and *p(X=0) = (1-p)*. *H(X)* is:

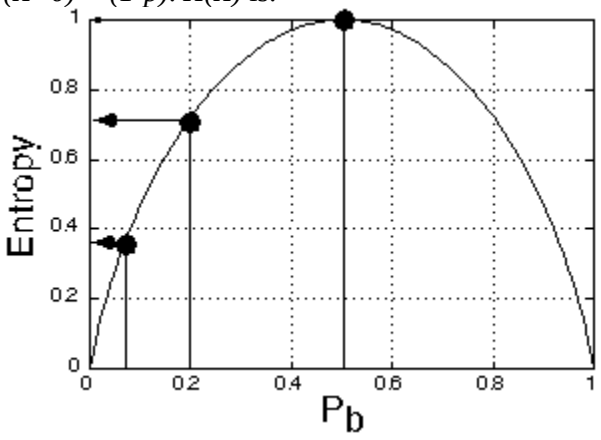

Figure 1 The entropy of a binary random variable.

Figure 1 claims that $H(X) \geq 0$ and $H(X) = 0$ only when *X* has a fixed value; hence, no information is embedded in this variable. This figure also illustrates that *H(X)* reach its maximum point when *p* is equal to 0.5, which means *X* has a uniform distribution.

To save the bandwidth of a communication channel, we prefer a model of *X* with less entropy so that we can use smaller bits to interpret the uncertainty (information) inside *X*. However, in this project, we want to build a model to maximize the entropy. It sounds as though we are violating the basic principle in entropy. Actually, the chief reason to do so is to preserve as less bias as possible when the certainty cannot be identified from the empirical evidence.

3.2 Maximum Entropy Model

If we treat a paragraph/corpus as a token stream, we can consider the SBD problem to be a random process, which delimits this paragraph/corpus into sentences. Such a process will produce an output value *y*, whose domain **y** is all of the possible boundary positions in the stream. We define a random variable *Y* over this domain, and *y* is a particular value of *Y*. In this random process, the value of *Y* may be affected by some contextual information *x*, whose domain ***x*** is all the possible textual combinations in the stream. We can assume **x** is an infinite set. Similar to *y*, we also define a random variable *X* from this infinite domain. To solve the SBD problem, we can build a stochastic model to correctly simulate this random process. Such a model is a conditional probability model - give a context stream *x* and predict the boundaries *y* – $p(y|x)$

Like other machine-learning approaches, we also adopt a training file to represent the real world scenarios. Our task is to construct a model that has the maximum likelihood value with the training file. Thus, we can rank this model as the best one to characterize the real world. At the first step to simulate the random process, a large number of samples – $(x_1, y_1), (x_2, y_2) \ldots (x_N, y_N)$ are extracted from the training set. We can define a joint empirical distribution over *x* and *y* from these samples:

$$\tilde{p}(x, y) = \frac{1}{N} \times number\ of\ (x,y) \quad (2)$$

Besides sampling the training file, we can also acquire some features from the training sample, which are quite helpful for this classification problem. For instance, we observe that if the word following a period is capitalized, then the period is a sentence boundary with a high probability. We can introduce a binary function *f* for this feature:

$$f(x, y) = \begin{cases} 1 \text{ if } x \text{ is a capitalized word following} \\ \text{a period } y, \text{the period is a boundary.} \\ 0 \text{ otherwise} \end{cases} \quad (3)$$

We may define many features for the SBD problem. Each of them places a constraint on the model: the expectation of the feature from the training sample

must be the same as the expectation of this feature in the model:

$$\sum_{x,y} \tilde{p}(x)\, p(y \mid x)\, f(x,y) = \sum_{x,y} \tilde{p}(x,y)\, f(x,y) \quad (4)$$

Where $\tilde{p}(x)$ in the constraint is the empirical distribution of *x* in the training sample.

However, there are still many conditional probabilty models, which can satisfy the constraints from the above equation. To find the best one for the training set, we should derive the model with the maximum entropy, which means that we need a most uniform distribution on all of the unknown features. Therefore, the best model *p** for the SBD should maximize the conditional entroy on *p(y|x)*:

$$H(p) = -\sum_{x,y} \tilde{p}(x)\, p(y \mid x) \log\, p(y \mid x) \quad (5)$$

$$p^* = \arg\max_{p} H(p) \quad (6)$$

And subject to the following constraints at the same time:

1. $p(y|x) \geq 0$. For all x,y.
2. $\sum_y p(y \mid x) = 1$. This and the previous condition guarantee that *p(y|x)* is a conditional probability distribution.
3. $\sum_{x,y} \tilde{p}(x)\, p(y \mid x)\, f_i(x,y) = \sum_{x,y} \tilde{p}(x,y)\, f_i(x,y)$ $i \in \{1,2...n\}$ For all of the selected features (constraints).

This is a typical constrained optimization problem, and we can use Lagrange multiplier method to solve it. We do not want to go through the whole induction procedure; the reader could refer to Berger's report on it (1996). The final result of this problem is a log-linear (exponential) model:

$$p^*(y \mid x) = Z(x) \exp\left(\sum_i \lambda_i f_i(x,y) \right) \quad (7)$$

where *Z(x)*, the normalizing factor, is given by

$$Z(x) = \sum_y \exp\left(\sum_i \lambda_i f_i(x,y) \right) \quad (8)$$

where $\lambda_i\ i \in \{1,2...n\}$ is the Lagrange multiplier associated with the constraint f_i. And it also measures the weight (importance) of feature f_i. Berger (1997) proved that the model from Equation 7 also maximizes the log-likelihood of the joint empirical distribution $\tilde{p}(x,y)$:

$$L_{\tilde{p}}(\Lambda) = \sum_{x,y} \tilde{p}(x,y) \log\, p_\Lambda(y \mid x) \quad (9)$$

where Λ is a vector of weights: $\{\lambda_1, \lambda_2 ... \lambda_n\}$

3.3 Generalized Iterative Scaling

For the Equation 7, there is no analytical method to obtain the value of λ_i in this log-linear distribution. Therefore, we choose generalize iterative scaling as our numerical approach to obtain the vector Λ^*. This iterative procedure will converge to the distribution *p**. The details of this algorithm can be found on pages 591- 593 of Manning's book (2003). We will describe how we implement this algorithm in the next section as well.

4. SYSTEM ARCHITECTURE

4.1 Train and Test Corpora

We obtained the train and test file from Dr. David Palmer. He claimed that these annotated files were constructed from the WSJ corpus (Palmer and Hearst, 1997). For our project, we create three files – train.dat, test.dat, and heldout.dat - from his raw data files. The train.dat file is used for training purpose in HMM and ME. There are 21,026 sentences in this training set. Within these sentences, 95.25% (20,028) of them are delimited by a period; 3.47% (727) end with a quotation mark; and 0.69% (146) of them end with a question mark. The heldout set, which has 9721 sentences, was used for cross-validation and performance tuning; while the test set, which has 9758 sentences, was only available for final performance measurements. The distributions of the terminal punctuations in the two files are similar to the training file. This guarantees that the trained model is not bias toward the test data set. Inside these three corpora, 236 sentences have no boundary punctuation at all. We observe that these sentences are usually in the form of an address or a title.

4.2 Benchmark Tool

To compare the performance of our machine-learning approaches with the rule-based system, we develop a rule-based tool and treat it as a benchmark in Bondec. This application justifies the sentence boundary only based on the syntax rules. According the real distribution in our test corpus, we implemented three criteria in this simple SBD utility program:

1. If the word after a question mark is capitalized, then the question mark is a sentence boundary.
2. If the token before a double quote is a period or question mark, then the double quote is a sentence boundary.
3. If the word after a period is capitalized and the word before this period is not, then the period is a sentence boundary.

With these simple syntax rules, this baseline system can predict the test cases with a precision of 99.56% and a recall of 76.95%. The value of the average performance – F measure - is about 86.81%.

4.3 Hidden Markov Model

In Bondec system, we also build a HMM for SBD. We would like to compare this model with the ME model, which is the core part in the whole architecture. We directly deploy the package edu.stanford.nlp.ie.hmm as our HMM module in the project. This package is the same as the one we applied in the CS276b projects.

Because we already have a self-contained package on HMM, the develop work left for us is to implement the pre and post processing system.

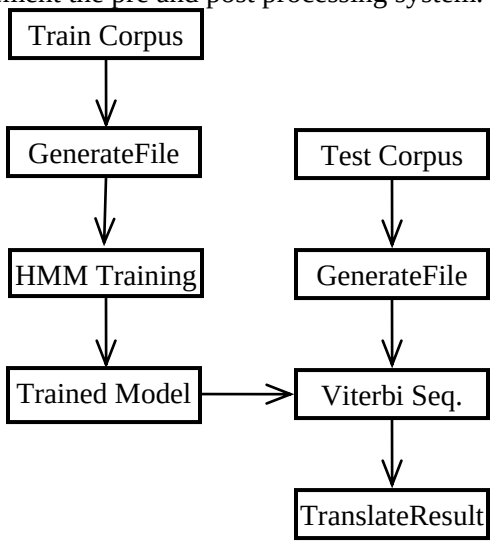

Figure 2. Flow Chart to Apply HMM in Bondec

Figure 2 shows how to associate the existing HMM package with the SBD problem. The pre-processing system includes the GenerateFile class, whose function is transforming the sentences in the original train/test file (See Figure 3) into the document format (See Figure 4) required by HMM package. Therefore, the package can train a HMM model from the well-formatted training set. This model will be exploited later to predict the sentence boundary in the well-formatted test file. The output predictions from this model are interpreted by the class TranlateResult for evaluation purposes.

<s> BEVERLY HILLS hardly seems the place for the little guy to get an even break, but some small-business owners recently struck a blow for equality in this ghetto of glitz. </s>
<s> The arena for this victory was the Beverly Hills Chamber of Commerce. </s>

Figure 3. Original Sentence Format in the Train/Test Files

As demonstrated in Figure 4, we use the tag <boundary> to indicate the target field which will be trained in HMM. Because there is only one target type in the SBD and this target field has just one token, we can pre-define a HMM model with simple format like simple11, and let the HMM package efficiently derive the parameters for this model.

BEVERLY HILLS hardly seems the place for the little guy to get an even break, but some small business owners recently struck a blow for equality in this ghetto of glitz <boundary>. </boundary>
The arena for this victory was the Beverly Hills Chamber of Commerce <boundary>. </boundary>
For
ENDOFDOC

Figure 4. A Document Unit for the HMM Package

4.4 Maximum Entropy

We implement a Maximum Entropy model for the SBD problem. It is not domain specific, except some basic assumptions about English language, such as the following punctuations can serve as the sentence full stop -- !, . , ?, or ”. Some other assumptions are made to make the features more discriminative.

However, within the framework of ME, they are easy to model and change, and to be automatically selected and weighted.

As mentioned above, in Maximum Entropy Modeling, feature values are determined by some input context and the classification. In theory, the whole document or data collection may be meaningful context. In practice, because of the limitation of computation complexity and data sparseness, contexts are restricted to local neighbor tokens. In our implementation, context includes one word precedes and two words follow the putative sentence boundary, which we call the Candidate. The words immediate precede and follow the Candidate are called the Left and the Right respectively. The word second to the right of the Candidate is called the Tail. Context is expandable as needed by new features.

We decide to mainly use lexical and syntactical features, which is more stable and does not suffer the sparseness of the data such as a n-gram model. Also, the Generalized Iterative Scaling (GIS) algorithm we use may suffer from slow convergence in case of a large number of features. Some relate research work has been done to address this problem (Mikheev,1998), which requires efforts beyond the scope of this project.

We also augment the lexical features with some small amount of external word specific resources, such as a list of 88,799 most common last names, 5494 most common first names from 90' US Census and 20 common honorifics. We automatically extract abbreviations from the training and test set following the Document Centered Approach by Mikheev (2000). All the above aids are somewhat helpful but not essential.

We started from a few manually created features, and built the optimal ME model using GIS algorithm. The model was then validated on the holdout data set and more rules were added to deal with misclassified cases. Given the small number of features, the program usually converges very fast. Since inter-feature dependency is not expected to be a problem in ME modeling, adding new features is quite easy. The system quickly achieved F-measure (based on the SBD) of better than 98% with 13 features.

We further explored an algorithm called Inductive Learning (Berger, 1996), which helps automatically build a model with a small set of most efficient features out of a pool of candidate features. It is essentially a greedy algorithm, which always chooses the next promising feature, adding the largest increase of log-likelihood of training data. The addition of new features stops when the F measure on the holdout data set does not improve any more.

Four more arbitrary features are added to the feature set and the algorithm is run on the set of seventeen candidate features. The algorithm selects eight features, all from the original thirteen features, and build a ME model which performs slightly better. The following is the list of eight features selected by the algorithm. They are sorted on the order selected by the Inductive Learning algorithm.

1. Left is a lowercased word, sentence boundary (SB);
2. Right is a lowercased word, NSB;
3. Right is '.', '?', ',', '',), }, -, NSB;
4. Left is an honorific, Candidate is '.', NSB;
5. Candidate is '', an odd quote, NSB;
5. Left is an initial, Candidate is '.', and not a sentence boundary (NSB);
7. Left is ',', Candidate is '', NSB;
8. Candidate is '.', Right is ', Tail is 's', NSB;

5. PERFORMANCE

We evaluated the performance of the three methods using standard precision, recall, F-measure and error rate in the following Table -1. Precision is defined as the total number of correctly extracted sentence boundaries over the total number of boundaries extracted by the system. Recall is defined as the total number of correctly extracted sentence boundaries over the total number true boundaries existing in the collection. F-measure is defined as the harmonic mean of precision and recall (2 times the sum of the inverse of precision and the inverse of recall), which is a good one-number indicator of systems performance. Error rate is also a popular measure, defined as the sum of false negatives and false positives over all possible sentence boundaries. One thing to note is that error rate may not be defined

exactly the same since the definition of possible sentence boundary sometimes is different between authors. In the following table, the definition of error rate for rule-based system and ME system are the same, including all cases of '.', '!', '?', and " in the test set, after trivial periods within a percentage number, monetary value, or a real number are replaces by the same tokenizer. The HMM package we use has its own tokenizer, which removes all double quotes, thus, its error rate is defined slightly different.

Method	Precision	Recall	F1	Error Rate
RuleBased	99.56%	76.95%	86.81%	16.25%
HMM	91.43%	94.46%	92.92%	10.00%
MaxEnt	99.16%	97.62%	98.38%	1.99%

Table-1 Performance comparison of three methods

From the above, we can see, ME modeling performs the best. It is not a surprise to us for the following reason.

Rule-based system can capture most cases precisely, with highest precision. However, to deal with less and less uncommon cases, the number of rules increases quickly and it is hard to balance between conflicting rules. We do not come up with a large number of rules in it; thus, it falls short on recall.

HMM is essentially an enhanced bi-gram model, with strong independence assumptions between transition and emission. Even though more target internal states, background states and better context modeling somewhat help, HMM can not easily model some of the above highly effective complex features in SBD. It can indeed model simple lexical features using feature decompositions.

ME is an excellent framework to integrate different complex features from heterogeneous knowledge sources. Lexical, syntactical features and bi-grams can be naturally modeled as features in a same model. For example, in the following sequence, *his compromise bill . " A committee staffer* is highly ambiguous. We cannot tell from the above local context, if the period or the double quote should be the sentence boundary. However, in ME, we add the parity of double quotes as a feature, which is highly effective, but cannot be derived from local context only.

Also, from the list of features in section 4.4 we can see, ME models not only can easily capture different complex features, but also handles overlapping features very well. We have noticed that even though ME can automatically weight different features, it does not simulate very well the discriminative power combined from several simple features. In such cases, a complex feature of logical compositions of simple features is added into the model, instead of a number of simple features only. The simple features can coexist in the model with the complex feature to compensate information not coming from the complex feature. And this type of feature modeling not only produces better prediction precision, but also helps the feature selection algorithm such as Inductive Learning. Inductive Learning is a greedy algorithm and only adds features one by one. Thus, a few simple features, which work together only, may be missed by the algorithm since any one of them alone does not bring in much valuable information to explain the training data.

Another potential problem is the need of smoothing. Some rare but useful features are not estimated reliably in a particular training set. In such cases, we have tried to group similar situations into one single feature, which has more instances in the training set (e.g., instead of having a separate feature to model a period is unlikely to be a SB if Right is '}', we have model the feature together with several other punctuations as in the above feature 3 in section 4.4).

Finally, model building can be very computational expensive. Using Generalized Iterative Scaling, training a ME with a set of seventeen features on our training set took about 10 minutes on a single CPU of Sun Blade 2000. It took about 90 minutes for Inductive Learning algorithm to train a ME model selecting the best eight features from the seventeen features.

The SBD is a well-defined problem; and with a very limited number of features, ME achieves reasonable performance. Figure-5 and Table-2 show the performance improvement with the growth of number of features selected by the Inductive

Learning algorithm. We can see the ME model achieves very good performance quickly with only 6 features.

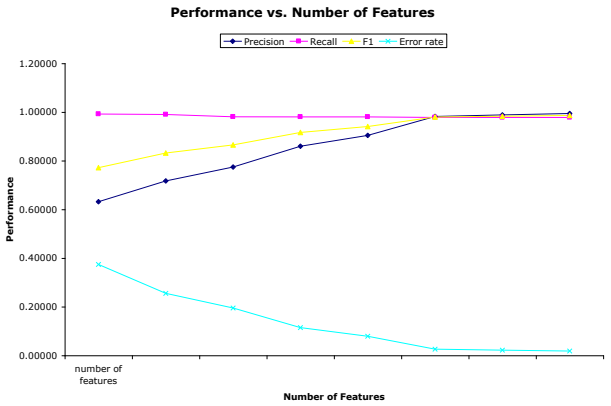

Figure-5 ME performance vs. number of features on holdout data

Number of features	1	2	3	4	5	6	7	8
Precision	63.1%	71.7%	77.4%	86.0%	90.4%	98.2%	98.8%	99.4%
Recall	99.2%	99.0%	98.0%	98.0%	98.0%	97.8%	97.8%	97.8%
F1	77.2%	83.2%	86.5%	91.6%	94.1%	98.0%	98.3%	98.6%
Error rate	37.4%	25.5%	19.5%	11.4%	7.9%	2.6%	2.2%	1.8%

Table-2 ME performance vs. number of features on holdout data

6. CONCLUSION

Not only can ME models capture different complex features easily, but also Generalized Iterative Scaling handles overlapping features very well. This advantage is what many other algorithms, such as Naïve Bayes and HMM, do not have. So even it is one of many log-linear classifiers, it has its unique strength.

The training of ME models can be computational expensive due to the possible slow convergence of iterative numerical method, GIS. Thus, feature modeling and selection are potentially important as well. For SBD problem, we have demonstrated that ME models can achieve decent performance using only a few well-modeled features.

The feature using abbreviation information is not selected by the IL algorithm, which means it does not help the model working with other existing features. As given in an example earlier in the introduction, it is expected to be useful pairing with knowledge to tell if a trailing capitalized word is a proper name or a common word. Due to the time we have, we did not implement such a feature. It can be a source of future improvements for our system on SBD.

We did not get the chance to train our systems on a larger corpus such as the whole Wall Street Journal Corpus (WSJ) and the Brown Corpus. It will be interesting to see how well the above features capture the SBD problem in a new corpus.

ACKNOWLEDGMENTS

We appreciate the help from Dr. Palmer, who provided the train and test data for our project. We would also like to thank Professor Christopher Manning, Rajat Raina, and Kristina Toutanova for their great work and efforts in the instruction of class CS224n - Natural Language Processing.

REFERENCES

[1] Aberdeen, J., J. Burger, D. Day, L. Hirschmann, P. Robinson, and M. Vilain. 1995. *Description of the alembic system used for muc-6.* In Proceedings of the Sixth Message Understanding Conference (MUC-6). Morgan Kaufmann.

[2] Berger A. 1996. *A Brief Maxent Tutorial*. http://www-2.cs.cmu.edu/~aberger/maxent.html.

[3] Berger A. 1997. *The improved iterative scaling algorithm: a gentle introduction*. http://www-2.cs.cmu.edu/~aberger/maxent.html.

[4] Manning, C.D. and H. Schütze. 2002. *Foundations of statistical natural language processing*. The MIT Press, Cambridge/London.

[5] Mikheev, A. 1998, *Feature Lattices and Maximum Entropy Models*.

[6] Mikheev, A. 2000. *Tagging Sentence Boundaries*. In NACL'2000 (Seattle) ACL, pp. 264 – 271.

[7] Mikheev, A. 2000. *Document Centered Approach to Text Normalization*. In SIGIR'2000 (Athens) ACM June 2000. pp. 136—143.

[8] Palmer, D.D. and M.A. Hearst. 1997. *Adaptive multilingual sentence boundary disambiguation*. Computational Linguistics, 23/3, pp. 241 – 267.

[9] Reynar, J.C. and A. Ratnaparkhi. 1997. *A Maximum Entropy Approach to Identifying Sentence Boundaries*. In Processing of the ANLP97, Washington, D.C.

[10] Shannon C.E. 1948. *A mathematical theory of communication*. Bell System Technical Journal 27:379 – 423, 623 – 656.