Search right and thou shalt find ... Using Web Queries for Learner Error Detection

Michael Gamon
Microsoft Research
One Microsoft Way
Redmond, WA 981052, USA
mgamon@microsoft.com

Claudia Leacock
Butler Hill Group
P.O. Box 935
Ridgefield, CT 06877, USA
Claudia.leacock@gmail.com

Abstract

We investigate the use of web search queries for detecting errors in non-native writing. Distinguishing a correct sequence of words from a sequence with a learner error is a baseline task that any error detection and correction system needs to address. Using a large corpus of error-annotated learner data, we investigate whether web search result counts can be used to distinguish correct from incorrect usage. In this investigation, we compare a variety of query formulation strategies and a number of web resources, including two major search engine APIs and a large web-based n-gram corpus.

1 Introduction

Data-driven approaches to the detection and correction of non-native errors in English have been researched actively in the past several years. Such errors are particularly amenable to data-driven methods because many prominent learner writing errors involve a relatively small class of phenomena that can be targeted with specific models, in particular article and preposition errors. Preposition and determiner errors (most of which are article errors) are the second and third most frequent errors in the *Cambridge Learner Corpus* (after the more intractable problem of content word choice). By targeting the ten most frequent prepositions involved in learner errors, more than 80% of preposition errors in the corpus are covered.

Typically, data-driven approaches to learner errors use a classifier trained on contextual information such as tokens and part-of-speech tags within a window of the preposition/article (Gamon et al. 2008, 2010, DeFelice and Pulman 2007, 2008, Han et al. 2006, Chodorow et al. 2007, Tetreault and Chodorow 2008).

Language models are another source of evidence that can be used in error detection. Using language models for this purpose is not a new approach, it goes back to at least Atwell (1987). Gamon et al. (2008) and Gamon (2010) use a combination of classification and language modeling. Once language modeling comes into play, the *quantity* of the training data comes to the forefront. It has been well-established that statistical models improve as the size of the training data increases (Banko and Brill 2001a, 2001b). This is particularly true for language models: other statistical models such as a classifier, for example, can be targeted towards a specific decision/classification, reducing the appetite for data somewhat, while language models provide probabilities for any sequence of words - a task that requires immense training data resources if the language model is to consider increasingly sparse longer n-grams.

Language models trained on data sources like the Gigaword corpus have become commonplace, but of course there is one corpus that dwarfs any other resource in size: the World Wide Web. This has drawn the interest of many researchers in natural language processing over the past decade. To mention just a few examples, Zhu and Rosenfeld (2001) combine trigram counts from the web with an existing language model where the estimates of the existing model are unreliable because of data sparseness. Keller and Lapata (2003) advocate the use of the web as a corpus to retrieve backoff probabilities for unseen bigrams. Lapata and Keller (2005) extend this method to a range of additional natural language processing tasks, but also caution that web counts have limitations and add noise. Kilgariff (2007) points out the shortcomings of accessing the web as a corpus through search queries: (a) there is no lemmatization or part-of-speech tagging in search indices, so a linguistically meaningful query can only be approximated, (b) search syntax, as implemented by search engine providers, is limited, (c) there is often a limit on the number of automatic queries that are allowed by search engines, (c) hit count estimates are estimates of retrieved *pages*, not of retrieved words. We would like to add to that list that hit count estimates on the web are just that -- estimates. They are computed on the fly by proprietary algorithms, and apparently the algorithms also access different slices of the web index, which causes a fluctuation over time, as Tetrault and Chodorow (2009) point out.

In 2006, Google made its web-based 5gram language model available through the Linguistic Data Consortium, which opens the possibility of using real n-gram statistics derived from the web directly, instead of using web search as a proxy.

In this paper we explore the use of the web as a corpus for a very specific task: distinguishing between a learner error and its correction. This is obviously not the same as the more ambitious question of whether a system can be built to detect and correct errors on the basis of web counts alone, and this is a distinction worth clarifying. Any system that successfully detects and corrects an error will need to accomplish three tasks[1]: (1) find a part of the user input that contains an error (*error detection*). (2) find one or multiple alternative string(s) for the alleged error (*candidate generation*) and (3) score the alternatives and the original to determine which alternative (if any) is a likely correction (*error correction*). Here, we are only concerned with the third task, specifically the comparison between the incorrect and the correct choice. This is an easily measured task, and is also a minimum requirement for any language model or language model approximation: if the model cannot distinguish an error from a well-formed string, it will not be useful.

[1] Note that these tasks need not be addressed by separate components. A contextual classifier for preposition choice, for example, can generate a probability distribution over a set of prepositions (candidate generation). If the original preposition choice has lower probability than one or more other prepositions, it is a potential error (error detection), and the prepositions with higher probability will be potential corrections (error correction).

We focus on two prominent learner errors in this study: preposition inclusion and choice and article inclusion and choice. These errors are among the most frequent learner errors (they comprise nearly one third of all errors in the learner corpus used in this study).

In this study, we compare three web data sources: The public Bing API, Google API, and the Google 5-gram language model. We also pay close attention to strategies of query formulation. The questions we address are summarized as follows:

Can web data be used to distinguish learner errors from correct phrases?

What is the better resource for web-data: the Bing API, the Google API, or the Google 5-gram data?

What is the best query formulation strategy when using web search results for this task? How much context should be included in the query?

2 Related Work

Hermet et al. (2008) use web search hit counts for preposition error detection and correction in French. They use a set of confusable prepositions to create a candidate set of alternative prepositional choices and generate queries for each of the candidates and the original. The queries are produced using linguistic analysis to identify both a governing and a governed element as a minimum meaningful context. On a small test set of 133 sentences, they report accuracy of 69.9% using the Yahoo! search engine.

Yi et al. (2008) target article use and collocation errors with a similar approach. Their system first analyzes the input sentence using part-of-speech tagging and a chunk parser. Based on this analysis, potential error locations for determiners and verb-noun collocation errors are identified. Query generation is performed at three levels of granularity: the sentence (or clause) level, chunk level and word level. Queries, in this approach, are not exact string searches but rather a set of strings combined with the chunk containing the potential error through a boolean operator. An example for a chunk level query for the sentence "I am learning economics at university" would be "[economics] AND [at university] AND [learning]". For article errors the hit count estimates (normalized for query length) are used directly. If the ratio of the normalized hit count estimate for the alternative article choice to the normalized hit count estimate of the original choice exceeds a manually determined threshold, the alternative is suggested as a correction. For verb-noun collocations, the situation is more complex since the system does not automatically generate possible alternative choices for noun/verb collocations. Instead, the snippets (document summaries) that are returned by the initial web search are analyzed and potential alternative collocation candidates are identified. They then submit a second round of queries to determine whether the suggestions are more frequent than the original collocation. Results on a 400+ sentence corpus of learner writing show 62% precision and 41% recall for determiners, and 30.7% recall and 37.3% precision for verb-noun collocation errors.

Tetreault and Chodorow (2009) make use of the web in a different way. Instead of using global web count estimates, they issue queries with a region-specific restriction and compare statistics across regions. The idea behind this approach is that regions that have a higher density of non-native speakers will show significantly higher frequency of erroneous productions than regions with a higher proportion of native speakers. For example, the verb-preposition combinations *married to* versus *married with* show very different counts in the UK versus France regions. The ratio of counts for *married to*/*married with* in the UK is 3.28, whereas it is 1.18 in France. This indicates that there is significant over-use of *married with* among native French speakers, which serves as evidence that this verb-preposition combination is likely to be an error predominant for French learners of English. They test their approach on a list of known verb-preposition errors. They also argue that, in a state-of-the-art preposition error detection system, recall on the verb-preposition errors under investigation is still so low that systems can only benefit from increased sensitivity to the error patterns that are discoverable through the region web estimates.

Bergsma et al (2009) are the closest to our work. They use the Google N-gram corpus to disambiguate usage of 34 prepositions in the *New York Times* portion of the Gigaword corpus. They use a sliding window of n-grams (n ranging from 2 to 5) across the preposition and collect counts for all resulting n-grams. They use two different methods to combine these counts. Their *SuperLM* model combines the counts as features in a linear SVM classifier, trained on a subset of the data. Their *SumLM* model is simpler, it sums all log counts across the n-grams. The preposition with the highest score is then predicted for the given context. Accuracy on the *New York Times* data in these experiments reaches 75.4% for SuperLM and 73.7% for SumLM.

Our approach differs from Bergsma et al. in three crucial respects. First, we evaluate insertion, deletion, and substitution operations, not just substitution, and we extend our evaluation to article errors. Second, we focus on finding the best query mechanism for each of these operations, which requires only a single query to the Web source. Finally, the focus of our work is on learner error detection, so we evaluate on real learner data as opposed to well-formed news text. This distinction is important: in our context, evaluation on edited text artificially inflates both precision and recall because the context surrounding the potential error site is error-free whereas learner writing can be, and often is, surrounded by errors. In addition, *New York Times* writing is highly idiomatic while learner productions often include unidiomatic word choices, even though the choice may not be considered an error.

3 Experimental Setup

3.1 Test Data

Our test data is extracted from the *Cambridge University Press Learners' Corpus* (CLC). Our version of CLC currently contains 20 million words from non-native English essays written as part of one of Cambridge's English language proficiency tests (ESOL) – at all proficiency levels. The essays are annotated for error type, erroneous span and suggested correction. We perform a number of preprocessing steps on the data. First, we correct all errors that were flagged as being spelling errors. Spelling errors that were flagged as morphology errors were left alone. We also changed confusable words that are covered by MS Word. In addition, we changed British English spelling to American English. We then eliminate all annotations for non-pertinent errors (i.e. non-preposition/article errors, or errors that do not involve any of the targeted prepositions), but we retain the original (erroneous) text for these. This makes our task harder since we will have to make predictions in text containing multiple errors, but it is more realistic given real learner writing. Finally, we eliminate sentences containing nested errors (where the annotation of one error contains an annotation for another error) and multiple article/preposition errors. Sentences that were flagged for a replacement error but contained no replacement were also eliminated from the data. The final set we use consists of a random selection of 9,006 sentences from the CLC with article errors and 9,235 sentences with preposition errors.

3.2 Search APIs and Corpora

We examine three different sources of data to distinguish learner errors from corrected errors. First, we use two web search engine APIs, Bing and Google. Both APIs allow the retrieval of a *page-count estimate* for an exact match query. Since these estimates are provided based on proprietary algorithms, we have to treat them as a "black box". The third source of data is the Google 5-gram corpus (Linguistic Data Consortium 2006) which contains n-grams with n ranging from 1 to 5. The count cutoff for unigrams is 200, for higher order n-grams it is 40.

3.3 Query Formulation

There are many possible ways to formulate an exact match (i.e. quoted) query for an error and its correction, depending on the amount of context that is included on the right and left side of the error. Including too little context runs the risk of missing the linguistically relevant information for determining the proper choice of preposition or determiner. Consider, for example, the sentence *we rely most of/on friends*. If we only include one word to the left and one word to the right of the preposition, we end up with the queries "most on friends" and "most of friends" - and the web hit count estimate may tell us that the latter is more frequent than the former. However, in this example, the verb *rely* determines the choice of preposition and when it is included in the query as in "rely most on friends" versus "rely most of friends", the estimated hit counts might correctly reflect the incorrect versus correct choice of preposition. Extending the query to cover too much of the context, on the other hand, can lead to low or zero web hit estimates because of data sparseness - if we include the pronoun *we* in the query as in "we rely most on friends" versus "we rely most of friends", we get zero web count estimates for both queries.

Another issue in query formulation is what strategy to use for corrections that involve deletions and insertions, where the number of tokens changes. If, for example, we use queries of length 3, the question for deletion queries is whether we use two words to the left and one to the right of the deleted word, or one word to the left and two to the right. In other words, in the sentence *we traveled to/0 abroad last year*, should the query for the correction (deletion) be "we traveled abroad" or "traveled abroad last"?

Finally, we can employ some linguistic information to design our query. By using part-of-speech tag information, we can develop heuristics to include a governing content word to the left and the head of the noun phrase to the right.

The complete list of query strategies that we tested is given below.

SmartQuery: using part-of-speech information to include the first content word to the left and the head noun to the right. If the content word on the left cannot be established within a window of 2 tokens and the noun phrase edge within 5 tokens, select a fixed window of 2 tokens to the left and 2 tokens to the right.

FixedWindow Queries: include *n* tokens to the left and *m* tokens to the right. We experimented with the following settings for *n* and *m*: 1_1, 2_1, 1_2, 2_2, 3_2, 2_3. The latter two 6-grams were only used for the API's, because the Google corpus does not contain 6-grams.

FixedLength Queries: queries where the length in tokens is identical for the error and the correction. For substitution errors, these are the same as the corresponding *FixedWindow* queries, but for substitutions and deletions we either favor the left or right context to include one additional token to make up for the deleted/inserted token. We experimented with trigrams, 4-grams, 5-grams and 6-grams, with left and right preference for each, they are referred to as *Left4g* (4-gram with left preference), etc.

3.4 Evaluation Metrics

For each query pair <q_{error}, $q_{correction}$>, we produce one of three different outcomes:

correct (the query results favor the correction of the learner error over the error itself):

$count(q_{correction}) > count(q_{error})$

incorrect (the query results favor the learner error over its correction):

$count(q_{error}) >= count(q_{correction})$

$where(count(q_{error}) \neq 0$ OR

$count(q_{correction}) \neq 0)$

noresult:

$count(q_{correction}) = count(q_{error}) = 0$

For each query type, each error (preposition or article), each correction operation (deletion, insertion, substitution) and each web resource (Bing API, Google API, Google N-grams) we collect these counts and use them to calculate three different metrics. *Raw accuracy* is the ratio of correct predictions to all query pairs:

$$Raw\ accuracy = \frac{corr}{corr + incorr + noresult}$$

We also calculate accuracy for the subset of query pairs where at least one of the queries resulted in a successful hit, i.e. a non-zero result. We call this metric *Non-Zero-Result-Accurracy (NZRA)*, it is the ratio of correct predictions to incorrect predictions, ignoring noresults:

$$NonZeroResultAccuracy = \frac{corr}{corr + incorr}$$

Finally, *retrieval ratio* is the ratio of queries that returned non-zero results:

4 Results

We show results from our experiments in Table 1 - Table 6. Since space does not permit a full tabulation of all the individual results, we restrict ourselves to listing only those query types that achieve best results (highlighted) in at least one metric.

Google 5-grams show significantly better results than both the Google and Bing APIs. This is good news in terms of implementation, because it frees the system from the vagaries involved in relying on search engine page estimates: (1) the latency, (2) query quotas, and (3) fluctuations of page estimates over time. The bad news is that the 5-gram corpus has much lower retrieval ratio because, presumably, of its frequency cutoff. Its use also limits the maximum length of a query to a 5-gram (although neither of the APIs outperformed Google 5-grams when retrieving 6-gram queries).

The results for substitutions are best, for fixed window queries. For prepositions, the SmartQueries perform with about 86% NZRA while a fixed length 2_2 query (targeted word with a ±2-token window) achieves the best results for articles, at about 85% (when there was at least one non-zero match). Retrieval ratio for the prepositions was about 6% lower than retrieval ratio for articles – 41% compared to 35%.

The best query type for insertions was fixed-length LeftFourgrams with about 95% NZRA and 71% retrieval ratio for articles and 89% and 78% retrieval ratio for prepositions. However, LeftFourgrams favor the suggested rewrites because, by keeping the query length at four tokens, the original has more syntactic/semantic context. If the original sentence contains *is referred as the* and the annotator inserted *to* before *as*, the original query will be *is referred as the* and the correction query *is referred to as*.

Conversely, with deletion, having a fixed window favors the shorter rewrite string. The best query types for deletions were: 2_2 queries for articles (94% NZRA and 46% retrieval ratio) and SmartQueries for prepositions (97% NZRA and 52% retrieval ratio). For prepositions the fixed length 1_1 query performs about the same as the SmartQueries, but that query is a trigram (or smaller at the edges of a sentence) whereas the average length of SmartQueries is 4.7 words for prepositions and 4.3 words for articles. So while the coverage for SmartQueries is much lower, the longer query string cuts the risk of matching on false positives.

The Google 5-gram Corpus differs from search engines in that it is sensitive to upper and lower case distinctions and to punctuation. While intuitively it seemed that punctuation would hurt n-gram performance, it actually helps because the punctuation is an indicator of a clause boundary. A recent Google search for *have a lunch* and *have lunch* produced estimates of about 14 million web pages for the former and only 2 million for the latter. Upon inspecting the snippets for *have a lunch*, the next word was almost always a noun such as *menu, break, date, hour, meeting, partner,* etc. The relative frequencies for *have a lunch* would be much different if a clause boundary marker were required. The 5-gram corpus also has sentence boundary markers which is especially helpful to identify changes at the beginning of a sentence.

	non-zero-result accuracy			retrieval ratio			raw accuracy		
Query type	B-API	G-API	G-Ngr	B-API	G-API	G-Ngr	B-API	G-API	G-Ngr
SmartQuery	0.8637	0.9548	**0.9742**	0.8787	0.8562	0.5206	0.7589	0.8176	0.5071
1_1	0.4099	0.9655	0.9721	**0.9986**	0.9978	0.9756	0.4093	**0.9634**	0.9484

Table 1: Preposition deletions (1395 query pairs).

	non-zero-result accuracy			retrieval ratio			raw accuracy		
Query type	B-API	G-API	G-Ngr	B-API	G-API	G-Ngr	B-API	G-API	G-Ngr
Left4g	0.7459	0.8454	**0.8853**	0.9624	0.9520	0.7817	0.7178	0.8048	0.6920
1_1	0.5679	0.2983	0.3550	**0.9973**	0.9964	0.9733	0.5661	0.2971	0.3456
Right3g	0.6431	0.8197	0.8586	0.9950	0.9946	0.9452	0.6399	**0.8152**	0.8116

Table 2: Preposition insertions (2208 query pairs).

	non-zero-result accuracy			retrieval ratio			raw accuracy		
Query type	B-API	G-API	G-Ngr	B-API	G-API	G-Ngr	B-API	G-API	G-Ngr
SmartQuery	0.7396	0.8183	**0.8633**	0.7987	0.7878	0.4108	0.5906	0.6446	0.5071
1_1=L3g=R3g	0.4889	0.6557	0.6638	**0.9870**	0.9856	0.9041	0.4826	0.6463	0.6001
1_2=R4g	0.6558	0.7651	0.8042	0.9178	0.9047	0.6383	0.6019	**0.6921**	0.5133

Table 3: Preposition substitutions (5632 query pairs).

	non-zero-result accuracy			retrieval ratio			raw accuracy		
Query type	B-API	G-API	G-Ngr	B-API	G-API	G-Ngr	B-API	G-API	G-Ngr
2_2	0.7678	0.9056	**0.9386**	0.8353	0.8108	0.4644	0.6414	0.7342	0.4359
1_1	0.3850	0.8348	0.8620	**0.9942**	0.9924	0.9606	0.3828	0.8285	0.8281
1_2	0.5737	0.8965	0.9097	0.9556	0.9494	0.7920	0.5482	**0.8512**	0.7205

Table 4: Article deletions (2769 query pairs).

	non-zero-result accuracy			retrieval ratio			raw accuracy		
Query type	B-API	G-API	G-Ngr	B-API	G-API	G-Ngr	B-API	G-API	G-Ngr
Left4g	0.8292	0.9083	**0.9460**	0.9505	0.9428	0.7072	0.7880	0.8562	0.6690
1_1	0.5791	0.3938	0.3908	**0.9978**	0.9975	0.9609	0.5777	0.3928	0.3755
Left3g	0.6642	0.8983	0.8924	0.9953	0.9955	0.9413	0.6611	**0.8942**	0.8400

Table 5: Article insertions (5520 query pairs).

	non-zero-result accuracy			retrieval ratio			raw accuracy		
Query type	B-API	G-API	G-Ngr	B-API	G-API	G-Ngr	B-API	G-API	G-Ngr
2_2=Left5g= Right5g	0.6970	0.7842	**0.8486**	0.8285	0.8145	0.4421	0.5774	0.6388	0.3752
1_1=L3g=R3g	0.4385	0.7063	0.7297	**0.9986**	0.9972	0.9596	0.4379	0.7043	0.7001
1_2=R4g	0.5268	0.7493	0.7917	0.9637	0.9568	0.8033	0.5077	**0.7169**	0.6360

Table 6: Article substitutions (717 query pairs).

5 Error Analysis

We manually inspected examples where the matches on the original string were greater than matches on the corrected string. The results of this error analysis are shown in table 7. Most of the time, (1) the context that determined article or preposition use and choice was not contained within the query. This includes, for articles, cases where article usage depends either on a previous mention or on the intended sense of a polysemous head noun. Some other patterns also emerged. Sometimes (2) both and the original and the correction seemed equally good in the context of the entire sentence, for example *it's very important* *to* *us* and *it's very important* *for* *us*. In other cases, (3) there was another error in the query string (recall that we retained all of the errors in the original sentences that were not the targeted error). Then there is a very subjective category (4) where the relative n-gram frequencies are unexpected, for example where the corpus has 171 trigrams *guilty for you* but only 137 for *guilty about you.* These often occur when both of the frequencies are either low and/or close. This category includes cases where it is very likely that one of the queries is retrieving an n-gram whose right edge is the beginning of a compound noun (as in with the trigram *have a lunch*). Finally, (5) some of the "corrections" either introduced an error into the sentence or the original and "correction" were equally bad. In this category, we also include British English article usage like *go to hospital.* For prepositions, (6) some of the corrections changed the meaning of the sentence – where the disambiguation context is often not in the sentence itself and either choice is syntactically correct, as in *I will buy it* *from* *you* changed to *I will buy it* *for* *you.*

	Articles		Preps	
	freq	ratio	freq	ratio
1.N-gram does not contain necessary context	187	.58	183	.52
2.Original and correction both good	39	.12	51	.11
3.Other error in n-gram	30	.9	35	.10
4.Unexpected ratio	36	.11	27	.09
5.Correction is wrong	30	.9	30	.08
6.Meaning changing	na	na	24	.07

Table 7: Error analysis

If we count categories 2 and 5 in Table 7 as not being errors, then the error rate for articles drops 20% and the error rate for prepositions drops 19%.

A disproportionately high subcategory of query strings that did not contain the disambiguating context (category 1) was at the edges of the sentence – especially for the LeftFourgrams at the beginning of a sentence where the query will always be a bigram.

6 Conclusion and Future Work

We have demonstrated that web source counts can be an accurate predictor for distinguishing between a learner error and its correction - as long as the query strategy is tuned towards the error type. Longer queries, i.e. 4-grams and 5-grams achieve the best non-zero-result accuracy for articles, while SmartQueries perform best for preposition errors. Google N-grams across the board achieve the best non-zero-result accuracy, but not surprisingly they have the lowest retrieval ratio due to count cutoffs. Between the two search APIs, Bing tends to have better retrieval ratio, while Google achieves higher accuracy.

In terms of practical use in an error detection system, a general "recipe" for a high precision component can be summarized as follows. First, use the Google Web 5-gram Corpus as a web source. It achieves the highest NZRA, and it avoids multiple problems with search APIs: results do not fluctuate over time, results are real n-gram counts as opposed to document count estimates, and a local implementation can avoid the high latency associated with search APIs. Secondly, carefully select the query strategy depending on the correction operation and error type.

We hope that this empirical investigation can contribute to a more solid foundation for future work in error detection and correction involving the web as a source for data. While it is certainly not sufficient to use only web data for this purpose, we believe that the accuracy numbers reported here indicate that web data can provide a strong additional signal in a system that combines different detection and correction mechanisms. One can imagine, for example, multiple ways to combine the n-gram data with an existing language model. Alternatively, one could follow Bergsma et al. (2009) and issue not just a single pair of queries but a whole series of queries and sum over the results. This would increase recall since at least some of the shorter queries are likely to return non-zero results. In a real-time system, however, issuing several dozen queries per potential error location and potential correction could cause performance issues. Finally, the n-gram counts can be incorporated as one of the features into a system such as the one described in Gamon (2010) that combines evidence from various sources in a principled way to optimize accuracy on learner errors.

Acknowledgments

We would like to thank Yizheng Cai for making the Google web ngram counts available through a web service and to the anonymous reviewers for their feedback.

References

Eric Steven Atwell. 1987. How to detect grammatical errors in a text without parsing it. *Proceedings of the 3rd EACL, Copenhagen, Denmark*, pp 38 - 45.

Michele Banko and Eric Brill. 2001a. Mitigating the paucity-of-data problem: Exploring the effect of training corpus size on classifier performance for natural language processing. In James Allan, editor, *Proceedings of the First International Conference on Human Language Technology Research*. Morgan Kaufmann, San Francisco.

Michele Banko and Eric Brill. 2001b. Scaling to very very large corpora for natural language disambiguation. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics and the 10th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 26–33, Toulouse, France.

Shane Bergsma, Dekang Lin, and Randy Goebel. 2009. Web-scale n-gram models for lexical disambiguation. In Proceedings for the 21st International Joint *Conference on Artificial Intelligence*, pp. 1507 – 1512.

Martin Chodorow, Joel Tetreault, and Na-Rae Han. 2007. Detection of grammatical errors involving prepositions. In *Proceedings of the Fourth ACL-SIGSEM Workshop on Prepositions,* pp. 25-30.

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

Rachele De Felice and Stephen Pulman. 2008. A classifier-based approach to preposition and determiner error correction in L2 English. *COLING*. Manchester, UK.

Michael Gamon, Jianfeng Gao, Chris Brockett, Alexander Klementiev, William Dolan, Dmitriy Belenko, and Lucy Vanderwende. 2008. Using contextual speller techniques and language modeling for ESL error correction. In P*roceedings of IJCNLP*, Hyderabad, India.

Michael Gamon. 2010. Using mostly native data to correct errors in learners' writing. In *Proceedings of NAACL*.

Na-Rae Han, Martin Chodorow, and Claudia Leacock. 2006. Detecting errors in English article usage by non-native speakers. *Natural Language Engineering*, 12(2), 115-129.

Matthieu Hermet, Alain Désilets, Stan Szpakowicz. 2008. Using the web as a linguistic resource to automatically correct lexico-syntactic errors. In *Proceedings of the 6th Conference on Language Resources and Evaluation* (LREC), pp. 874-878.

Frank Keller and Mirella Lapata. 2003. Using the web to obtain frequencies for unseen bigrams. *Computational Linguistics,* 29(3): 459-484.

Adam Kilgariff. 2007. Googleology is bad science. *Computational Linguistics* 33(1): 147-151.

Mirella Lapata and Frank Keller. 2005. Web-Based Models for Natural Language Processing. *ACM Transactions on Speech and Language Processing (TSLP)*, 2(1):1-31.

Linguistic Data Consortium. 2006. Web 1T 5-gram version 1. http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2006T13 .

Joel Tetreault and Martin Chodorow. 2008. The ups and downs of preposition error detection in ESL. *COLING*. Manchester, UK.

Joel Tetreault and Martin Chodorow. 2009. Examining the use of region web counts for ESL error detection. *Web as Corpus Workshop* (WAC-5), San Sebastian, Spain.

Xing Yi, Jianfeng Gao and Bill Dolan. 2008. A web-based English proofing system for English as a second language users. In *Proceedings of the Third International Joint Conference on Natural Language Processing* (IJCNLP). Hyderabad, India.

Zhu, X. and Rosenfeld, R. 2001. Improving trigram language modeling with the world wide web. In *Proceedings of International Conference on Acoustics Speech and Signal Processing*. Salt Lake City.