Updating an NLP System to Fit New Domains: an empirical study on the sentence segmentation problem

Tong Zhang† and **Fred Damerau** ‡ and **David Johnson**§

IBM T.J. Watson Research Center
Yorktown Heights
New York, 10598, USA

†tzhang@watson.ibm.com ‡damerau@watson.ibm.com §dejohns@us.ibm.com

Abstract

Statistical machine learning algorithms have been successfully applied to many natural language processing (NLP) problems. Compared to manually constructed systems, statistical NLP systems are often easier to develop and maintain since only annotated training text is required. From annotated data, the underlying statistical algorithm can build a model so that annotations for future data can be predicted. However, the performance of a statistical system can also depend heavily on the characteristics of the training data. If we apply such a system to text with characteristics different from that of the training data, then performance degradation will occur. In this paper, we examine this issue empirically using the sentence boundary detection problem. We propose and compare several methods that can be used to update a statistical NLP system when moving to a different domain.

1 Introduction

An important issue for a statistical machine learning based NLP system is that its performance can depend heavily on the characteristics of the training data used to build the system. Consequently if we train a system on some data but apply it to other data with different characteristics, then the system's performance can degrade significantly. It is therefore natural to investigate the following related issues:

- How to detect the change of underlying data characteristics, and to estimate the corresponding system performance degradation.
- If performance degradation is detected, how to update a statistical system to improve its performance with as little human effort as possible.

This paper investigates some methodological and practical aspects of the above issues. Although ideally such a study would include as many different statistical algorithms as possible, and as many different linguistic problems as possible (so that a very general conclusion might be drawn), in reality such an undertaking is not only difficult to carry out, but also can hide essential observations and obscure important effects that may depend on many variables. An alternative is to study a relatively simple and well-understood problem to try to gain understanding of the fundamental issues. Causal effects and essential observations can be more easily isolated and identified from simple problems since there are fewer variables that can affect the outcome of the experiments.

In this paper, we take the second approach and focus on a specific problem using a specific underlying statistical algorithm. However, we try to use only some fundamental properties of the algorithm so that our methods are readily applicable to other systems with similar properties. Specifically, we use the sentence boundary detection problem to perform experiments since not only is it relatively simple and well-understood, but it also provides the basis for other more advanced linguistic problems. Our hope is that some characteristics of this problem are universal to language processing so that they can be generalized to more complicated linguistic tasks. In this paper we use the generalized Winnow method (Zhang et al., 2002) for all experiments. Applied to text chunking, this method resulted in state of the art performance. It is thus reasonable to conjecture that it is also suitable to other linguistic problems including sentence segmentation.

Although issues addressed in this paper are very important for practical applications, there have only been limited studies on this topic in the existing literature. In speech processing, various adaption techniques have been proposed for language modeling. However, the language modeling problem is essentially unsupervised (density estimation) in the sense that it does not require any annotation. Therefore techniques developed there cannot be applied to our problems. Motivated from adap-

tive language modeling, transformation based adaptation techniques have also been proposed for certain supervised learning tasks (Gales and Woodland, 1996). However, typically they only considered very specific statistical models where the idea is to fit certain transformation parameters. In particular they did not consider the main issues investigated in this paper as well as generally applicable supervised adaptation methodologies such as what we propose. In fact, it will be very difficult to extend their methods to natural language processing problems that use different statistical models. The adaption idea in (Gales and Woodland, 1996) is also closely related to the idea of combining supervised and unsupervised learning in the same domain (Merialdo, 1994). In machine learning, this is often referred to as semi-supervised learning or learning with unlabeled data. Such methods are not always reliable and can often fail(Zhang and Oles, 2000). Although potentially useful for small distributional parameter shifts, they cannot recover labels for examples not (or inadequately) represented in the old training data. In such cases, it is necessary to use supervised adaption methods which we study in this paper. Another related idea is so-called active learning paradigm (Lewis and Catlett, 1994; Zhang and Oles, 2000), which selectively annotates the most informative data (from the same domain) so as to reduce the total number of annotations required to achieve a certain level of accuracy. See (Tang et al., 2002; Steedman et al., 2003) for related studies in statistical natural language parsing.

2 Generalized Winnow for Sentence Boundary Detection

For the purpose of this paper, we consider the following form of the sentence boundary detection problem: to determine for each period "." whether it denotes a sentence boundary or not (most non-sentence boundary cases occur in abbreviations). Although other symbols such as "?" and "!" may also denote sentence boundaries, they occur relatively rarely and when they occur, are easy to determine. There are a number of special situations, for example: three (or more) periods to denote omission, where we only classify the third period as an end of sentence marker. The treatment of these special situations are not important for the purpose of this paper.

The above formulation of the sentence segmentation problem can be treated as a binary classification problem. One method that has been successfully applied to a number of linguistic problems is the Winnow algorithm (Littlestone, 1988; Khardon et al., 1999). However, a drawback of this method is that the algorithm does not necessarily converge for data that are not linearly separable. A generalization was recently proposed, and applied to the text chunking problem (Zhang et al., 2002), where it was shown that this generalization can indeed improve the performance of Winnow.

Applying the generalized Winnow algorithm on the sentence boundary detection problem is straight forward since the method solves a binary classification problem directly. In the following, we briefly review this algorithm, and properties useful in our study.

Consider the binary classification problem: to determine a label $y \in \{-1, 1\}$ associated with an input vector x. A useful method for solving this problem is through linear discriminant functions, which consist of linear combinations of components of the input vector. Specifically, we seek a weight vector w and a threshold θ with the following decision rule: if $w^T x < \theta$ we predict that the label $y = -1$, and if $w^T x \geq \theta$, we predict that the label $y = 1$. We denote by d the dimension of the weight vector w which equals the dimension of the input vector x. The weight w and threshold θ can be computed from the generalized Winnow method, which is based on the following optimization problem:

$$
\begin{aligned}
(w, \theta) = \arg \min_{w^+, w^-} \Bigg[& \sum_{j=1}^{d} w_j^+ \ln \frac{w_j^+}{e\mu} + \theta^+ \ln \frac{\theta^+}{e\mu} \\
& + \sum_{j=1}^{d} w_j^- \ln \frac{w_j^-}{e\mu} + \theta^- \ln \frac{\theta^-}{e\mu} \\
& + c \sum_{i=1}^{n} f((w^T x^i - \theta) y^i) \Bigg], \qquad (1) \\
\text{s.t. } & w = w^+ - w^-, \theta = \theta^+ - \theta^-,
\end{aligned}
$$

where

$$
f(v) = \begin{cases} -2v & v < -1 \\ \frac{1}{2}(v-1)^2 & v \in [-1, 1] \\ 0 & v > 1. \end{cases}
$$

The numerical method which we use to solve this problem, as presented in Algorithm 1, is based on a dual formulation of the above problem. See (Zhang et al., 2002) for detailed derivation of the algorithm and its relationship with the standard Winnow.

In all experiments, we use the same parameters suggested in (Zhang et al., 2002) for the text chunking problem: $K = 40, \mu = 0.1, \eta = 0.01$, and $c = 0.1$. The above parameter choices may not be optimal for sentence segmentation. However since the purpose of this paper is not to demonstrate the best possible sentence segmentation system using this approach, we shall simply fix these parameters for all experiments.

Algorithm 1 *(Generalized Winnow)*

input: training data $(x^1, y^1), \ldots, (x^n, y^n)$
output: weight vector w and threshold θ
let $\alpha_i = 0$ $(i = 1, \ldots, n)$
let $w_j^+ = w_j^- = \mu$ $(j = 1, \ldots, d)$
let $\theta^+ = \theta^- = \mu$
for $k = 1, \ldots, K$
  for $i = 1, \ldots, n$
    $p = (w^+ - w^-)^T x^i y^i - (\theta^+ - \theta^-) y^i$
    $\Delta\alpha^i = \max(\min(2c - \alpha^i, \eta(\frac{c - \alpha^i}{c} - p)), -\alpha^i)$
    $w_j^+ = w_j^+ \exp(\Delta\alpha^i x_j^i y^i)$ $(j = 1, \ldots, d)$
    $w_j^- = w_j^- \exp(-\Delta\alpha^i x_j^i y^i)$ $(j = 1, \ldots, d)$
    $\theta^+ = \theta^+ \exp(-\Delta\alpha^i y^i)$
    $\theta^- = \theta^- \exp(\Delta\alpha^i y^i)$
    $\alpha^i = \alpha^i + \Delta\alpha^i$
  end
end
let $w = w^+ - w^-$
let $\theta = \theta^+ - \theta^-$

It was shown in (Zhang et al., 2002) that if (w, θ) is obtained from Algorithm 1, then it also approximately minimizes $E_x(2P(y = 1|x) - 1 - T(w^T x - \theta))^2$, where $P(y = 1|x)$ denotes the conditional probability of $y = 1$ at a data point x. Here we have used $T(p)$ to denote the truncation of p onto $[-1, 1]$: $T(p) = \min(1, \max(-1, p))$. This observation implies that the quantity $(T(w^T x - \theta) + 1)/2$ can be regarded as an estimate for the in-class conditional probability. As we will see, this property will be very useful for our purposes.

For each period in the text, we construct a feature vector x as the input to the generalized Winnow algorithm, and use its prediction to determine whether the period denotes a sentence boundary or not. In order to construct x, we consider linguistic features surrounding the period, as listed in Table 1. Since the feature construction routine is written in the Java language, "type of character" features correspond to the Java character types, which can be found in any standard Java manual. We picked these features by looking at features used previously, as well as adding some of our own which we thought might be useful. However, we have not examined which features are actually important to the algorithm (for example, by looking at the size of the weights), and which features are not.

We use an encoding scheme similar to that of (Zhang et al., 2002). For each data point, the associated features are encoded as a binary vector x. Each component of x corresponds to a possible feature value v of a feature f in Table 1. The value of the component corresponds to a test which has value one if the corresponding feature f has value v, or value zero if the corresponding feature f has another feature value.

token before the period
token after the period
character to the right
type of character to the right
character to the left
type of character to the left
character to the right of blank after word
type of character to the right of blank after word
character left of first character of word
type of character left of first character of word
first character of the preceding word
type of first character of the preceding word
length of preceding word
distance to previous period

Table 1: Linguistic Features

The features presented here may not be optimal. In particular, unlike (Zhang et al., 2002), we do not use higher order features (for example, combinations of the above features). However, this list of features has already given good performance, comparing favorably with previous approaches (see (Reynar and Ratnaparkhi, 1997; Mikheev, 2000) and references therein).

The standard evaluation data is the Wall-Street Journal (WSJ) tree-bank. Based on our processing scheme, the training set contains about seventy-four thousand periods, and the test set contains about thirteen thousand periods. If we train on the training set, and test on the test set, the accuracy is 99.7%. Another data set which has been annotated is the Brown corpus. If we train on the WSJ training set, and test on the Brown corpus, the accuracy is 99.2%. The error rate is three times larger.

3 Experimental Design and System Update Methods

In our study of system behavior under domain changes, we have also used manually constructed rules to filter out some of the periods. The specific set of rules we have used are:

- If a period terminates a non-capitalized word, and is followed by a blank and a capitalized word, then we predict that it is a sentence boundary.
- If a period is both preceded and followed by alphanumerical characters, then we predict that it is not a sentence boundary.

The above rules achieve error rates of less than 0.1% on both the WSJ and Brown datasets, which is sufficient for our purpose. Note that we did not try to make the above rules as accurate as possible. For example, the first

rule will misclassifiy situations such as "A vs. B". Eliminating such mistakes is not essential for the purpose of this study.

All of our experiments are performed and reported on the remaining periods that are not filtered out by the above manual rules. In this study, the filtering scheme serves two purposes. The first purpose is to magnify the errors. Roughly speaking, the rules will classify more than half of the periods. These periods are also relatively easy to classify using a statistical classifier. Therefore the error rate on the remaining periods is more than doubled. Since the sentence boundary detection problem has a relatively small error rate, this magnification effect is useful for comparing different algorithms. The second purpose is to reduce our manual labeling effort. In this study, we had used a number of datasets that are not annotated. Therefore for experimentation purpose, we have to label each period manually.

After filtering, the WSJ training set contains about twenty seven thousand data points, and the test set contains about five thousand data points. The Brown corpus contains about seventeen thousand data points. In addition, we also manually labeled the following data:

- Reuters: This is a standard dataset for text categorization, available from *http://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html*. We only use the test-data in the ModApte split, which contains about eight thousand periods after filtering.

- MedLine: Medical abstracts with about seven thousand periods, available from *www1.ics.uci.edu/~mlearn/MLRepository.html*.

It is perhaps not surprising that a sentence boundary classifier trained on WSJ does not perform nearly as well on some of the other data sets. However it is useful to examine the source of these extra errors. We observed that most of the errors are clearly caused by the fact that other domains contain examples that are not represented in the WSJ training set. There are two sources for these previously unseen examples: 1. change of writing style; 2. new linguistic expressions. For example, quote marks are represented as two single quote (or back quote) characters in WSJ, but typically as one double quote character elsewhere. In some data sets such as Reuters, phrases such as "U.S. Economy" or "U.S. Dollar" frequently have the word after the country name capitalized (they also appear in lower case sometimes, in the same data). The above can be considered as a change of writing style. In some other cases, new expressions may occur. For example, in the MedLine data, new expressions such as "4 degrees C." are used to indicate temperature, and expressions such as "Bioch. Biophys. Res. Commun. 251, 744-747" are used for citations. In addition, new acronyms and even formulas containing tokens ending with periods occur in such domains.

It is clear that the majority of errors are caused by data that are not represented in the training set. This fact suggests that when we apply a statistical system to a new domain, we need to check whether the domain contains a significant number of previously unseen examples which may cause performance deterioration. This can be achieved by measuring the similarity of the new test domain to the training domain. One way is to compute statistics on the training domain, and compare them to statistics computed on the new test domain; another way is to calculate a properly defined distance between the test data and the training data. However, it is not immediately obvious what data statistics are important for determining classification performance. Similarly it is not clear what distance metric would be good to use. To avoid such difficulties, in this paper we assume that the classifier itself can provide a confidence measure for each prediction, and we use this information to estimate the classifier's performance.

As we have mentioned earlier, the generalized Winnow method approximately minimizes the quantity $E_x(2P(y=1|x)-1-T(w^Tx-\theta))^2$. It is thus natural to use $(T(w^Tx-\theta)+1)/2$ as an estimate of the conditional probability $P(y=1|x)$. From simple algebra, we obtain an estimate of the classification error as $E_x|1-T(w^Tx-\theta)|/2$. Since $T(w^Tx-\theta)$ is only an approximation of the conditional probability, this estimate may not be entirely accurate. However, one would expect it to give a reasonably indicative measure of the classification performance. In Table 2, we compare the true classification accuracy from the annotated test data to the estimated accuracy using this method. It clearly shows that this estimate indeed correlates very well with the true classification performance. Note that this estimate does not require knowing the true labels of the data. Therefore we are able to detect the potential performance degradation of the classifier on a new domain using this metric without the ground truth information.

accuracy	WSJ	Brown	Reuters	MedLine
true	99.3	97.7	93.0	94.8
estimated	98.6	98.2	93.3	96.4

Table 2: True and estimated accuracy

As pointed out before, a major source of error for a new application domain comes from data that are not represented in the training set. If we can identify those data, then a natural way to enhance the underlying classifier's performance would be to include them in the training data, and then retrain. However, a human is required

to obtain labels for the new data, but our goal is to reduce the human labeling effort as much as possible. Therefore we examine the potential of using the classifier to determine which part of the data it has difficulty with, and then ask a human to label that part. If the underlying classifier can provide confidence information, then it is natural to assume that confidence for unseen data will likely be low. Therefore for labeling purposes, one can choose data from the new domain for which the confidence is low. This idea is very similar to certain methods used in active learning. In particular a confidence-based sample selection scheme was proposed in (Lewis and Catlett, 1994). One potential problem for this approach is that by choosing data with lower confidence levels, noisy data that are difficult to classify tend to be chosen; another problem is that it tends to choose similar data multiple times. However, in this paper we do not investigate methods that solve these issues.

For baseline comparison, we consider the classifier obtained from the old training data (see Table 3), as well as classifiers trained on random samples from the new domain (see Table 4). In this study, we explore the following three ideas to improve the performance:

- Data balancing: Merge labeled data from the new domain with the existing training data from the old domain; we also balance their relative proportion so that the effect of one domain does not dominate the other.
- Feature augmentation: Use the old classifier (first level classifier) to create new features for the data, and then train another classifier (second level classifier) with augmented features (on newly labeled data from the new domain).
- Confidence based feature selection: Instead of random sampling, select data from the new domain with lowest confidence based on the old classifier.

One may combine the above ideas. In particular, we will compare the following methods in this study:

- Random: Randomly selected data from the new domain.
- Balanced: Use WSJ training set + randomly selected data from the new domain. However, we supersample the randomly selected data so that the effective sample size is β-times that of the WSJ training set, where β is a balancing factor.
- Augmented (Random): Use the default classifier output to form additional features. Then train a second level classifier on randomly selected data from the new domain, with these additional features. In our experiments, four binary features are added; they correspond to tests $c > 1, c > 0, c < 0, c < -1$ (where $c = w^T x - \theta$ is the output of the first level classifier).
- Augmented-balanced: As indicated, use additional features as well as the original WSJ training set for the second level classifier.
- Confidence-Balanced: Instead of random sampling from the new domain, choose the least confident data (which is more likely to provide new information), and then balance with the WSJ training set.
- Augmented-Confidence-Balanced: This method is similar to Augmented-balanced. However, we label the least confident data instead of random sampling.

4 Experimental Results

We carried out experiments on the Brown, Reuters, and MedLine datasets. We randomly partition each dataset into training and testing. All methods are trained using only information from the training set, and their performance are evaluated on the test set. Each test set contains 4000 data points randomly selected. This sample size is chosen to make sure that an estimated accuracy based on these empirical samples will be reasonably close to the true accuracy. For a binary classifier, the standard deviation between the empirical mean $\hat{q}$ with a sample size $m = 4000$, and the true mean $\bar{q}$, is $\sqrt{\bar{q}(1-\bar{q})/m}$. Since $\hat{q} \approx \bar{q}$, we can replace $\bar{q}$ by $\hat{q}$. Now, if $\hat{q} \geq 0.9$, then the error is less than 0.5%; if $\hat{q} \geq 0.98$, then the standard deviation is no more than about 0.2%. From the experiments, we see that the accuracy of all algorithms will be improved to about 0.98 for all three datasets. Therefore the test set size we have is sufficiently large to distinguish a difference of 0.5% with reasonable confidence.

Table 3 lists the test set performance of classifiers trained on the WSJ training set (denoted by WSJ), the training set from the same domain (that is, Brown, Reuters, and MedLine respectively for the corresponding testsets), denoted by Self, and their combination. This indicates upper limits on what can be achieved using the corresponding training set information. It is also interesting to see that the combination does not necessarily improve the performance. We compare different updating schemes based on the number of new labels required from the new domain. For this purpose, we use the following number of labeled instances: $100, 200, 400, 800$ and 1600, corresponding to the "new data" column in the tables. For all experiments, if a specific result requires random sampling, then five different random runs were performed, and the corresponding result is reported in the format of "mean $\pm$ std. dev." over the five runs.

Table 4 contains the performance of classifiers trained on randomly selected data from the new domain alone. It

trainset	Brown	Reuters	MedLine
WSJ	97.5	93.1	94.6
Self	99.1	98.4	98.2
WSJ+Self	98.9	98.9	97.9

Table 3: baseline accuracy

is interesting to observe that even with a relatively small number of training examples, the corresponding classifiers can out-perform those obtained from the default WSJ training set, which contains a significantly larger amount of data. Clearly this indicates that in some NLP applications, using data with the right characteristics can be more important than using more data. This also provides strong evidence that one should update a classifier if the underlying domain is different from the training domain.

new data	Brown	Reuters	MedLine
100	94.5 ± 0.9	94.8 ± 1.4	93.2 ± 1.1
200	94.8 ± 1.2	95.8 ± 0.9	95.5 ± 0.6
400	96.8 ± 0.3	96.8 ± 0.4	96.6 ± 0.4
800	97.2 ± 0.5	97.6 ± 0.1	97.2 ± 0.2
1600	97.9 ± 0.1	98.0 ± 0.1	97.8 ± 0.2

Table 4: Random Selection

Table 5 contains the results of using the balancing idea. With the same amount of newly labeled data, the improvement over the random method is significant. This shows that even though the domain has changed, training data from the old domain are still very useful. Observe that not only is the average performance improved, but the variance is also reduced. Note that in this table, we have fixed $\beta = 0.5$. The performance with different β values on the MedLine dataset is reported in Table 6. It shows that different choices of β make relatively small differences in accuracy. At this point, it is interesting to check whether the estimated accuracy (using the method described for Table 2) reflects the change in performance improvement. The result is given in Table 7. Clearly the method we propose still leads to reasonable estimates.

new data	Brown	Reuters	MedLine
100	97.7 ± 0.1	97.1 ± 0.6	95.8 ± 0.4
200	97.9 ± 0.2	97.7 ± 0.3	96.6 ± 0.3
400	97.9 ± 0.1	98.1 ± 0.3	97.2 ± 0.2
800	98.1 ± 0.2	98.3 ± 0.3	97.6 ± 0.2
1600	98.4 ± 0.1	98.7 ± 0.1	97.9 ± 0.1

Table 5: Balanced ($\beta = 0.5$)

Table 8 and Table 9 report the performance using

β	1/8	1/4	1/2	1	2
100	96.0	95.7	95.8	96.0	94.9
200	96.3	96.5	96.6	96.3	96.6
400	96.8	97.0	97.2	97.1	96.8
800	97.3	97.5	97.6	97.5	97.4
1600	97.4	97.8	97.9	98.0	97.7

Table 6: Effect of β on MedLine using the balancing scheme

accuracy	Brown	Reuters	MedLine
true	98.1	98.3	97.6
estimated	98.4	97.9	98.2

Table 7: True and estimated accuracy (balancing scheme with 800 samples and $\beta = 0.5$)

augmented features, either with the random sampling scheme, or with the balancing scheme. It can be seen that with feature augmentation, the random sampling and the balancing schemes perform similarly. Although the feature augmentation method does not improve the overall performance (compared with balancing scheme alone), one advantage is that we do not have to rely on the old training data any more. In principle, one may even use a two-level classification scheme: use the old classifier if it gives a high confidence; use the new classifier trained on the new domain otherwise. However, we have not explored such combinations.

new data	Brown	Reuters	MedLine
100	97.5 ± 0.0	97.7 ± 0.2	95.5 ± 1.0
200	97.6 ± 0.1	97.6 ± 0.3	95.9 ± 0.8
400	97.7 ± 0.1	97.8 ± 0.2	97.0 ± 0.9
800	97.8 ± 0.1	98.1 ± 0.4	97.6 ± 0.3
1600	98.1 ± 0.1	98.3 ± 0.3	97.9 ± 0.1

Table 8: Augmented (Random)

Table 10 and Table 11 report the performance using confidence based data selection, instead of random sampling. This method helps to some extent, but not as much as we originally expected. However, we have only used the simplest version of this method, which is susceptible to two problems mentioned earlier: it tends (a) to select data that are inherently hard to classify, and (b) to select redundant data. Both problems can be avoided with a more elaborated implementation, but we have not explored this. Another possible reason that using confidence based sample selection does not result in significant performance improvement is that for our examples, the performance is already quite good with even a small number of new samples.

new data	Brown	Reuters	MedLine
100	97.8 ± 0.3	97.0 ± 1.0	95.4 ± 0.7
200	97.8 ± 0.2	97.7 ± 0.3	95.9 ± 0.6
400	98.0 ± 0.1	98.0 ± 0.3	96.8 ± 0.6
800	98.2 ± 0.3	98.4 ± 0.3	97.2 ± 0.3
1600	98.4 ± 0.2	98.7 ± 0.3	97.5 ± 0.2

Table 9: Augmented + Balanced

new data	Brown	Reuters	MedLine
100	98.0	97.5	96.9
200	98.1	97.4	97.0
400	98.2	97.8	97.6
800	98.7	98.6	98.0
1600	98.8	98.8	98.0

Table 10: Confidence + Balanced

5 Conclusion

In this paper, we studied the problem of updating a statistical system to fit a domain with characteristics different from that of the training data. Without updating, performance will typically deteriorate, perhaps quite drastically.

We used the sentence boundary detection problem to compare a few different updating methods. This provides useful insights into the potential value of various ideas. In particular, we have made the following observations: 1. An NLP system trained on one data set can perform poorly on another because there can be new examples not adequately represented in the old training set; 2. It is possible to estimate the degree of system performance degradation, and to determine whether it is necessary to perform a system update; 3. When updating a classifier to fit a new domain, even a small amount of newly labeled data can significantly improve the performance (also, the right training data characteristics can be more important than the quantity of training data); 4. Combining the old training data with the newly labeled data in an appropriate way (e.g., by balancing or feature augmentation) can be effective.

Although the sentence segmentation problem considered in this paper is relatively simple, we are currently investigating other problems. We anticipate that the observations from this study can be applied to more complicated NLP tasks.

new data	Brown	Reuters	MedLine
100	97.3	97.8	96.9
200	97.8	97.7	96.9
400	98.1	97.7	97.6
800	98.7	98.6	98.1
1600	98.8	98.9	98.2

Table 11: Augmented + Confidence + Balanced

References

M.J. Gales and P.C. Woodland. 1996. Mean and variance adaptation within the MLLR framework. *Computer Speech and Language*, 10:249–264.

R. Khardon, D. Roth, and L. Valiant. 1999. Relational learning for NLP using linear threshold elements. In *Proceedings IJCAI-99*.

D. Lewis and J. Catlett. 1994. Heterogeneous uncertainty sampling for supervised learning. In *Proceedings of the Eleventh International Conference on Machine Learning*, pages 148–156.

N. Littlestone. 1988. Learning quickly when irrelevant attributes abound: a new linear-threshold algorithm. *Machine Learning*, 2:285–318.

Bernard Merialdo. 1994. Tagging english text with a probabilistic model. *Computational Linguistics*, 20:155–171.

A. Mikheev. 2000. Tagging sentence boundaries. In *NACL'2000*, pages 264–271.

J. Reynar and A. Ratnaparkhi. 1997. A maximum entropy approach to identifying sentence boundaries. In *Proceedings of the Fifth Conference on Applied Natural Language Processing*, pages 16–19.

M. Steedman, R. Hwa, S. Clark, M. Osborne, A. Sarkar, J. Hockenmaier, P. Ruhlen, S. Baker, and J. Crim. 2003. Example selection for bootstrapping statistical parsers. In *NAACL*. to appear.

M. Tang, X. Luo, and S. Roukos. 2002. Active learning for statistical natural language parsing. In *Proceedings of the Association for Computational Linguistics 40th Anniversary Meeting*, pages 120–127.

Tong Zhang and Frank J. Oles. 2000. A probability analysis on the value of unlabeled data for classification problems. In *ICML 00*, pages 1191–1198.

Tong Zhang, Fred Damerau, and David E. Johnson. 2002. Text chunking based on a generalization of Winnow. *Journal of Machine Learning Research*, 2:615–637.