Grammatical Error Correction with Alternating Structure Optimization

Daniel Dahlmeier[1] and **Hwee Tou Ng**[1,2]
[1]NUS Graduate School for Integrative Sciences and Engineering
[2]Department of Computer Science, National University of Singapore
{danielhe,nght}@comp.nus.edu.sg

Abstract

We present a novel approach to grammatical error correction based on Alternating Structure Optimization. As part of our work, we introduce the *NUS Corpus of Learner English (NUCLE)*, a fully annotated one million words corpus of learner English available for research purposes. We conduct an extensive evaluation for article and preposition errors using various feature sets. Our experiments show that our approach outperforms two baselines trained on non-learner text and learner text, respectively. Our approach also outperforms two commercial grammar checking software packages.

1 Introduction

Grammatical error correction (GEC) has been recognized as an interesting as well as commercially attractive problem in natural language processing (NLP), in particular for learners of English as a foreign or second language (EFL/ESL).

Despite the growing interest, research has been hindered by the lack of a large annotated corpus of learner text that is available for research purposes. As a result, the standard approach to GEC has been to train an off-the-shelf classifier to re-predict words in non-learner text. Learning GEC models directly from annotated learner corpora is not well explored, as are methods that combine learner and non-learner text. Furthermore, the evaluation of GEC has been problematic. Previous work has either evaluated on artificial test instances as a substitute for real learner errors or on proprietary data that is not available to other researchers. As a consequence, existing methods have not been compared on the same test set, leaving it unclear where the current state of the art really is.

In this work, we aim to overcome both problems. First, we present a novel approach to GEC based on Alternating Structure Optimization (ASO) (Ando and Zhang, 2005). Our approach is able to train models on annotated learner corpora while still taking advantage of large non-learner corpora. Second, we introduce the *NUS Corpus of Learner English (NUCLE)*, a fully annotated one million words corpus of learner English available for research purposes. We conduct an extensive evaluation for article and preposition errors using six different feature sets proposed in previous work. We compare our proposed ASO method with two baselines trained on non-learner text and learner text, respectively. To the best of our knowledge, this is the first extensive comparison of different feature sets on real learner text which is another contribution of our work. Our experiments show that our proposed ASO algorithm significantly improves over both baselines. It also outperforms two commercial grammar checking software packages in a manual evaluation.

The remainder of this paper is organized as follows. The next section reviews related work. Section 3 describes the tasks. Section 4 formulates GEC as a classification problem. Section 5 extends this to the ASO algorithm. The experiments are presented in Section 6 and the results in Section 7. Section 8 contains a more detailed analysis of the results. Section 9 concludes the paper.

Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics, pages 915–923,
Portland, Oregon, June 19-24, 2011.

2 Related Work

In this section, we give a brief overview on related work on article and preposition errors. For a more comprehensive survey, see (Leacock et al., 2010).

The seminal work on grammatical error correction was done by Knight and Chander (1994) on article errors. Subsequent work has focused on designing better features and testing different classifiers, including memory-based learning (Minnen et al., 2000), decision tree learning (Nagata et al., 2006; Gamon et al., 2008), and logistic regression (Lee, 2004; Han et al., 2006; De Felice, 2008). Work on preposition errors has used a similar classification approach and mainly differs in terms of the features employed (Chodorow et al., 2007; Gamon et al., 2008; Lee and Knutsson, 2008; Tetreault and Chodorow, 2008; Tetreault et al., 2010; De Felice, 2008). All of the above works only use non-learner text for training.

Recent work has shown that training on annotated learner text can give better performance (Han et al., 2010) and that the observed word used by the writer is an important feature (Rozovskaya and Roth, 2010b). However, training data has either been small (Izumi et al., 2003), only partly annotated (Han et al., 2010), or artificially created (Rozovskaya and Roth, 2010b; Rozovskaya and Roth, 2010a).

Almost no work has investigated ways to combine learner and non-learner text for training. The only exception is Gamon (2010), who combined features from the output of logistic-regression classifiers and language models trained on non-learner text in a meta-classifier trained on learner text. In this work, we show a more direct way to combine learner and non-learner text in a single model.

Finally, researchers have investigated GEC in connection with web-based models in NLP (Lapata and Keller, 2005; Bergsma et al., 2009; Yi et al., 2008). These methods do not use classifiers, but rely on simple n-gram counts or page hits from the Web.

3 Task Description

In this work, we focus on article and preposition errors, as they are among the most frequent types of errors made by EFL learners.

3.1 Selection vs. Correction Task

There is an important difference between training on annotated learner text and training on non-learner text, namely whether the observed word can be used as a feature or not. When training on non-learner text, the observed word cannot be used as a feature. The word choice of the writer is "blanked out" from the text and serves as the correct class. A classifier is trained to re-predict the word given the surrounding context. The *confusion set* of possible classes is usually pre-defined. This *selection task* formulation is convenient as training examples can be created "for free" from any text that is assumed to be free of grammatical errors. We define the more realistic *correction task* as follows: given a particular word and its context, propose an appropriate correction. The proposed correction can be identical to the observed word, i.e., no correction is necessary. The main difference is that the word choice of the writer can be encoded as part of the features.

3.2 Article Errors

For article errors, the classes are the three articles *a*, *the*, and the *zero-article*. This covers article insertion, deletion, and substitution errors. During training, each noun phrase (NP) in the training data is one training example. When training on learner text, the correct class is the article provided by the human annotator. When training on non-learner text, the correct class is the observed article. The context is encoded via a set of feature functions. During testing, each NP in the test set is one test example. The correct class is the article provided by the human annotator when testing on learner text or the observed article when testing on non-learner text.

3.3 Preposition Errors

The approach to preposition errors is similar to articles but typically focuses on preposition substitution errors. In our work, the classes are 36 frequent English prepositions (*about, along, among, around, as, at, beside, besides, between, by, down, during, except, for, from, in, inside, into, of, off, on, onto, outside, over, through, to, toward, towards, under, underneath, until, up, upon, with, within, without*), which we adopt from previous work. Every prepositional phrase (PP) that is governed by one of the

36 prepositions is one training or test example. We ignore PPs governed by other prepositions.

4 Linear Classifiers for Grammatical Error Correction

In this section, we formulate GEC as a classification problem and describe the feature sets for each task.

4.1 Linear Classifiers

We use classifiers to approximate the unknown relation between articles or prepositions and their contexts in learner text, and their valid corrections. The articles or prepositions and their contexts are represented as feature vectors $\mathbf{X} \in \mathcal{X}$. The corrections are the classes $Y \in \mathcal{Y}$.

In this work, we employ binary linear classifiers of the form $\mathbf{u}^T\mathbf{X}$ where $\mathbf{u}$ is a weight vector. The outcome is considered $+1$ if the score is positive and -1 otherwise. A popular method for finding $\mathbf{u}$ is *empirical risk minimization* with *least square regularization*. Given a training set $\{\mathbf{X}_i, Y_i\}_{i=1,\ldots,n}$, we aim to find the weight vector that minimizes the empirical loss on the training data

$$\hat{\mathbf{u}} = \arg\min_{\mathbf{u}} \left(\frac{1}{n} \sum_{i=1}^{n} L(\mathbf{u}^T\mathbf{X}_i, Y_i) + \lambda \, ||\mathbf{u}||^2 \right), \quad (1)$$

where L is a loss function. We use a modification of Huber's robust loss function. We fix the regularization parameter λ to 10^{-4}. A multi-class classification problem with m classes can be cast as m binary classification problems in a *one-vs-rest* arrangement. The prediction of the classifier is the class with the highest score $\hat{Y} = \arg\max_{Y \in \mathcal{Y}} (\mathbf{u}_Y^T\mathbf{X})$. In earlier experiments, this linear classifier gave comparable or superior performance compared to a logistic regression classifier.

4.2 Features

We re-implement six feature extraction methods from previous work, three for articles and three for prepositions. The methods require different linguistic pre-processing: chunking, CCG parsing, and constituency parsing.

4.2.1 Article Errors

- **DeFelice** The system in (De Felice, 2008) for article errors uses a CCG parser to extract a rich set of syntactic and semantic features, including part of speech (POS) tags, hypernyms from WordNet (Fellbaum, 1998), and named entities.
- **Han** The system in (Han et al., 2006) relies on shallow syntactic and lexical features derived from a chunker, including the words before, in, and after the NP, the head word, and POS tags.
- **Lee** The system in (Lee, 2004) uses a constituency parser. The features include POS tags, surrounding words, the head word, and hypernyms from WordNet.

4.2.2 Preposition Errors

- **DeFelice** The system in (De Felice, 2008) for preposition errors uses a similar rich set of syntactic and semantic features as the system for article errors. In our re-implementation, we do not use a subcategorization dictionary, as this resource was not available to us.
- **TetreaultChunk** The system in (Tetreault and Chodorow, 2008) uses a chunker to extract features from a two-word window around the preposition, including lexical and POS n-grams, and the head words from neighboring constituents.
- **TetreaultParse** The system in (Tetreault et al., 2010) extends (Tetreault and Chodorow, 2008) by adding additional features derived from a constituency and a dependency parse tree.

For each of the above feature sets, we add the observed article or preposition as an additional feature when training on learner text.

5 Alternating Structure Optimization

This section describes the ASO algorithm and shows how it can be used for grammatical error correction.

5.1 The ASO algorithm

Alternating Structure Optimization (Ando and Zhang, 2005) is a multi-task learning algorithm that takes advantage of the *common structure* of multiple related problems. Let us assume that we have m binary classification problems. Each classifier $\mathbf{u}_i$ is a

weight vector of dimension p. Let Θ be an orthonormal $h \times p$ matrix that captures the common structure of the m weight vectors. We assume that each weight vector can be decomposed into two parts: one part that models the particular i-th classification problem and one part that models the common structure

$$\mathbf{u}_i = \mathbf{w}_i + \Theta^T \mathbf{v}_i. \quad (2)$$

The parameters $[\{\mathbf{w}_i, \mathbf{v}_i\}, \Theta]$ can be learned by *joint empirical risk minimization*, i.e., by minimizing the joint empirical loss of the m problems on the training data

$$\sum_{l=1}^{m} \left(\frac{1}{n} \sum_{i=1}^{n} L\left(\left(\mathbf{w}_l + \Theta^T \mathbf{v}_l\right)^T \mathbf{X}_i^l, Y_i^l \right) + \lambda \left|\left|\mathbf{w}_l\right|\right|^2 \right). \quad (3)$$

The key observation in ASO is that the problems used to find Θ do not have to be same as the *target problems* that we ultimately want to solve. Instead, we can automatically create *auxiliary problems* for the sole purpose of learning a better Θ.

Let us assume that we have k target problems and m auxiliary problems. We can obtain an approximate solution to Equation 3 by performing the following algorithm (Ando and Zhang, 2005):

1. Learn m linear classifiers $\mathbf{u}_i$ independently.
2. Let $U = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_m]$ be the $p \times m$ matrix formed from the m weight vectors.
3. Perform Singular Value Decomposition (SVD) on U: $U = V_1 D V_2^T$. The first h column vectors of V_1 are stored as rows of Θ.
4. Learn $\mathbf{w}_j$ and $\mathbf{v}_j$ for each of the target problems by minimizing the empirical risk:
$$\frac{1}{n} \sum_{i=1}^{n} L\left(\left(\mathbf{w}_j + \Theta^T \mathbf{v}_j\right)^T \mathbf{X}_i, Y_i \right) + \lambda \left|\left|\mathbf{w}_j\right|\right|^2.$$
5. The weight vector for the j-th target problem is:
$$\mathbf{u}_j = \mathbf{w}_j + \Theta^T \mathbf{v}_j.$$

5.2 ASO for Grammatical Error Correction

The key observation in our work is that the selection task on non-learner text is a highly informative *auxiliary problem* for the correction task on learner text. For example, a classifier that can predict the presence or absence of the preposition *on* can be helpful for correcting wrong uses of *on* in learner text, e.g., if the classifier's confidence for *on* is low but the writer used the preposition *on*, the writer might have made a mistake. As the auxiliary problems can be created automatically, we can leverage the power of very large corpora of non-learner text.

Let us assume a grammatical error correction task with m classes. For each class, we define a binary auxiliary problem. The feature space of the auxiliary problems is a restriction of the original feature space $\mathcal{X}$ to all features except the observed word: $\mathcal{X} \backslash \{X_{obs}\}$. The weight vectors of the auxiliary problems form the matrix U in Step 2 of the ASO algorithm from which we obtain Θ through SVD. Given Θ, we learn the vectors $\mathbf{w}_j$ and $\mathbf{v}_j$, $j = 1, \ldots, k$ from the annotated learner text using the complete feature space $\mathcal{X}$.

This can be seen as an instance of *transfer learning* (Pan and Yang, 2010), as the auxiliary problems are trained on data from a different domain (non-learner text) and have a slightly different feature space ($\mathcal{X} \backslash \{X_{obs}\}$). We note that our method is general and can be applied to any classification problem in GEC.

6 Experiments

6.1 Data Sets

The main corpus in our experiments is the *NUS Corpus of Learner English (NUCLE)*. The corpus consists of about 1,400 essays written by EFL/ESL university students on a wide range of topics, like environmental pollution or healthcare. It contains over one million words which are completely annotated with error tags and corrections. All annotations have been performed by professional English instructors. We use about 80% of the essays for training, 10% for development, and 10% for testing. We ensure that no sentences from the same essay appear in both the training and the test or development data. NUCLE is available to the community for research purposes.

On average, only 1.8% of the articles and 1.3% of the prepositions in NUCLE contain an error. This figure is considerably lower compared to other learner corpora (Leacock et al., 2010, Ch. 3) and shows that our writers have a relatively high proficiency of English. We argue that this makes the task considerably more difficult. Furthermore, to keep the task as realistic as possible, we do not filter the

test data in any way.

In addition to NUCLE, we use a subset of the New York Times section of the Gigaword corpus[1] and the Wall Street Journal section of the Penn Treebank (Marcus et al., 1993) for some experiments. We pre-process all corpora using the following tools: We use NLTK[2] for sentence splitting, OpenNLP[3] for POS tagging, YamCha (Kudo and Matsumoto, 2003) for chunking, the C&C tools (Clark and Curran, 2007) for CCG parsing and named entity recognition, and the Stanford parser (Klein and Manning, 2003a; Klein and Manning, 2003b) for constituency and dependency parsing.

6.2 Evaluation Metrics

For experiments on non-learner text, we report accuracy, which is defined as the number of correct predictions divided by the total number of test instances. For experiments on learner text, we report F_1-measure

$$F_1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

where precision is the number of suggested corrections that agree with the human annotator divided by the total number of proposed corrections by the system, and recall is the number of suggested corrections that agree with the human annotator divided by the total number of errors annotated by the human annotator.

6.3 Selection Task Experiments on WSJ Test Data

The first set of experiments investigates predicting articles and prepositions in non-learner text. This primarily serves as a reference point for the correction task described in the next section. We train classifiers as described in Section 4 on the Gigaword corpus. We train with up to 10 million training instances, which corresponds to about 37 million words of text for articles and 112 million words of text for prepositions. The test instances are extracted from section 23 of the WSJ and no text from the WSJ is included in the training data. The observed article or preposition choice of the writer is the class we want to predict. Therefore, the article or preposition cannot be part of the input features. Our proposed ASO method is not included in these experiments, as it uses the observed article or preposition as a feature which is only applicable when testing on learner text.

6.4 Correction Task Experiments on NUCLE Test Data

The second set of experiments investigates the primary goal of this work: to automatically correct grammatical errors in learner text. The test instances are extracted from NUCLE. In contrast to the previous selection task, the observed word choice of the writer can be different from the correct class and the observed word is available during testing. We investigate two different baselines and our ASO method.

The first baseline is a classifier trained on the Gigaword corpus in the same way as described in the selection task experiment. We use a simple thresholding strategy to make use of the observed word during testing. The system only flags an error if the difference between the classifier's confidence for its first choice and the confidence for the observed word is higher than a threshold t. The threshold parameter t is tuned on the NUCLE development data for each feature set. In our experiments, the value for t is between 0.7 and 1.2.

The second baseline is a classifier trained on NUCLE. The classifier is trained in the same way as the Gigaword model, except that the observed word choice of the writer is included as a feature. The correct class during training is the correction provided by the human annotator. As the observed word is part of the features, this model does not need an extra thresholding step. Indeed, we found that thresholding is harmful in this case. During training, the instances that do not contain an error greatly outnumber the instances that do contain an error. To reduce this imbalance, we keep all instances that contain an error and retain a random sample of q percent of the instances that do not contain an error. The undersample parameter q is tuned on the NUCLE development data for each data set. In our experiments, the value for q is between 20% and 40%.

Our ASO method is trained in the following way. We create binary auxiliary problems for articles or prepositions, i.e., there are 3 auxiliary problems for

[1] LDC2009T13

[2] www.nltk.org

[3] opennlp.sourceforge.net

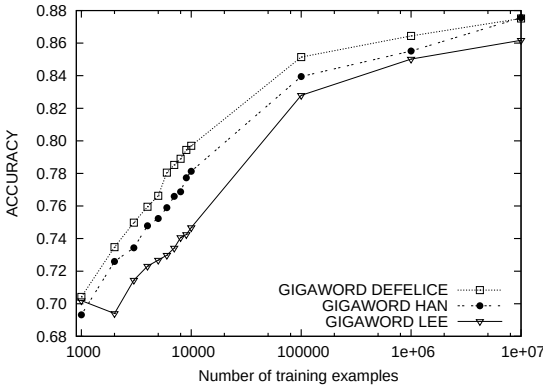
articles and 36 auxiliary problems for prepositions. We train the classifiers for the auxiliary problems on the complete 10 million instances from Gigaword in the same ways as in the selection task experiment. The weight vectors of the auxiliary problems form the matrix U. We perform SVD to get $U = V_1 D V_2^T$. We keep all columns of V_1 to form Θ. The target problems are again binary classification problems for each article or preposition, but this time trained on NUCLE. The observed word choice of the writer is included as a feature for the target problems. We again undersample the instances that do not contain an error and tune the parameter q on the NUCLE development data. The value for q is between 20% and 40%. No thresholding is applied.

We also experimented with a classifier that is trained on the concatenated data from NUCLE and Gigaword. This model always performed worse than the better of the individual baselines. The reason is that the two data sets have different feature spaces which prevents simple concatenation of the training data. We therefore omit these results from the paper.

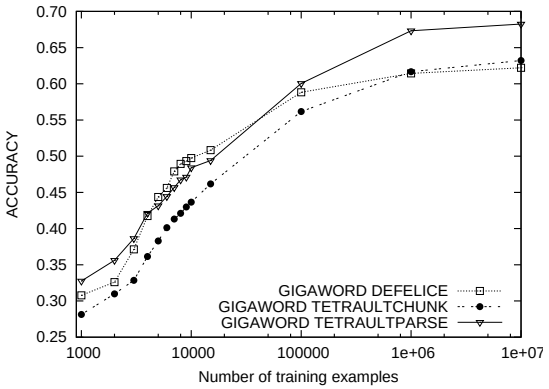
7 Results

The learning curves of the selection task experiments on WSJ test data are shown in Figure 1. The three curves in each plot correspond to different feature sets. Accuracy improves quickly in the beginning but improvements get smaller as the size of the training data increases. The best results are 87.56% for articles (Han) and 68.25% for prepositions (TetreaultParse). The best accuracy for articles is comparable to the best reported results of 87.70% (Lee, 2004) on this data set.

The learning curves of the correction task experiments on NUCLE test data are shown in Figure 2 and 3. Each sub-plot shows the curves of three models as described in the last section: ASO trained on NUCLE and Gigaword, the baseline classifier trained on NUCLE, and the baseline classifier trained on Gigaword. For ASO, the x-axis shows the number of target problem training instances. The first observation is that high accuracy for the selection task on non-learner text does not automatically entail high F_1-measure on learner text. We also note that feature sets with similar performance on non-learner text can show very different performance on



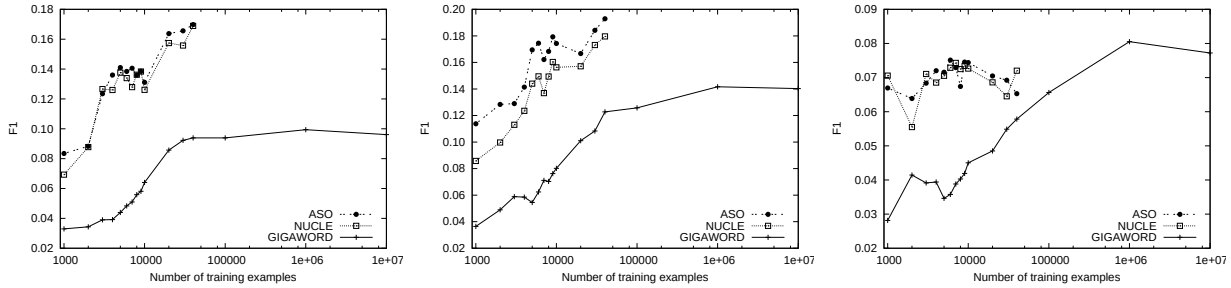
(a) Articles



(b) Prepositions

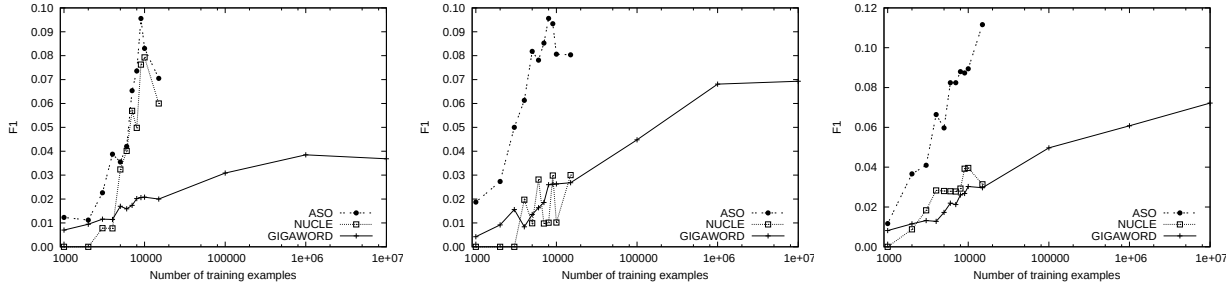
Figure 1: Accuracy for the selection task on WSJ test data.

learner text. The second observation is that training on annotated learner text can significantly improve performance. In three experiments (articles DeFelice, Han, prepositions DeFelice), the NUCLE model outperforms the Gigaword model trained on 10 million instances. Finally, the ASO models show the best results. In the experiments where the NUCLE models already perform better than the Gigaword baseline, ASO gives comparable or slightly better results (articles DeFelice, Han, Lee, prepositions DeFelice). In those experiments where neither baseline shows good performance (TetreaultChunk, TetreaultParse), ASO results in a large improvement over either baseline. The best results are 19.29% F_1-measure for articles (Han) and 11.15% F_1-measure for prepositions (TetreaultParse) achieved by the ASO model.



(a) DeFelice (b) Han (c) Lee

Figure 2: F_1-measure for the article correction task on NUCLE test data. Each plot shows ASO and two baselines for a particular feature set.



(a) DeFelice (b) TetreaultChunk (c) TetreaultParse

Figure 3: F_1-measure for the preposition correction task on NUCLE test data. Each plot shows ASO and two baselines for a particular feature set.

8 Analysis

In this section, we analyze the results in more detail and show examples from our test set for illustration.

Table 1 shows precision, recall, and F_1-measure for the best models in our experiments. ASO achieves a higher F_1-measure than either baseline. We use the sign-test with bootstrap re-sampling for statistical significance testing. The sign-test is a non-parametric test that makes fewer assumptions than parametric tests like the t-test. The improvements in F_1-measure of ASO over either baseline are statistically significant ($p < 0.001$) for both articles and prepositions.

The difficulty in GEC is that in many cases, more than one word choice can be correct. Even with a threshold, the Gigaword baseline model suggests too many corrections, because the model cannot make use of the observed word as a feature. This results in low precision. For example, the model replaces **as** with **by** in the sentence *"This group should be categorized **as the vulnerable group**"*, which is wrong.

Articles			
Model	**Prec**	**Rec**	**F_1**
Gigaword (Han)	10.33	**21.81**	14.02
NUCLE (Han)	**29.48**	12.91	17.96
ASO (Han)	26.44	15.18	**19.29**
Prepositions			
Model	**Prec**	**Rec**	**F_1**
Gigaword (TetreaultParse)	4.77	**14.81**	7.21
NUCLE (DeFelice)	13.84	5.55	7.92
ASO (TetreaultParse)	**18.30**	8.02	**11.15**

Table 1: Best results for the correction task on NUCLE test data. Improvements for ASO over either baseline are statistically significant ($p < 0.001$) for both tasks.

In contrast, the NUCLE model learns a bias towards the observed word and therefore achieves higher precision. However, the training data is

smaller and therefore recall is low as the model has not seen enough examples during training. This is especially true for prepositions which can occur in a large variety of contexts. For example, the preposition **in** should be **on** in the sentence *"... psychology had an impact **in** the way we process and manage technology"*. The phrase *"impact on the way"* does not appear in the NUCLE training data and the NUCLE baseline fails to detect the error.

The ASO model is able to take advantage of both the annotated learner text and the large non-learner text, thus achieving overall high F_1-measure. The phrase *"impact on the way"*, for example, appears many times in the Gigaword training data. With the common structure learned from the auxiliary problems, the ASO model successfully finds and corrects this mistake.

8.1 Manual Evaluation

We carried out a manual evaluation of the best ASO models and compared their output with two commercial grammar checking software packages which we call *System A* and *System B*. We randomly sampled 1000 test instances for articles and 2000 test instances for prepositions and manually categorized each test instance into one of the following categories: *(1) Correct* means that both human and system flag an error and suggest the same correction. If the system's correction differs from the human but is equally acceptable, it is considered *(2) Both Ok*. If the system identifies an error but fails to correct it, we consider it *(3) Both Wrong*, as both the writer and the system are wrong. *(4) Other Error* means that the system's correction does not result in a grammatical sentence because of another grammatical error that is outside the scope of article or preposition errors, e.g., a noun number error as in *"all **the** dog"*. If the system corrupts a previously correct sentence it is a *(5) False Flag*. If the human flags an error but the system does not, it is a *(6) Miss*. *(7) No Flag* means that neither the human annotator nor the system flags an error. We calculate precision by dividing the count of category (1) by the sum of counts of categories (1), (3), and (5), and recall by dividing the count of category (1) by the sum of counts of categories (1), (3), and (6). The results are shown in Table 2. Our ASO method outperforms both commercial software packages. Our evaluation shows that even commercial software packages achieve low F_1-measure for article and preposition errors, which confirms the difficulty of these tasks.

Articles			
	ASO	**System A**	**System B**
(1) Correct	4	1	1
(2) Both Ok	16	12	18
(3) Both Wrong	0	1	0
(4) Other Error	1	0	0
(5) False Flag	1	0	4
(6) Miss	3	5	6
(7) No Flag	975	981	971
Precision	**80.00**	50.00	20.00
Recall	**57.14**	14.28	14.28
F_1	**66.67**	22.21	16.67
Prepositions			
	ASO	**System A**	**System B**
(1) Correct	3	3	0
(2) Both Ok	35	39	24
(3) Both Wrong	0	2	0
(4) Other Error	0	0	0
(5) False Flag	5	11	1
(6) Miss	12	11	15
(7) No Flag	1945	1934	1960
Precision	**37.50**	18.75	0.00
Recall	**20.00**	18.75	0.00
F_1	**26.09**	18.75	0.00

Table 2: Manual evaluation and comparison with commercial grammar checking software.

9 Conclusion

We have presented a novel approach to grammatical error correction based on Alternating Structure Optimization. We have introduced the NUS Corpus of Learner English (NUCLE), a fully annotated corpus of learner text. Our experiments for article and preposition errors show the advantage of our ASO approach over two baseline methods. Our ASO approach also outperforms two commercial grammar checking software packages in a manual evaluation.

Acknowledgments

This research was done for CSIDM Project No. CSIDM-200804 partially funded by a grant from the National Research Foundation (NRF) administered by the Media Development Authority (MDA) of Singapore.

References

R.K. Ando and T. Zhang. 2005. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6.

S. Bergsma, D. Lin, and R. Goebel. 2009. Web-scale n-gram models for lexical disambiguation. In *Proceedings of IJCAI*.

M. Chodorow, J. Tetreault, and N.R. Han. 2007. Detection of grammatical errors involving prepositions. In *Proceedings of the 4th ACL-SIGSEM Workshop on Prepositions*.

S. Clark and J.R. Curran. 2007. Wide-coverage efficient statistical parsing with CCG and log-linear models. *Computational Linguistics*, 33(4).

R. De Felice. 2008. *Automatic Error Detection in Non-native English*. Ph.D. thesis, University of Oxford.

C. Fellbaum, editor. 1998. *WordNet: An electronic lexical database*. MIT Press, Cambridge,MA.

M. Gamon, J. Gao, C. Brockett, A. Klementiev, W.B. 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: A meta-classifier approach. In *Proceedings of HLT-NAACL*.

N.R. Han, M. Chodorow, and C. Leacock. 2006. Detecting errors in English article usage by non-native speakers. *Natural Language Engineering*, 12(02).

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

E. Izumi, K. Uchimoto, T. Saiga, T. Supnithi, and H. Isahara. 2003. Automatic error detection in the Japanese learners' English spoken data. In *Companion Volume to the Proceedings of ACL*.

D. Klein and C.D. Manning. 2003a. Accurate unlexicalized parsing. In *Proceedings of ACL*.

D. Klein and C.D. Manning. 2003b. Fast exact inference with a factored model for natural language processing. *Advances in Neural Information Processing Systems (NIPS 2002)*, 15.

K. Knight and I. Chander. 1994. Automated postediting of documents. In *Proceedings of AAAI*.

T Kudo and Y. Matsumoto. 2003. Fast methods for kernel-based text analysis. In *Proceedings of ACL*.

M. Lapata and F. Keller. 2005. Web-based models for natural language processing. *ACM Transactions on Speech and Language Processing*, 2(1).

C. Leacock, M. Chodorow, M. Gamon, and J. Tetreault. 2010. *Automated Grammatical Error Detection for Language Learners*. Morgan & Claypool Publishers, San Rafael,CA.

J. Lee and O. Knutsson. 2008. The role of PP attachment in preposition generation. In *Proceedings of CICLing*.

J. Lee. 2004. Automatic article restoration. In *Proceedings of HLT-NAACL*.

M.P. Marcus, M.A. Marcinkiewicz, and B. Santorini. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19.

G. Minnen, F. Bond, and A. Copestake. 2000. Memory-based learning for article generation. In *Proceedings of CoNLL*.

R. Nagata, A. Kawai, K. Morihiro, and N. Isu. 2006. A feedback-augmented method for detecting errors in the writing of learners of English. In *Proceedings of COLING-ACL*.

S.J. Pan and Q. Yang. 2010. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10).

A. Rozovskaya and D. Roth. 2010a. Generating confusion sets for context-sensitive error correction. In *Proceedings of EMNLP*.

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

J. Tetreault and M. Chodorow. 2008. The ups and downs of preposition error detection in ESL writing. In *Proceedings of COLING*.

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

X. Yi, J. Gao, and W.B. Dolan. 2008. A web-based English proofing system for English as a second language users. In *Proceedings of IJCNLP*.