KU Leuven at HOO-2012: A Hybrid Approach to Detection and Correction of Determiner and Preposition Errors in Non-native English Text

Li Quan, Oleksandr Kolomiyets, Marie-Francine Moens
Department of Computer Science
KU Leuven
Celestijnenlaan 200A
3001 Heverlee, Belgium
li.quan@student.kuleuven.be
{oleksandr.kolomiyets, sien.moens}@cs.kuleuven.be

Abstract

In this paper we describe the technical implementation of our system that participated in the Helping Our Own 2012 Shared Task (HOO-2012). The system employs a number of preprocessing steps and machine learning classifiers for correction of determiner and preposition errors in non-native English texts. We use maximum entropy classifiers trained on the provided HOO-2012 development data and a large high-quality English text collection. The system proposes a number of highly-probable corrections, which are evaluated by a language model and compared with the original text. A number of deterministic rules are used to increase the precision and recall of the system. Our system is ranked among the three best performing HOO-2012 systems with a precision of 31.15%, recall of 22.08% and F_1-score of 25.84% for correction of determiner and preposition errors combined.

1 Introduction

The Helping Our Own Challenge (Dale and Kilgarriff, 2010) is a shared task that was proposed to address automated error correction of non-native English texts. In particular, the Helping Our Own 2012 Shared Task (HOO-2012) (Dale et al., 2012) focuses on determiners and prepositions as they are well-known sources for errors produced by non-native English writers. For instance, Bitchener et al. (2005) reported error rates of respectively 20% and 29%.

Determiners are in particular challenging because they depend on a large discourse context and world knowledge, and moreover, they simply do not exist in many languages, such as Slavic and South-East Asian languages (Ghomeshi et al., 2009). The use of prepositions in English is idiomatic and thus very difficult for learners of English. On the one hand, prepositions connect noun phrases to other words in a sentence (e.g. *... by bus*), on the other hand, they can also be part of phrasal verbs such as *carry on*, *hold on*, etc.

In this paper we describe our system implementation and results in HOO-2012. The paper is structured as follows. Section 2 gives the task definition, errors addressed, data resources and evaluation criteria and metrics. Section 3 shows some background and related work. Section 4 gives the full system description, while Section 5 reports and discusses the results of the experiments. Section 6 concludes with an error analysis and possible further improvements.

2 HOO-2012 Tasks and Resources

2.1 Tasks

In the scope of HOO-2012 the following six possible error types[1] are targeted:

- Replace determiner (RD):
  *Have **the** nice day. → Have **a** nice day.*
- Missing determiner (MD):
  *That is great idea. → That is **a** great idea.*
- Unnecessary determiner (UD):
  *I like **the** pop music. → I like pop music.*

[1]The set of error tags is based on the Cambridge University Press Error Coding System, fully described in (Nicholls, 2003).

The 7th Workshop on the Innovative Use of NLP for Building Educational Applications, pages 263–271,
Montréal, Canada, June 3-8, 2012.

- Replace preposition (RT):
  ***In** the other hand... → **On** the other hand...*
- Missing preposition (MT):
  *She woke up 6 o'clock. → She woke up **at** 6 o'clock.*
- Unnecessary preposition (UT):
  *He must go **to** home. → He must go home.*

2.2 Data

The HOO development dataset consists of 1000 exam scripts drawn from a subset of the CLC FCE Dataset (Yannakoudakis et al., 2011). This corpus contains texts written by students who attended the Cambridge ESOL First Certificate in English examination in 2000 and 2001. The entire development dataset comprises 374680 words, with an average of 375 words per file. The test data consists of a further 100 files provided by Cambridge University Press (CUP), with 18013 words, and an average of 180 words per file.

Type	# Dev	# Test A	# Test B
RD	609	38	37
MD	2230	125	131
UD	1048	53	62
Det	3887	217	230
RT	2618	136	148
MT	1104	57	56
UT	822	43	39
Prep	4545	236	243
Total	8432	453	473
Words/Error	44.18	39.77	38.08

Table 1: Data error statistics.

Counts of the different error types are provided in Table 1. The table shows counts for the development dataset ('Dev') and two versions of the gold standard test data: the original version as derived from the CUP-provided dataset ('Test A'), and a revised version ('Test B') which was compiled in response to requests for corrections from participating teams. The datasets and the revision process are further explained in (Dale et al., 2012).

2.3 Evaluation Criteria and Metrics

For evaluation in the HOO framework, a distinction is made between scores and measures. The complete evaluation mechanism is described in detail in (Dale and Narroway, 2012) and on the HOO-2012 website.[2]

Scores Three different scores are used:

1. Detection: does the system determine that an edit of the specified type is required at some point in the text?
2. Recognition: does the system correctly determine the extent of the source text that requires editing?
3. Correction: does the system offer a correction that is identical to that provided in the gold standard?

Measures For each score, three measures are calculated: precision (1), recall (2) and F-score (3).

$$precision = \frac{tp}{tp + fp} \quad (1)$$

$$recall = \frac{tp}{tp + fn} \quad (2)$$

where tp is the number of true positives (the number of instances that are correctly found by the system), fp the number of false positives (the number of instances that are incorrectly found), and fn the number of false negatives (missing results).

$$F_\beta = (1 + \beta^2)\frac{precision \cdot recall}{\beta^2 \cdot precision + recall} \quad (3)$$

where β is used as a weight factor regulating the trade-off between recall and precision. We use the balanced F-score, i.e. $\beta = 1$, such that recall and precision are equally weighted.

Combined We provide results on prepositions and determiners combined, and for each of these two subcategories separately. We also report on each of the different error types separately.

[2]See `http://www.correcttext.org/hoo2012`.

3 Related Work

HOO-2012 follows on from the HOO-2011 Shared Task Pilot Round (Dale and Kilgarriff, 2011). That task targeted a broader range of error types, and used a much smaller dataset.

Most work on models for determiner and preposition generation has been developed in the context of machine translation output (e.g. (Knight and Chander, 1994), (Minnen et al., 2000), (De Felice and Pulman, 2007) and (Toutanova and Suzuki, 2007)). Some of these methods depend on full parsing of text, which is not reliable in the context of noisy non-native English texts.

Only more recently, models for automated error detection and correction of non-native texts have been explicitly developed and studied. Most of these methods use large corpora of well-formed native English text to train statistical models, e.g. (Han et al., 2004), (Gamon et al., 2008) and (De Felice and Pulman, 2008). Yi et al. (2008) used web counts to determine correct article usage, while Han et al. (2010) trained a classifier solely on a large error-tagged learner corpus for preposition error correction.

4 System Description

4.1 Global System Workflow

The system utilizes a hybrid approach that combines statistical machine learning classifiers and a rule-based system. The global system architecture is presented in Figure 1. This section describes the global system workflow. The subsequent sections elaborate on the machine learning classifiers and heuristics implemented in the system.

The system workflow is divided in the following processing steps:

1. Text Preprocessing: The system performs a *preliminary text analysis* by automated spelling correction and subsequent syntactic analysis, such as tokenization and part-of-speech (POS) tagging.

2. Error Detection, Recognition and Correction: The system identifies if a correction is needed, and the type and extent of that correction. Two families of error correction tasks that separately address determiners and prepositions are performed in parallel.

3. Correction validation: Once a correction has been proposed, it is *validated* by a language model derived from a large corpus of high-quality English text.

4.1.1 Text Preprocessing

In HOO-2012, texts submitted for automated corrections are written by learners of English. Besides the error types that are addressed in HOO-2012, misspellings are another type of highly-frequent errors. For example, one student writes the following: *In my point of vue, Internet is the most important discover of the 2000 centery.*

When using automated natural language processing tools, incorrect spelling (and grammar) can introduce an additional bias. To reduce the bias propagated from the preprocessing steps, the text is first automatically corrected by the open-source spell checker GNU Aspell.[3]

At the next step, the text undergoes a shallow syntactic analysis that includes sentence boundary detection, tokenization, part-of-speech tagging, chunking, lemmatization, relation finding and prepositional phrase attachment. These tasks are performed by MBSP (De Smedt et al., 2010).[4]

4.1.2 Error Detection, Recognition and Correction

In general, the task of automated error correction is addressed by a number of subtasks of finding the position in text, recognizing the type of error, and the proposal for a correction. In our implementation we approach these tasks in a two-step approach as proposed in (Gamon et al., 2008). With two families of errors, the system therefore employs four classifiers in total.

For determiner error corrections, a classifier (C1 in Figure 1) first predicts whether a determiner is required in the observed context. If it is required, another classifier (C2 in Figure 1) estimates which one. The same approach is employed for the preposition error correction task (classifiers C3 and C4 in Figure 1). The details on how the classifiers were implemented are highlighted in Section 4.2.

[3] `http://aspell.net/`

[4] MBSP is a text analysis system based on the TiMBL and MBT memory based learning applications developed at CLiPS and ILK (Daelemans and van den Bosch, 2005).

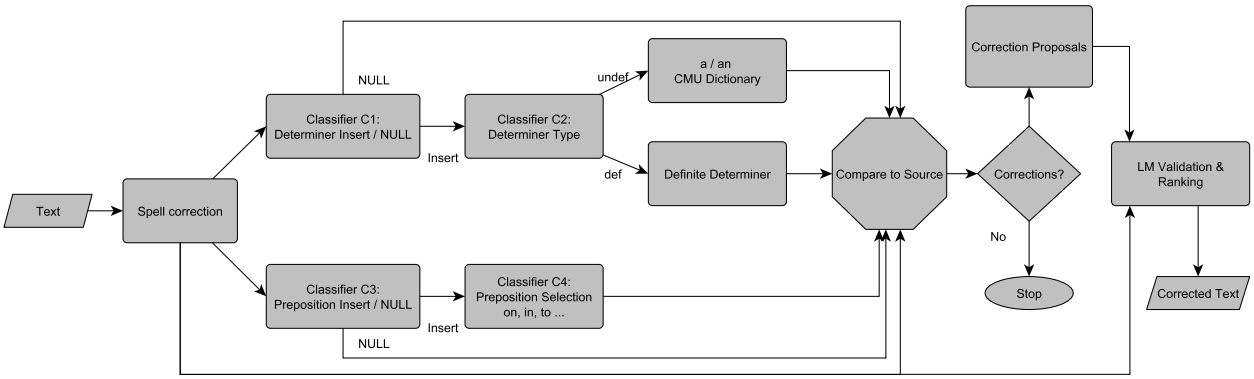

Figure 1: System architecture.

4.1.3 Correction Validation

Our error correction system implements a correction validation mechanism as proposed in (Gamon et al., 2008). The validation mechanism makes use of a language model that is derived from a large corpus of English. We use a trigram language model trained on the English Gigaword corpus with a 64K-word vocabulary (using interpolated Kneser-Ney smoothing with a bigram cutoff of 3 and trigram cutoff of 5).

The language model serves to increase the precision at the cost of recall as false positives can be confusing for learners for English. The original sentence and the error-corrected version are passed to the language model. Only if the difference in probability of being generated by the language model exceeds a heuristic threshold (estimated using a tuning set) is the correction finally accepted.

4.2 Machine Learning Classifiers

As already mentioned, the system employs four machine learning classifiers in total (C1–C4 — two for each family of errors). Classifiers C1 and C3 respectively estimate the presence of determiners and prepositions in the observed context. If one is expected, the second set of classifiers estimates which one is the most likely.

For the determiner choice classifier (C2), we restrict the determiner choice class values to the *indefinite* and *definite* articles: *a/an* and *the*. The preposition choice class values for the preposition choice classifier (C4) are restricted to set of the following 10 common prepositions: *on*, *in*, *at*, *for*, *of*, *about*, *from*, *to*, *by*, *with* and (*other*).

All the classifiers are implemented by discriminative maximum entropy classification models (ME) (Ratnaparkhi, 1998). Such models have been proven effective for a number of natural language processing tasks by combining heterogeneous forms of evidence (Ratnaparkhi, 2010).

Training Classifiers and Inference As training instances we consider each noun phrase (NP) in every sentence of the training data. For the binary classifiers (C1 and C3), a positive example is a noun phrase that follows a determiner/preposition, and a negative example is one that does not. The multiclass classifiers (C2 and C4) are trained respectively to distinguish specific instances of determiners (definite and indefinite for C2) and the set of prepositions mentioned above. For each classifier, a training instance is represented by the following features:

- Tokens in NP.
- Tokens' POS tags in NP.
- Tokens' lemmas in NP.
- Tokens in a contextual window of 3 tokens to the left and to the right from the potential correction position.
- Tokens' POS tags in a contextual window of 3 tokens from the potential correction position.
- Tokens' lemmas in a contextual window of 3 tokens from the potential correction position.
- Trigrams of concatenated tokens before and after NP.

- Trigrams of concatenated tokens' POS tags before and after NP.
- Trigrams of concatenated tokens' lemmas before and after NP.
- Head noun in NP.
- POS tag of head noun in NP.
- Lemma of head noun in NP.

Once the classification models have been derived, the classifiers are ready to be employed in the system. For the text correction task, each sentence undergoes the same preprocessing analysis as described in Section 4.1.1. Then, for each noun phrase in the input sentence, we extract the feature context, and use the models to predict the need for the presence of a determiner or preposition, and if so, which one. Our system only accepts classifier predictions if they are obtained with a high confidence. The confidence thresholds were empirically estimated from pre-evaluation experiments with a tuning dataset (Section 5.1).

4.3 Rule-based Modules

Our system also has a number of rule-based modules. The first rule-based module is in charge of making the choice between *a* and *an* if the determiner type classifier (C2) predicts the presence of an indefinite determiner. The choice is determined by a lookup in the CMU pronouncing dictionary[5] (*a/an* CMU Dictionary in Figure 1). In this dictionary each word entry is mapped to one or a number of pronunciations in the phonetic transcription code system Arpabet. If the pronunciation of the word that follows the estimated correction position starts with a consonant, *a* is used; if it starts with a vowel, *an* is selected.

The second rule-based module corrects confusion errors of determiner-noun agreement, e.g. *this/these* and *that/those* (Definite Determiner in Figure 1). It is implemented by introducing rules with patterns based on whether the noun was tagged as singular or plural.

The third rule-based module is used to filter out unnecessary corrections proposed by the classifiers (C1-C4) and augmented by the already described rule-based modules. Each correction is examined against the input text and if it yields a different text than the original input text, such a correction is considered as a necessary correction.

However, sometimes automatically proposed corrections have to be rejected because they are out of scope of the addressed errors. We do not replace possessive determiners such as *my*, *your*, *his*, *our*, *their* by the definite article *the*. Similarly, some prepositions can be grouped in opposite pairs, for example *from* and *to*, for which we do not propose any correction as it requires a deep semantic analysis of text.

5 Experiments and Results

In this section we describe the pre-evaluation experiments and the results of the final evaluation on the HOO-2012 test set. Table 2 shows the characteristics of the datasets used in the experiments.

Dataset	Sentences	Tokens
HOO training	21925	340693
HOO tuning	2560	40966
HOO held-out	2749	42325
Reuters	207083	5487021
Wikipedia	53370	1430428
HOO test	1376	20606

Table 2: Datasets used.

5.1 Pre-Evaluation Experiments

In the course of system development, we split the files in the HOO development dataset into a training set (80%), a tuning set (10%) and a held-out test set (10%). From the beginning it was clear that the provided development dataset alone was too small to address the automated error correction tasks by employing machine learning classification techniques. Additionally to that dataset, we used a set of Reuters news data and the Wikipedia corpus for training the classifiers.

Once the classification models had been derived, the system was evaluated on the tuning data and adjusted in order to increase the overall performance.

[5] http://www.speech.cs.cmu.edu/cgi-bin/cmudict

After that, the system was evaluated on the held-out test set for which the results are shown in Table 3.

Type	Precision	Recall	F_1-score
Det	64.11	14.89	24.17
Prep	52.32	16.38	25.32
All	60.19	15.38	24.50

Table 3: Correction results on held-out test set.

5.2 Final System Configuration and Evaluation Results

For the final evaluation, we retrained the models using the complete HOO development data (again, in addition to the Reuters and Wikipedia corpus mentioned above). The number of training instances are shown in Table 4.

Classifier	# Training instances
C1	1746128
C2	530885
C3	1763784
C4	706775

Table 4: Number of training instances used for the ME models.

In the HOO framework, precision and recall are weighted equally. However, in the domain of error correction for non-native writers, precision is probably more important because false positives can be very confusing and demotivating for learners of English. For this reason, we submitted two different runs which also gave us insights into the impact of the language model. 'Run 0' denotes the system excluding the language model and using lower thresholds, such that neither precision nor recall is favored in particular, while 'Run 1' focuses on precision by using the language model as a filter, and having higher thresholds. Thus, we present the results for two different runs on the final HOO test set, both before and after manual revision (see Section 2.2). Table 5 presents the results for recognition and Table 6 those for correction.

The difficulty of the HOO 2012 Shared Task is reflected by rather low system performance levels (Dale et al., 2012). Nonetheless, we observed some interesting patterns. In terms of the overall system performance, our system achieved better results for determiner errors than for preposition errors.

With respect to determiners, missing determiners are handled best by our system, while unnecessary determiners and replacement errors are more difficult. Concerning prepositions, missing prepositions are found to be the most challenging. This confirms the difficulty of choosing the right preposition due to the large number of possible alternatives, and their sometimes subtle differences in usage and meaning.

While 'Run 1' achieved a higher precision (at the cost of recall), 'Run 0' performed better in terms of overall performance (F_1-score). This result can be explained by the relative small size and limited tuning of the language model. Moreover, it also shows that the use of the F_1-score might not be the most informative evaluation metric in this context.

6 Conclusions

Determiners and prepositions present real challenges for non-native English writers. For automated determiner and preposition error correction in HOO-2012, we implemented a hybrid system that combines statistical machine learning classifiers and a rule-based system. By employing a language model for correction validation, the system achieved a precision of 42.16%, recall of 9.49% and F_1-score of 15.50%. Without the language model, a precision of 31.15%, recall of 22.08% and F_1-score of 25.84% were reached, and our system was ranked third in terms of F_1-score.

Three major bottlenecks were identified in the implementation: (i) spelling errors should first be corrected due to the noisy input texts; (ii) classifier thresholds must be carefully adjusted to minimize false positives; and (iii) overall, preposition errors are handled worse than determiner errors, although there is also a large difference among the various error types.

For future work, we will focus on models that explicitly utilize the writer's background. Also, a full evaluation of the system should include a thorough user-centric study with evaluation criteria and metrics beyond the traditional precision, recall and F-score.

Type	Precision	Recall	F_1-score
RD	17.95	17.95	17.95
MD	60.76	38.40	47.06
UD	22.67	32.08	26.56
Det	37.31	33.18	35.12
RT	55.88	13.97	22.35
MT	50.00	5.26	9.52
UT	14.77	30.23	19.85
Prep	27.34	14.83	19.23
All	33.33	23.62	27.65

(a) Run 0 (before revision)

Type	Precision	Recall	F_1-score
RD	19.44	17.95	18.67
MD	65.82	39.69	49.52
UD	26.67	32.26	29.20
Det	40.93	34.50	37.44
RT	61.76	14.09	22.95
MT	50.00	5.36	9.68
UT	15.91	35.90	22.05
Prep	29.69	15.57	20.43
All	29.47	24.74	29.47

(b) Run 0 (after revision).

Type	Precision	Recall	F_1-score
RD	37.50	7.69	12.77
MD	66.67	12.80	21.48
UD	16.67	1.89	3.39
Det	52.63	9.22	15.69
RT	51.61	11.76	19.16
MT	40.00	3.51	6.45
UT	32.14	20.93	25.35
Prep	42.19	11.44	18.00
All	46.08	10.38	16.94

(c) Run 1 (before revision).

Type	Precision	Recall	F_1-score
RD	37.50	8.33	13.64
MD	79.17	14.50	24.52
UD	33.33	3.23	5.88
Det	63.16	10.48	17.98
RT	54.84	11.41	18.89
MT	40.00	3.57	6.56
UT	35.71	25.64	29.85
Prep	45.31	11.89	18.83
All	51.96	11.21	18.43

(d) Run 1 (after revision).

Table 5: Recognition results of the runs on the test set.

Type	Precision	Recall	F_1-score
RD	17.95	17.95	17.95
MD	54.43	34.40	42.16
UD	22.67	32.08	26.56
Det	34.72	30.88	32.68
RT	50.00	12.50	20.00
MT	50.00	5.26	9.52
UT	14.77	30.23	19.85
Prep	25.78	13.98	18.13
All	31.15	22.08	25.84

(a) Run 0 (before revision)

Type	Precision	Recall	F_1-score
RD	17.95	19.44	18.67
MD	59.49	35.88	44.76
UD	26.67	32.26	29.20
Det	38.34	32.31	35.07
RT	55.88	12.75	20.77
MT	50.00	5.36	9.68
UT	15.91	35.90	22.05
Prep	28.13	14.81	19.41
All	34.27	23.26	27.71

(b) Run 0 (after revision).

Type	Precision	Recall	F_1-score
RD	37.50	7.69	12.77
MD	62.50	12.00	20.13
UD	16.67	1.89	3.39
Det	50.00	8.76	14.90
RT	41.94	9.56	15.57
MT	40.00	3.51	6.45
UT	32.14	20.93	25.35
Prep	37.50	10.17	16.00
All	42.16	9.49	15.50

(c) Run 1 (before revision).

Type	Precision	Recall	F_1-score
RD	37.50	8.33	13.64
MD	75.00	13.74	23.23
UD	33.33	3.23	5.88
Det	60.05	10.04	17.23
RT	45.16	9.40	15.56
MT	40.00	3.57	6.56
UT	35.71	25.64	29.85
Prep	40.63	10.66	16.88
All	48.04	10.36	17.04

(d) Run 1 (after revision).

Table 6: Correction results of the runs on the test set.

References

John Bitchener, Stuart Young, and Denise Cameron. 2005. The effect of different types of corrective feedback on ESL student writing. *Journal of Second Language Writing*, 14:191–205.

Walter Daelemans and Antal van den Bosch. 2005. *Memory-Based Language Processing*. Studies in Natural Language Processing. Cambridge University Press.

Robert Dale and Adam Kilgarriff. 2010. Helping Our Own: Text massaging for computational linguistics as a new shared task. In *Proceedings of the 6th International Natural Language Generation Conference*, pages 261–266, Dublin, Ireland, 7–9 July 2010.

Robert Dale and Adam Kilgarriff. 2011. Helping Our Own: The HOO 2011 pilot shared task. In *Proceedings of the 13th European Workshop on Natural Language Generation*, pages 242–249, Nancy, France, 28–30 September 2011.

Robert Dale and George Narroway. 2012. A framework for evaluating text correction. In *Proceedings of the Eight International Conference on Language Resources and Evaluation*, Istanbul, Turkey, 21–27 May 2012.

Robert Dale, Ilya Anisimoff, and George Narroway. 2012. HOO 2012: A report on the preposition and determiner error correction shared task. In *Proceedings of the Seventh Workshop on Innovative Use of NLP for Building Educational Applications*, Montreal, Canada, 3–8 June 2012.

Rachele De Felice and Stephen G. Pulman. 2007. Automatically acquiring models of preposition use. In *Proceedings of the Fourth ACL-SIGSEM Workshop on Prepositions*, pages 45–50, Prague, Czech Republic, 28 June 2007.

Rachele De Felice and Stephen G. 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*, pages 169–176, Manchester, United Kingdom, 18–22 August 2008.

Tom De Smedt, Vincent Van Asch, and Walter Daelemans. 2010. Memory-based shallow parser for Python. *CLiPS Technical Report Series (CTRS)*, 2.

Michael Gamon, Lucy Vanderwende, Jianfeng Gao, Chris Brockett, Alexandre Klementiev, William B. Dolan, and Dmitriy Belenko. 2008. Using contextual speller techniques and language modeling for ESL error correction. In *Proceedings of the International Joint Conference on Natural Language Processing*, pages 449–456, Hyderabad, India, 7–12 January 2008.

Jila Ghomeshi, Paul Ileana, and Martina Wiltschko. 2009. *Determiners: Universals and Variation*. Linguistik Aktuell/Linguistics Today. John Benjamins Publishing Company.

Na-Rae Han, Martin Chodorow, and Claudia Leacock. 2004. Detecting errors in English article usage with a maximum entropy classifier trained on a large, diverse corpus. In *Proceedings of the 4th International Conference on Language Resources and Evaluation*, Lisbon, Portugal, 26–28 May 2004.

Na-Rae Han, Joel Tetreault, Soo-Hwa Lee, and Jin-Young Ha. 2010. Using an error-annotated learner corpus to develop an ESL/EFL error correction system. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation*, Valletta, Malta, 19–21 May 2010.

Kevin Knight and Ishwar Chander. 1994. Automatic postediting of documents. In *Proceedings of the 12th National Conference on Artificial Intelligence*, pages 779–784, Seattle, Washington, USA, 31 July–4 August 1994.

Guido Minnen, Francis Bond, and Ann Copestake. 2000. Memory-based learning for article generation. In *Proceedings of the 4th Conference on Computational Natural Language Learning and the Second Learning Language in Logic Workshop*, pages 43–48, Lisbon, Portugal, 13–14 September 2000.

Diane Nicholls. 2003. The Cambridge Learner Corpus—error coding and analysis for lexicography and ELT. In *Proceedings of the Corpus Linguistics 2003 Conference*, pages 572–581, Lancaster, UK, 29 March–2 April 2003.

Adwait Ratnaparkhi. 1998. *Maximum entropy models for natural language ambiguity resolution*. Ph.D. thesis, Philadelphia, PA, USA. AAI9840230.

Adwait Ratnaparkhi. 2010. Maximum entropy models for natural language processing. In *Encyclopedia of Machine Learning*, pages 647–651.

Kristina Toutanova and Hisami Suzuki. 2007. Generating case markers in machine translation. In *Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 49–56, Rochester, New York, USA, 22–27 April 2007.

Helen Yannakoudakis, Ted Briscoe, and Ben Medlock. 2011. A new dataset and method for automatically grading ESOL texts. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 180–189, Portland, Oregon, USA, 19–24 June 2011.

Xing Yi, Jianfeng Gao, and William B. Dolan. 2008. A web-based English proofing system for English as a second language users. In *Proceedings of the Third International Join Conference on Natural Language Processing*, Hyderabad, India, 7–12 January 2008.