Algorithm Selection and Model Adaptation for ESL Correction Tasks

Alla Rozovskaya and Dan Roth
University of Illinois at Urbana-Champaign
Urbana, IL 61801
{rozovska,danr}@illinois.edu

Abstract

We consider the problem of correcting errors made by English as a Second Language (ESL) writers and address two issues that are essential to making progress in ESL error correction - algorithm selection and model adaptation to the first language of the ESL learner.

A variety of learning algorithms have been applied to correct ESL mistakes, but often comparisons were made between incomparable data sets. We conduct an extensive, fair comparison of four popular learning methods for the task, reversing conclusions from earlier evaluations. Our results hold for different training sets, genres, and feature sets.

A second key issue in ESL error correction is the adaptation of a model to the first language of the writer. Errors made by non-native speakers exhibit certain regularities and, as we show, models perform much better when they use knowledge about error patterns of the non-native writers. We propose a novel way to adapt a learned algorithm to the first language of the writer that is both cheaper to implement and performs better than other adaptation methods.

1 Introduction

There has been a lot of recent work on correcting writing mistakes made by English as a Second Language (ESL) learners (Izumi et al., 2003; Eeg-Olofsson and Knuttson, 2003; Han et al., 2006; Felice and Pulman, 2008; Gamon et al., 2008; Tetreault and Chodorow, 2008; Elghaari et al., 2010; Tetreault et al., 2010; Gamon, 2010; Rozovskaya and Roth, 2010c). Most of this work has focused on correcting mistakes in article and preposition usage, which are some of the most common error types among non-native writers of English (Dalgish, 1985; Bitchener et al., 2005; Leacock et al., 2010). Examples below illustrate some of these errors:

1. "They listen to *None*/the* lecture carefully."
2. "He is an engineer with a passion *to*/for* what he does."

In (1) the definite article is incorrectly omitted. In (2), the writer uses an incorrect preposition.

Approaches to correcting preposition and article mistakes have adopted the methods of the *context-sensitive spelling correction task*, which addresses the problem of correcting spelling mistakes that result in legitimate words, such as confusing *their* and *there* (Carlson et al., 2001; Golding and Roth, 1999). A *candidate set* or a *confusion set* is defined that specifies a list of confusable words, e.g., {*their, there*}. Each occurrence of a confusable word in text is represented as a vector of features derived from a context window around the target, e.g., words and part-of-speech tags. A classifier is trained on text assumed to be error-free. At decision time, for each word in text, e.g. *there*, the classifier predicts the most likely candidate from the corresponding confusion set {*their, there*}.

Models for correcting article and preposition errors are similarly trained on error-free native English text, where the confusion set includes all articles or prepositions (Izumi et al., 2003; Eeg-Olofsson and Knuttson, 2003; Han et al., 2006; Felice and Pulman, 2008; Gamon et al., 2008; Tetreault and Chodorow, 2008; Tetreault et al., 2010).

Although the choice of a particular learning algorithm differs, with the exception of decision trees (Gamon et al., 2008), all algorithms used are linear learning algorithms, some discriminative (Han et al., 2006; Felice and Pulman, 2008; Tetreault and Chodorow, 2008; Rozovskaya and Roth, 2010c; Rozovskaya and Roth, 2010b), some probabilistic (Gamon et al., 2008; Gamon, 2010), or "counting" (Bergsma et al., 2009; Elghaari et al., 2010).

While model comparison has not been the goal of the earlier studies, it is quite common to compare systems, even when they are trained on different data sets and use different features. Furthermore, since there is no shared ESL data set, systems are also evaluated on data from different ESL sources or even on native data. Several conclusions have been made when comparing systems developed for ESL correction tasks. A *language model* was found to outperform a *maximum entropy classifier* (Gamon, 2010). However, the language model was trained on the Gigaword corpus, $17 \cdot 10^9$ words (Linguistic Data Consortium, 2003), a corpus several orders of magnitude larger than the corpus used to train the classifier. Similarly, web-based models built on Google Web1T 5-gram Corpus (Bergsma et al., 2009) achieve better results when compared to a maximum entropy model that uses a corpus $10,000$ times smaller (Chodorow et al., 2007)[1].

In this work, we compare four popular learning methods applied to the problem of correcting preposition and article errors and evaluate on a common ESL data set. We compare two probabilistic approaches – *Naïve Bayes* and *language modeling*; a discriminative algorithm *Averaged Perceptron*; and a count-based method *SumLM* (Bergsma et al., 2009), which, as we show, is very similar to Naïve Bayes, but with a different free coefficient. We train our models on data from several sources, varying training sizes and feature sets, and show that there are significant differences in the performance of these algorithms. Contrary to previous results (Bergsma et al., 2009; Gamon, 2010), we find that when trained on the same data with the same features, Averaged Perceptron achieves the best performance, followed by Naïve Bayes, then the language model, and finally the count-based approach. Our results hold for training sets of different sizes, genres, and feature sets. We also explain the performance differences from the perspective of each algorithm.

The second important question that we address is that of adapting the decision to the source language of the writer. Errors made by non-native speakers exhibit certain regularities. Adapting a model so that it takes into consideration the specific error patterns of the non-native writers was shown to be extremely helpful in the context of discriminative classifiers (Rozovskaya and Roth, 2010c; Rozovskaya and Roth, 2010b). However, this method requires generating new training data and training a separate classifier for each source language. Our key contribution here is a novel, simple, and elegant adaptation method within the framework of the Naïve Bayes algorithm, which yields even greater performance gains. Specifically, we show how the error patterns of the non-native writers can be viewed as *a different distribution on candidate priors* in the confusion set. Following this observation, we train Naïve Bayes in a traditional way, regardless of the source language of the writer, and then, only at decision time, change the prior probabilities of the model from the ones observed in the native training data to the ones corresponding to error patterns in the non-native writer's source language (Section 4). A related idea has been applied in Word Sense Disambiguation to adjust the model priors to a new domain with different sense distributions (Chan and Ng, 2005).

The paper has two main contributions. First, we conduct a fair comparison of four learning algorithms and show that the discriminative approach Averaged Perceptron is the best performing model (Sec. 3). Our results do not support earlier conclusions with respect to the performance of count-based models (Bergsma et al., 2009) and language models (Gamon, 2010). In fact, we show that SumLM is comparable to Averaged Perceptron trained with a *10 times* smaller corpus, and language model is comparable to Averaged Perceptron trained with a *2 times* smaller corpus.

The second, and most significant, of our contributions is a novel way to adapt a model to the source language of the writer, *without re-training the model* (Sec. 4). As we show, adapting to the source language of the writer provides significant performance improvement, and our new method also performs

[1]These two models also use different features.

better than previous, more complicated methods.

Section 2 presents the theoretical component of the linear learning framework. In Section 3, we describe the experiments, which compare the four learning models. Section 4 presents the key result of this work, a novel method of adapting the model to the source language of the learner.

2 The Models

The standard approach to preposition correction is to cast the problem as a multi-class classification task and train a classifier on features defined on the surrounding context[2]. The model selects the most likely candidate from the confusion set, where the set of candidates includes the top n most frequent English prepositions. Our confusion set includes the top ten prepositions[3]: $ConfSet = \{$*on, from, for, of, about, to, at, in, with, by*$\}$. We use p to refer to a candidate preposition from $ConfSet$.

Let *preposition context* denote the preposition and the window around it. For instance, "a passion *to* what he" is a context for window size 2. We use three feature sets, varying window size from 2 to 4 words on each side (see Table 1). All feature sets consist of word n-grams of various lengths spanning p and all the features are of the form $s^{-k}ps^{+m}$, where s^{-k} and s^{+m} denote k words before and m words after p; we show two 3-gram features for illustration:

1. a passion p
2. passion p what

We implement four linear learning models: the discriminative method *Averaged Perceptron* (AP); two probabilistic methods – *a language model* (LM) and *Naïve Bayes* (NB); and a "counting" method *SumLM* (Bergsma et al., 2009).

Each model produces a score for a candidate in the confusion set. Since all of the models are linear, the hypotheses generated by the algorithms differ only in the weights they assign to the features

[2]We also report one experiment on the article correction task. We take the preposition correction task as an example; the article case is treated in the same way.

[3]This set of prepositions is also considered in other works, e.g. (Rozovskaya and Roth, 2010b). The usage of the ten most frequent prepositions accounts for 82% of all preposition errors (Leacock et al., 2010).

Feature set	Preposition context	N-gram lengths
Win2	a passion [to] what he	2,3,4
Win3	with a passion [to] what he does	2,3,4
Win4	engineer with a passion [to] what he does .	2,3,4,5

Table 1: **Description of the three feature sets** used in the experiments. All feature sets consist of word n-grams of various lengths spanning the preposition and vary by n-gram length and window size.

Method	Free Coefficient	Feature weights
AP	bias parameter	mistake-driven
LM	$\lambda \cdot prior(p)$	$\sum_{v_l \circ v_r} \lambda_{v_r} \cdot log(P(u\|v_r))$
NB	$log(prior(p))$	$log(P(f\|p))$
SumLM	$\|F(S,p)\| \cdot log(C(p))$	$log(P(f\|p))$

Table 2: **Summary of the learning methods**. $C(p)$ denotes the number of times preposition p occurred in training. λ is a smoothing parameter, u is the rightmost word in f, $v_l \circ v_r$ denotes all concatenations of substrings v_l and v_r of feature f without u.

(Roth, 1998; Roth, 1999). Thus a score computed by each of the models for a preposition p in the context S can be expressed as follows:

$$g(S,p) = C(p) + \sum_{f \in F(S,p)} w_a(f), \qquad (1)$$

where $F(S,p)$ is the set of features active in context S relative to preposition p, $w_a(f)$ is the weight algorithm a assigns to feature $f \in F$, and $C(p)$ is a free coefficient. Predictions are made using the winner-take-all approach: $argmax_p g(S,p)$. The algorithms make use of the same feature set F and differ only by how the weights $w_a(f)$ and $C(p)$ are computed. Below we explain how the weights are determined in each method. Table 2 summarizes the four approaches.

2.1 Averaged Perceptron

Discriminative classifiers represent the most common learning paradigm in error correction. AP (Freund and Schapire, 1999) is a discriminative mistake-driven online learning algorithm. It maintains a vector of feature weights w and processes one training example at a time, updating w if the current weight assignment makes a mistake on the training example. In the case of AP, the $C(p)$ coefficient refers to the bias parameter (see Table 2).

We use the regularized version of AP in Learning Based Java[4] (LBJ, (Rizzolo and Roth, 2007)). While classical Perceptron comes with a generalization bound related to the margin of the data, Averaged Perceptron also comes with a PAC-like generalization bound (Freund and Schapire, 1999). This linear learning algorithm is known, both theoretically and experimentally, to be among the best linear learning approaches and is competitive with SVM and Logistic Regression, while being more efficient in training. It also has been shown to produce state-of-the-art results on many natural language applications (Punyakanok et al., 2008).

2.2 Language Modeling

Given a feature $f = s^{-k}ps^{+m}$, let u denote the rightmost word in f and $v_l \circ v_r$ denote all concatenations of substrings v_l and v_r of feature f without u. The language model computes several probabilities of the form $P(u|v_r)$. If f ="with a passion *p* what", then u ="what", and $v_r \in$ {"with a passion *p*", "a passion *p*", "passion *p*", "*p*" }. In practice, these probabilities are smoothed and replaced with their corresponding log values, and the total weight contribution of f to the scoring function of p is $\sum_{v_l \circ v_r} \lambda_{v_r} \cdot log(P(u|v_r))$. In addition, this scoring function has a coefficient that only depends on p: $C(p) = \lambda \cdot prior(p)$ (see Table 2). The *prior* probability of a candidate p is:

$$prior(p) = \frac{C(p)}{\sum_{q \in ConfSet} C(q)}, \quad (2)$$

where $C(p)$ and $C(q)$ denote the number of times preposition p and q, respectively, occurred in the training data. We implement a count-based LM with Jelinek-Mercer linear interpolation as a smoothing method[5] (Chen and Goodman, 1996), where each n-gram length, from 1 to n, is associated with an interpolation smoothing weight λ. Weights are optimized on a held-out set of ESL sentences.

Win2 and Win3 features correspond to 4-gram LMs and Win4 to 5-gram LMs. Language models are trained with SRILM (Stolcke, 2002).

[4]LBJ can be downloaded from `http://cogcomp.cs.illinois.edu`.

[5]Unlike other LM methods, this approach allows us to train LMs on very large data sets. Although we found that backoff LMs may perform slightly better, they still maintain the same hierarchy in the order of algorithm performance.

2.3 Naïve Bayes

NB is another linear model, which is often hard to beat using more sophisticated approaches. NB architecture is also particularly well-suited for adapting the model to the first language of the writer (Section 4). Weights in NB are determined, similarly to LM, by the feature counts and the *prior* probability of each candidate p (Eq. (2)). For each candidate p, NB computes the joint probability of p and the feature space F, assuming that the features are conditionally independent given p:

$$\begin{aligned} g(S,p) &= log\{prior(p) \cdot \prod_{f \in F(S,p)} P(f|p)\} \\ &= log(prior(p)) + \\ &+ \sum_{f \in F(S,p)} log(P(f|p)) \quad (3) \end{aligned}$$

NB weights and its free coefficient are also summarized in Table 2.

2.4 SumLM

For candidate p, SumLM (Bergsma et al., 2009)[6] produces a score by summing over the logs of all feature counts:

$$\begin{aligned} g(S,p) &= \sum_{f \in F(S,p)} log(C(f)) \\ &= \sum_{f \in F(S,p)} log(P(f|p)C(p)) \\ &= |F(S,p)|C(p) + \sum_{f \in F(S,p)} log(P(f|p)) \end{aligned}$$

where $C(f)$ denotes the number of times n-gram feature f was observed with p in training. It should be clear from equation 3 that SumLM is very similar to NB, with a different free coefficient (Table 2).

3 Comparison of Algorithms

3.1 Evaluation Data

We evaluate the models using a corpus of ESL essays, annotated[7] by native English speakers (Rozovskaya and Roth, 2010a). For each preposition

[6]SumLM is one of several related methods proposed in this work; its accuracy on the preposition selection task on native English data nearly matches the best model, SuperLM (73.7% vs. 75.4%), while being much simpler to implement.

[7]The annotation of the ESL corpus can be downloaded from `http://cogcomp.cs.illinois.edu`.

Source language	Prepositions		Articles	
	Total	Incorrect	Total	Incorrect
Chinese	953	144	1864	150
Czech	627	28	575	55
Italian	687	43	-	-
Russian	1210	85	2292	213
Spanish	708	52	-	-
All	4185	352	4731	418

Table 3: **Statistics on prepositions and articles in the ESL data.** Column *Incorrect* denotes the number of cases judged to be incorrect by the annotator.

(article) used incorrectly, the annotator indicated the correct choice. The data include sentences by speakers of five first languages. Table 3 shows statistics by the source language of the writer.

3.2 Training Corpora

We use two training corpora. The first corpus, *WikiNYT*, is a selection of texts from English Wikipedia and the New York Times section of the Gigaword corpus and contains 10^7 preposition contexts. We build models of 3 sizes[8]: 10^6, $5 \cdot 10^6$, and 10^7.

To experiment with larger data sets, we use the Google Web1T 5-gram Corpus, which is a collection of n-gram counts of length one to five over a corpus of 10^{12} words. The corpus contains $2.6 \cdot 10^{10}$ prepositions. We refer to this corpus as *GoogleWeb*.

We stress that *GoogleWeb* does not contain complete sentences, but only n-gram counts. Thus, we cannot generate training data for AP for feature sets Win3 and Win4: Since the algorithm does not assume feature independence, we need to have 7 and 9-word sequences, respectively, with a preposition in the middle (as shown in Table 1) and their corpus frequencies. The other three models can be evaluated with the n-gram counts available. For example, we compute NB scores by obtaining the count of each feature independently, e.g. the count for left context 5-gram "engineer with a passion *p*" and right context 5-gram "*p* what he does .", due to the conditional independence assumption that NB makes. On *GoogleWeb*, we train NB, SumLM, and LM with three feature sets: Win2, Win3, and Win4.

From *GoogleWeb*, we also generate a smaller training set of size 10^8: We use 5-grams with a preposition in the middle and generate a new count, proportional to the size of the smaller corpus[9]. For instance, a preposition 5-gram with a count of 2600 in *GoogleWeb*, will have a count of 10 in $GoogleWeb$-10^8.

3.3 Results

Our key results of the fair comparison of the four algorithms are shown in Fig. 1 and summarized in Table 4. The table shows that AP trained on $5 \cdot 10^6$ preposition contexts performs as well as NB trained on 10^7 (i.e., with twice as much data; the performance of LM trained on 10^7 contexts is better than that of AP trained with 10 times less data (10^6), but not as good as that of AP trained with half as much data ($5{\cdot}10^6$); AP outperforms SumLM, when the latter uses 10 times more data. Fig. 1 demonstrates the performance results reported in Table 4; it shows the behavior of different systems with respect to *precision* and *recall* on the error correction task. We generate the curves by varying the decision threshold on the confidence of the classifier (Carlson et al., 2001) and propose a correction only when the confidence of the classifier is above the threshold. A higher precision and a lower recall are obtained when the decision threshold is high, and vice versa.

Key results
$AP > NB > LM > SumLM$
$AP \sim 2 \cdot NB$
$5 \cdot AP > 10 \cdot LM > AP$
$AP > 10 \cdot SumLM$

Table 4: **Key results on the comparison of algorithms**. $2 \cdot NB$ refers to NB trained with twice as much data as AP; $10 \cdot LM$ refers to LM trained with 10 times more data as AP; $10{\cdot}SumLM$ refers to $SumLM$ trained with 10 times more data as AP. These results are also shown in Fig. 1.

We now show a fair comparison of the four algorithms for different window sizes, training data and training sizes. Figure 2 compares the models trained on $WikiNYT$-10^7 corpus for Win4. AP is the superior model, followed by NB, then LM, and finally SumLM.

Results for other training sizes and feature[10] set

[8] *Training size* refers to the number of preposition contexts.

[9] Scaling down $GoogleWeb$ introduces some bias but we believe that it should not have an effect on our experiments.

[10] We have also experimented with additional POS-based features that are commonly used in these tasks and observed similar behavior.

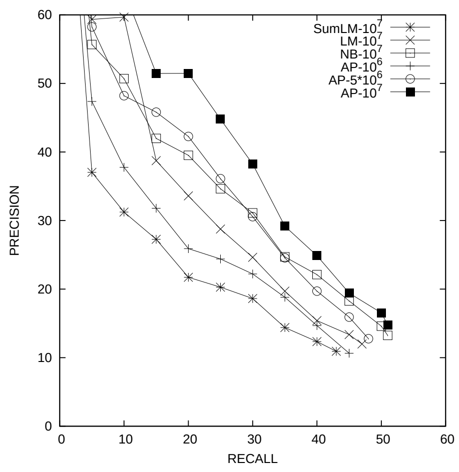

Figure 1: **Algorithm comparison across different training sizes.** (*WikiNYT, Win3*). AP (10^6 preposition contexts) performs as well as SumLM with 10 times more data, and LM requires at least twice as much data to achieve the performance of AP.

configurations show similar behavior and are reported in Table 5, which provides model comparison in terms of *Average Area Under Curve* (AAUC, (Hanley and McNeil, 1983)). AAUC is a measure commonly used to generate a summary statistic and is computed here as an average precision value over 12 recall points (from 5 to 60):

$$AAUC = \frac{1}{12} \cdot \sum_{i=1}^{12} Precision(i \cdot 5)$$

The Table also shows results on the *article correction task*[11].

Training data	**Feature set**	**Performance** ($AAUC$)			
		AP	NB	LM	SumLM
$WikiNYT$-$5 \cdot 10^6$	$Win3$	**26**	22	20	13
$WikiNYT$-10^7	$Win4$	**33**	28	24	16
$GoogleWeb$-10^8	$Win2$	**30**	29	28	15
$GoogleWeb$	$Win4$	-	**44**	41	32
Article $WikiNYT$-$5 \cdot 10^6$	$Win3$	**40**	39	-	30

Table 5: **Performance Comparison** of the four algorithms for different training data, training sizes, and window sizes. Each row shows results for training data of the same size. The last row shows performance on the *article correction* task. All other results are for prepositions.

[11]We do not evaluate the LM approach on the article correction task, since with LM it is difficult to handle missing article errors, one of the most common error types for articles, but the expectation is that it will behave as it does for prepositions.

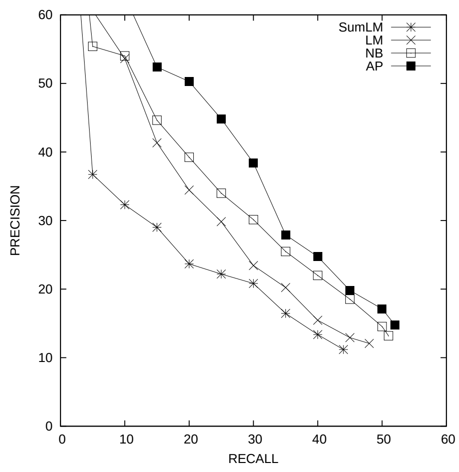

Figure 2: **Model Comparison for training data of the same size**: Performance of models for feature set *Win4* trained on $WikiNYT$-10^7.

3.3.1 Effects of Window Size

We found that expanding window size from 2 to 3 is helpful for all of the models, but expanding window to 4 is only helpful for the models trained on *GoogleWeb* (Table 6). Compared to *Win3*, *Win4* has five additional 5-gram features. We look at the proportion of features in the ESL data that occurred in two corpora: $WikiNYT$-10^7 and *GoogleWeb* (Table 7). We observe that only 4% of test 5-grams occur in $WikiNYT$-10^7. This number goes up 7 times to 28% for *GoogleWeb*, which explains why increasing the window size is helpful for this model. By comparison, a set of native English sentences (different from the training data) has 50% more 4-grams and about 3 times more 5-grams, because ESL sentences often contain expressions not common for native speakers.

Training data	**Performance** ($AAUC$)		
	Win2	Win3	Win4
$GoogleWeb$	35	39	44

Table 6: **Effect of Window Size** in terms of $AAUC$. Performance improves, as the window increases.

4 Adapting to Writer's Source Language

In this section, we discuss adapting error correction systems to the first language of the writer. Non-native speakers make mistakes in a systematic manner, and errors often depend on the first language of the writer (Lee and Seneff, 2008; Rozovskaya and

Test	**Train**	**N-gram length**			
		2	3	4	5
ESL	$WikiNYT$-10^7	98%	66%	22%	4%
Native	$WikiNYT$-10^7	98%	67%	32%	13%
ESL	*GoogleWeb*	99%	92%	64%	28%
Native-*B09*	*GoogleWeb*	-	99%	93%	70%

Table 7: **Feature coverage for ESL and native data**. Percentage of test n-gram features that occurred in training. *Native* refers to data from Wikipedia and NYT. *B09* refers to statistics from Bergsma et al. (2009).

Roth, 2010a). For instance, a Chinese learner of English might say "congratulations *to* this achievement" instead of "congratulations *on* this achievement", while a Russian speaker might say "congratulations *with* this achievement".

A system performs much better when it makes use of knowledge about typical errors. When trained on annotated ESL data instead of native data, systems improve both precision and recall (Han et al., 2010; Gamon, 2010). Annotated data include both the writer's preposition and the intended (correct) one, and thus the knowledge about typical errors is made available to the system.

Another way to adapt a model to the first language is to generate in native training data artificial errors mimicking the typical errors of the non-native writers (Rozovskaya and Roth, 2010c; Rozovskaya and Roth, 2010b). Henceforth, we refer to this method, proposed within the discriminative framework AP, as *AP-adapted*. To determine typical mistakes, error statistics are collected on a small set of annotated ESL sentences. However, for the model to use these language-specific error statistics, a separate classifier for each source language needs to be trained.

We propose a novel adaptation method, which shows performance improvement over *AP-adapted*. Moreover, this method is much simpler to implement, since there is no need to train per source language; only one classifier is trained. The method relies on the observation that error regularities can be viewed as a *distribution on priors over the correction candidates*. Given a preposition *s* in text, the *prior* for candidate p is the probability that p is the correct preposition for *s*. If a model is trained on native data without adaptation to the source language, candidate priors correspond to the relative frequencies of the candidates in the native training data. More importantly, these priors remain the same regardless of the source language of the writer or of the preposition used in text. From the model's perspective, it means that a correction candidate, for example *to*, is equally likely given that the author's preposition is *for* or *from*, which is clearly incorrect and disagrees with the notion that errors are regular and language-dependent.

We use the annotated ESL data and define *adapted* candidate priors that are dependent on the author's preposition and the author's source language. Let *s* be a preposition appearing in text by a writer of source language L_1, and p a correction candidate. Then the *adapted* prior of p given *s* is:

$$prior(p, s, L_1) = \frac{C_{L_1}(s, p)}{C_{L_1}(s)},$$

where $C_{L_1}(s)$ denotes the number of times *s* appeared in the ESL data by L_1 writers, and $C_{L_1}(s, p)$ denotes the number of times p was the correct preposition when *s* was used by an L_1 writer.

Table 8 shows adapted candidate priors for two author's choices – when an ESL writer used *on* and *at* – based on the data from Chinese learners. One key distinction of the adapted priors is the high probability assigned to the author's preposition: the new prior for *on* given that it is also the preposition found in text is 0.70, vs. the 0.07 prior based on the native data. The adapted prior of preposition p, when p is used, is always high, because the majority of prepositions are used correctly. Higher probabilities are also assigned to those candidates that are most often observed as corrections for the author's preposition. For example, the adapted prior for *at* when the writer chose *on* is 0.10, since *on* is frequently incorrectly chosen instead of *at*.

To determine a mechanism to inject the adapted priors into a model, we note that while all of our models use priors in some way, NB architecture directly specifies the prior probability as one of its parameters (Sec. 2.3). We thus train NB in a traditional way, on native data, and then replace the prior component in Eq. (3) with the adapted prior, language and preposition dependent, to get the score for p of the *NB-adapted* model:

$$g(S, p) = log\{prior(p, s, L_1) \cdot \prod_{f \in F(S,p)} P(f|p)\}$$

Candidate	Global prior	Adapted prior			
		author's choice	prior	author's choice	prior
of	0.25	on	0.03	at	0.02
to	0.22	on	0.06	at	0.00
in	0.15	on	0.04	at	0.16
for	0.10	on	0.00	at	0.03
on	0.07	on	**0.70**	at	0.09
by	0.06	on	0.00	at	0.02
with	0.06	on	0.04	at	0.00
at	0.04	on	0.10	at	**0.75**
from	0.04	on	0.00	at	0.02
about	0.01	on	0.03	at	0.00

Table 8: **Examples of *adapted* candidate priors for two author's choices – *on* and *at* – based on the errors made by Chinese learners**. *Global prior* denotes the probability of the candidate in the standard model and is based on the relative frequency of the candidate in native training data. *Adapted priors* are dependent on the author's preposition and the author's first language. Adapted priors for the author's choice are very high. Other candidates are given higher priors if they often appear as corrections for the author's choice.

We stress that in the new method there is no need to train per source language, as with previous adaption methods. Only one model is trained, and only at decision time, we change the prior probabilities of the model. Also, while we need a lot of data to train the model, only one parameter depends on annotated data. Therefore, with rather small amounts of data, it is possible to get reasonably good estimates of these prior parameters.

In the experiments below, we compare four models: *AP*, *NB* *AP-adapted* and *NB-adapted*. *AP-adapted* is the adaptation through artificial errors and *NB-adapted* is the method proposed here. Both of the adapted models use the same error statistics in k-fold cross-validation (CV): We randomly partition the ESL data into k parts, with each part tested on the model that uses error statistics estimated on the remaining $k-1$ parts. We also remove all preposition errors that occurred only once (23% of all errors) to allow for a better evaluation of the adapted models. Although we observe similar behavior on all the data, the models especially benefit from the adapted priors when a particular error occurred more than once. Since the majority of errors are not due to chance, we focus on those errors that the writers will make repeatedly.

Fig. 3 shows the four models trained on $WikiNYT$-10^7. First, we note that the adapted

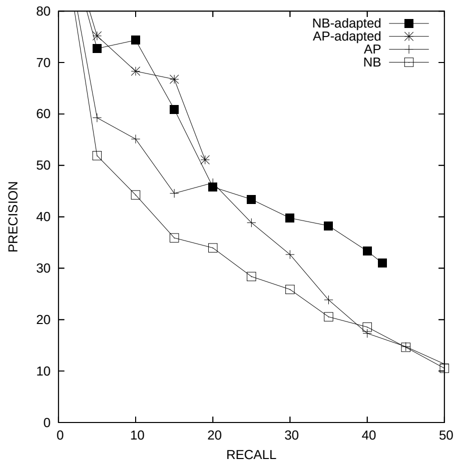

Figure 3: **Adapting to Writer's Source Language**. *NB-adapted* is the method proposed here. *AP-adapted* and *NB-adapted* results are obtained using 2-fold CV, with 50% of the ESL data used for estimating the new priors. All models are trained on $WikiNYT$-10^7.

models outperform their non-adapted counterparts with respect to precision. Second, for the recall points less than 20%, the adapted models obtain very similar precision values. This is interesting, especially because *NB* does not perform as well as *AP*, as we also showed in Sec. 3.3. Thus, *NB-adapted* not only improves over *NB*, but its gap compared to the latter is much wider than the gap between the AP-based systems. Finally, an important performance distinction between the two adapted models is the loss in recall exhibited by *AP-adapted* – its curve is shorter because *AP-adapted* is very conservative and does not propose many corrections. In contrast, *NB-adapted* succeeds in improving its precision over *NB* with almost no recall loss.

To evaluate the effect of the size of the data used to estimate the new priors, we compare the performance of *NB-adapted* models in three settings: *2-fold CV*, *10-fold CV*, and *Leave-One-Out* (Figure 4). In *2-fold CV*, priors are estimated on 50% of the ESL data, in *10-fold* on 90%, and in *Leave-One-Out* on all data but the testing example. Figure 4 shows the averaged results over 5 runs of CV for each setting. The model converges very quickly: there is almost no difference between *10-fold CV* and *Leave-One-Out*, which suggests that we can get a good estimate of the priors using just a little annotated data.

Table 9 compares *NB* and *NB-adapted* for two corpora: $WikiNYT$-10^7 and $GoogleWeb$. Since

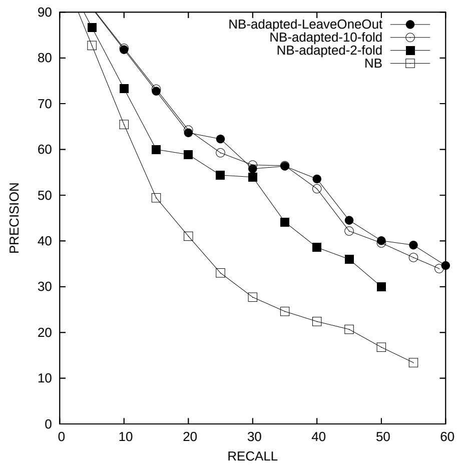

Figure 4: **How much data are needed to estimate adapted priors**. Comparison of *NB-adapted* models trained on *GoogleWeb* that use different amounts of data to estimate the new priors. In *2-fold CV*, priors are estimated on 50% of the data; in *10-fold* on 90% of the data; in *Leave-One-Out*, the new priors are based on all the data but the testing example.

$GoogleWeb$ is several orders of magnitude larger, the adapted model behaves better for this corpus.

So far, we have discussed performance in terms of precision and recall, but we can also discuss it in terms of accuracy, to see how well the algorithm is performing compared to the *baseline* on the task. Following Rozovskaya and Roth (2010c), we consider as the baseline the accuracy of the ESL data before applying the model[12], or *the percentage of prepositions used correctly* in the test data. From Table 3, the baseline is 93.44%[13]. Compared to this high baseline, NB trained on $WikiNYT$-10^7 achieves an accuracy of 93.54, and *NB-adapted* achieves an accuracy of 93.93[14].

Training data	**Algorithms**	
	NB	NB-adapted
$WikiNYT$-10^7	29	**53**
$GoogleWeb$	38	**62**

Table 9: **Adapting to writer's source language.** Results are reported in terms of $AAUC$. *NB-adapted* is the model with adapted priors. Results for *NB-adapted* are based on 10-fold CV.

[12]Note that this baseline is different from the *majority* baseline used in the preposition *selection* task, since here we have the author's preposition in text.

[13]This is the baseline after removing the singleton errors.

[14]We select the best accuracy among different values that can be achieved by varying the decision threshold.

5 Conclusion

We have addressed two important issues in ESL error correction, which are essential to making progress in this task. First, we presented an extensive, fair comparison of four popular linear learning models for the task and demonstrated that there are significant performance differences between the approaches. Since all of the algorithms presented here are linear, the only difference is in how they learn the weights. Our experiments demonstrated that the discriminative approach (AP) is able to generalize better than any of the other models. These results correct earlier conclusions, made with incomparable data sets. The model comparison was performed using two popular tasks – correcting errors in article and preposition usage – and we expect that our results will generalize to other ESL correction tasks.

The second, and most important, contribution of the paper is a novel method that allows one to adapt the learned model to the source language of the writer. We showed that error patterns can be viewed as a distribution on priors over the correction candidates and proposed a method of injecting the adapted priors into the learned model. In addition to performing much better than the previous approaches, this method is also very cheap to implement, since it does not require training a separate model for each source language, but adapts the system to the writer's language at decision time.

Acknowledgments

The authors thank Nick Rizzolo for many helpful discussions. The authors also thank Josh Gioja, Nick Rizzolo, Mark Sammons, Joel Tetreault, Yuancheng Tu, and the anonymous reviewers for their insightful comments. This research is partly supported by a grant from the U.S. Department of Education.

References

S. Bergsma, D. Lin, and R. Goebel. 2009. Web-scale n-gram models for lexical disambiguation. In *21st International Joint Conference on Artificial Intelligence*, pages 1507–1512.

J. Bitchener, S. Young, and D. Cameron. 2005. The effect of different types of corrective feedback on ESL student writing. *Journal of Second Language Writing*.

A. Carlson, J. Rosen, and D. Roth. 2001. Scaling up context sensitive text correction. In *Proceedings of the*

National Conference on Innovative Applications of Artificial Intelligence (IAAI), pages 45–50.

Y. S. Chan and H. T. Ng. 2005. Word sense disambiguation with distribution estimation. In *Proceedings of IJCAI 2005*.

S. Chen and J. Goodman. 1996. An empirical study of smoothing techniques for language modeling. In *Proceedings of ACL 1996*.

M. Chodorow, J. Tetreault, and N.-R. Han. 2007. Detection of grammatical errors involving prepositions. In *Proceedings of the Fourth ACL-SIGSEM Workshop on Prepositions*, pages 25–30, Prague, Czech Republic, June. Association for Computational Linguistics.

G. Dalgish. 1985. Computer-assisted ESL research. *CALICO Journal*, 2(2).

J. Eeg-Olofsson and O. Knuttson. 2003. Automatic grammar checking for second language learners - the use of prepositions. *Nodalida*.

A. Elghaari, D. Meurers, and H. Wunsch. 2010. Exploring the data-driven prediction of prepositions in english. In *Proceedings of COLING 2010*, Beijing, China.

R. De Felice and S. Pulman. 2008. A classifier-based approach to preposition and determiner error correction in L2 English. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 169–176, Manchester, UK, August.

Y. Freund and R. E. Schapire. 1999. Large margin classification using the perceptron algorithm. *Machine Learning*, 37(3):277–296.

M. Gamon, J. Gao, C. Brockett, A. Klementiev, W. Dolan, D. Belenko, and L. Vanderwende. 2008. Using contextual speller techniques and language modeling for ESL error correction. In *Proceedings of IJCNLP*.

M. Gamon. 2010. Using mostly native data to correct errors in learners' writing. In *NAACL*, pages 163–171, Los Angeles, California, June.

A. R. Golding and D. Roth. 1999. A Winnow based approach to context-sensitive spelling correction. *Machine Learning*, 34(1-3):107–130.

N. Han, M. Chodorow, and C. Leacock. 2006. Detecting errors in English article usage by non-native speakers. *Journal of Natural Language Engineering*, 12(2):115–129.

N. Han, J. Tetreault, S. Lee, and J. Ha. 2010. Using an error-annotated learner corpus to develop and ESL/EFL error correction system. In *LREC*, Malta, May.

J. Hanley and B. McNeil. 1983. A method of comparing the areas under receiver operating characteristic curves derived from the same cases. *Radiology*, 148(3):839–843.

E. Izumi, K. Uchimoto, T. Saiga, T. Supnithi, and H. Isahara. 2003. Automatic error detection in the Japanese learners' English spoken data. In *The Companion Volume to the Proceedings of 41st Annual Meeting of the Association for Computational Linguistics*, pages 145–148, Sapporo, Japan, July.

C. Leacock, M. Chodorow, M. Gamon, and J. Tetreault. 2010. Morgan and Claypool Publishers.

J. Lee and S. Seneff. 2008. An analysis of grammatical errors in non-native speech in English. In *Proceedings of the 2008 Spoken Language Technology Workshop*.

V. Punyakanok, D. Roth, and W. Yih. 2008. The importance of syntactic parsing and inference in semantic role labeling. *Computational Linguistics*, 34(2).

N. Rizzolo and D. Roth. 2007. Modeling Discriminative Global Inference. In *Proceedings of the First International Conference on Semantic Computing (ICSC)*, pages 597–604, Irvine, California, September. IEEE.

D. Roth. 1998. Learning to resolve natural language ambiguities: A unified approach. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pages 806–813.

D. Roth. 1999. Learning in natural language. In *Proc. of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 898–904.

A. Rozovskaya and D. Roth. 2010a. Annotating ESL errors: Challenges and rewards. In *Proceedings of the NAACL Workshop on Innovative Use of NLP for Building Educational Applications*.

A. Rozovskaya and D. Roth. 2010b. Generating confusion sets for context-sensitive error correction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

A. Rozovskaya and D. Roth. 2010c. Training paradigms for correcting errors in grammar and usage. In *Proceedings of the NAACL-HLT*.

A. Stolcke. 2002. Srilm-an extensible language modeling toolkit. In *Proceedings International Conference on Spoken Language Processing*, pages 257–286, November.

J. Tetreault and M. Chodorow. 2008. The ups and downs of preposition error detection in ESL writing. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 865–872, Manchester, UK, August.

J. Tetreault, J. Foster, and M. Chodorow. 2010. Using parse features for preposition selection and error detection. In *ACL*.