Bilingual Random Walk Models for Automated Grammar Correction of ESL Author-Produced Text

Randy West and **Y. Albert Park**
Department of Computer Science & Engineering
University of California, San Diego
La Jolla, CA 92093-0533
{rdwest,yapark}@cs.ucsd.edu

Roger Levy
Department of Linguistics
University of California, San Diego
La Jolla, CA 92093-0533
rlevy@ucsd.edu

Abstract

We present a novel noisy channel model for correcting text produced by English as a second language (ESL) authors. We model the English word choices made by ESL authors as a random walk across an undirected bipartite dictionary graph composed of edges between English words and associated words in an author's native language. We present two such models, using cascades of weighted finite-state transducers (wFSTs) to model language model priors, random walk-induced noise, and observed sentences, and expectation maximization (EM) to learn model parameters after Park and Levy (2011). We show that such models can make intelligent word substitutions to improve grammaticality in an unsupervised setting.

1 Introduction

How do language learners make word choices as they compose text in a language in which they are not fluent? Anyone who has attempted to learn a foreign language can attest to spending a great deal of time leafing through the pages of a bilingual dictionary. However, dictionaries, especially those without a wealth of example sentences or accompanying word sense information, can often lead even the most scrupulous of language learners in the wrong direction. Consider an example: the English noun "head" has several senses, e.g. the physical head and the head of an organization. However, the Japanese *atama* can only mean the physical head or mind, and likewise *shuchou*, meaning "chief," can only map to the second sense of head. A native English speaker and Japanese learner faced with the choice of these two words and no additional explanation of which Japanese word corresponds to which sense is liable to make a mistake on the flip of a coin.

One could of course conceive of more subtle examples where the semantics of a set of choices are not so blatantly orthogonal. "Complete" and "entire" are synonyms, but they are not necessarily interchangeable. "Complete stranger" is a common two-word phrase, but "entire stranger" sounds completely strange, if not entirely ungrammatical, to the native English speaker, who will correct "entire" to "complete" in a surprisingly automatic fashion. Thus, correct word choice in non-native language production is essential not only to the preservation of intended meaning, but also to fluent expression of the correct meaning.

The development of software to correct ESL text is valuable for both learning and communication. A language learner provided instant grammaticality feedback during self-study is less likely to fall into patterns of misuse, and the comprehension difficulties one may encounter when corresponding with non-native speakers would be ameliorated by an automated system to improve text fluency. Additionally, since machine-translated text is often ungrammatical, automated grammar correction algorithms can be deployed as part of a machine translation system to improve the quality of output.

We propose that word choice production errors on the part of the language learner can be modeled as follows. Given an observed word and an undirected bipartite graph with nodes representing

Proceedings of the Sixth Workshop on Innovative Use of NLP for Building Educational Applications, pages 170–179,
Portland, Oregon, 24 June 2011.

words in one of two languages, i.e. English and the sentence author's native tongue, and edges between words in each language and their dictionary translation in the other (see Figure 1 for an example), there exists some function $f \mapsto [0, 1]$ that defines the parameters of a random walk along graph edges, conditioned on the source word. By composing this graph with a language model prior such as an n-gram model or probabilistic context-free grammar, we can "correct" an observed sentence by inferring the most likely unobserved sentence from which it originated.

More concretely, given that we know f, we can compute $\text{argmax}_{\boldsymbol{w'}} p(\boldsymbol{w'}|\boldsymbol{w}, f, \theta)$, where $\boldsymbol{w}$ is the observed sentence, θ is the language model, and $\boldsymbol{w'}$ is the "corrected," unobserved sentence. Under this view, some $\boldsymbol{w'}$ drawn from the distribution θ is subjected to some noise process f, which perturbs the sentence author's intended meaning and outputs $\boldsymbol{w}$. We perform this computation in the standard way from the statistical machine translation (SMT) literature (Brown et al., 1993), namely by using Bayes' theorem to write

$$p(\boldsymbol{w'}|\boldsymbol{w}, f, \theta) = \frac{p(\boldsymbol{w'}|\theta)p(\boldsymbol{w}|\boldsymbol{w'}, f, \theta)}{p(\boldsymbol{w}|\theta)}$$

Since the denominator of the RHS is independent of $\boldsymbol{w'}$, we can rewrite our argmax as

$$\underset{\boldsymbol{w'}}{\text{argmax}}\, p(\boldsymbol{w'}|\theta)p(\boldsymbol{w}|\boldsymbol{w'}, f, \theta)$$

We have now decomposed our original equation into two manageable parts, a prior belief about the grammaticality of an unobserved sentence $\boldsymbol{w'}$, which we can compute using a language model θ learned separately using standard supervised techniques (in particular, n-gram estimation), and the probability of the observed sentence $\boldsymbol{w}$ given $\boldsymbol{w'}$, f, and θ. Together, these constitute a noisy channel model from information theory (Shannon, 1948). All that remains is to learn an appropriate f, for which we will employ unsupervised methods, namely expectation maximization.

The rest of this paper is organized as follows. In Section 2, we will discuss related work. In Section 3, we will present the implementation, methodology and results of two experiments with different f. In Section 4, we will discuss our experimental results, and we will conclude in Section 5.

2 Related Work

The literature on automated grammar correction is mostly focused on rule-based methods and error identification rather than correction. However, there has been a recent outgrowth in the application of machine translation (MT) techniques to address the problem of single-language grammar correction. Park and Levy (2011) propose a noisy channel model for learning to correct various types of errors, including article and preposition errors, word-form errors, and spelling mistakes, to which this paper is an extension. As the present work builds on Park and Levy's basic model, we will reserve a more detailed discussion of their work for Section 3.

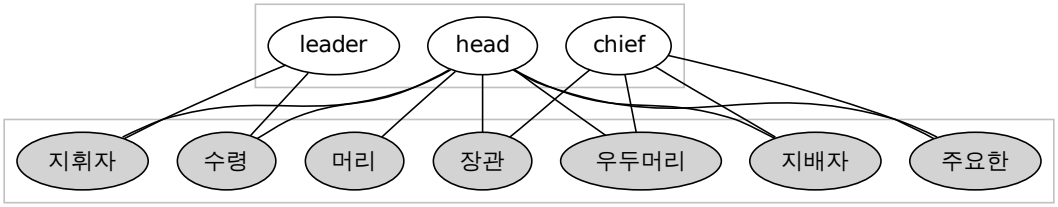
Brockett et al. (2006) use phrasal SMT techniques to identify and correct mass noun errors of ESL students with some success, but they correct no other production error classes to our knowledge.

Lee and Seneff (2006) learn a method to aid ESL students in language acquisition by reducing sentences to their canonical form, i.e. a lemmatized form devoid of articles, prepositions, and auxiliaries, and then building an over-specified lattice by reinserting all word inflections and removed word classes. They then score this lattice using a trigram model and PCFG. While this method has many advantages, it does not take into account the full context of the original sentence.

Kok and Brockett (2010) use random walks over bi- and multilingual graphs generated by aligning English sentences with translations in 10 other European languages to learn paraphrases, which they then evaluate in the context of the original sentence. While their approach shares many high-level similarities with ours, both their task, paraphrasing correct sentences, and the details of their methodology are divergent from the present work.

Désilets and Hermet (2009) employ round-trip machine translation from L1 to L2 and back again to correct second language learner text by keeping track of the word alignments between translations. They operate on a very similar hypothesis to that of this work, namely that language learners make overly-literal translations when the produce text in their second language. However, they go about correcting these errors in a very different way than the present work, which is novel to the best of

Figure 1: Example English-Korean dictionary graph for a subset of the edges out of the English *head*, *leader*, and *chief*.



our knowledge, and their technique of using error-annotated sentences for evaluation makes a comparison difficult.

3 Model Implementation and Experiments

We present the results of two experiments with different random walk parametrizations. We begin by describing our dataset, then proceed to an overview of our model and experimental procedures, and finally detail the experiments themselves.

3.1 Dataset

We use the dataset of Park and Levy (2011), a collection of approximately 25,000 essays comprised of 478,350 sentences scraped from web postings made by Korean ESL students studying for the Test of English as a Foreign Language (TOEFL). Of these, we randomly select 10,000 sentences for training, 504 as a development set, and 1017 held out for final model evaluation.

Our English-Korean dictionary is scraped from `http://endic2009.naver.com`, a widely-used and trusted online dictionary source in South Korea. We are unfortunately unaware of any freely available, downloadable English-Korean dictionary databases.

3.2 Model and Experimental Procedures

3.2.1 Overview

The bulk of our experimental methodology and machinery is borrowed from Park and Levy (2011), so we will summarize that portion of it only briefly here. At a high level, there are three major components to the model of a sentence: a language prior, a noise model, and an observed sentence. Each of these is implemented as a wFST and composed together into a single transducer whose accepting paths represent all possibilities of transducing from an (unobserved) input sentence to the (observed) output sentence, with the path weight being associated probability. See Figure 2 for an example.

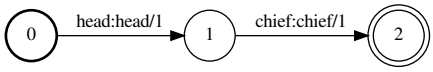
3.2.2 Language Model

For our language model, we use a Kneser-Ney smoothed trigram model learned from a version of the British National Corpus modified to use Americanized spellings (Chen and Goodman, 1996; Burnard, 1995). The implementation of an n-gram model as a wFST requires that each state represent a context, and so one must necessarily instantiate arcs for all words in the alphabet from each state. In order to reduce model size and minimize memory usage, it is standard practice to remove relatively uninformative higher-order n-grams from the model, but under the wFST regime one cannot, for example, remove some trigrams from a bigram context without removing all of them. Instead, we retain only the 1,000 most informative bigram *contexts*, as measured by the Kullback-Leibler divergence between each bigram context and its unigram counterpart. This is in contrast to standard cutoff models, which remove n-grams occurring less than some cutoff number of times in the corpus.

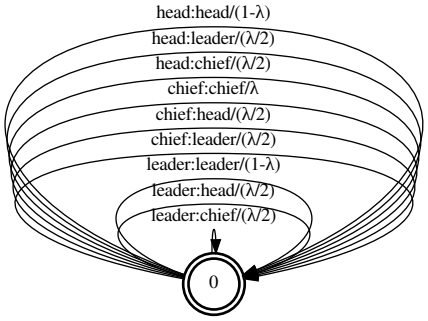
3.2.3 Noise Models

The structure of the noise wFST differs for each noise model; for our model of word-choice error, we can use a single initial/final state with arcs labeled with unobserved words as input, observed words as output, and a weight defined by the function f that governs the parameters of a random walk across our dictionary graph (again, see Figure 2 for an example). We will reserve the definition of f, which is

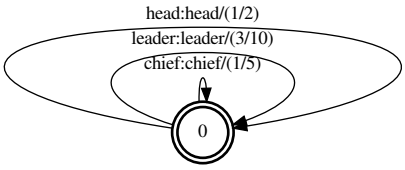
Figure 2: Example wFSTs for the sentence "head chief". From top to bottom, the pictured transducers are the observed sentence s, a noise model n with parameter λ, a unigram language model l representing the normalized frequency of each word, and the fully composed model, $l \circ n \circ s$.



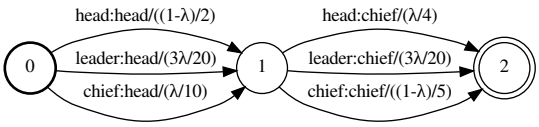
Observed Sentence



Noise Model



Language Model



Composed Model

different for each experiment, for Section 3.3.

We have thus far proceeded by describing the construction of an ideal noise model that completely implements the dictionary graph described previously. However, due to the size of the dictionary graph, such a model would be computationally prohibitive[1]. Moreover, we must handle the non-trivial peculiars of arbitrary lookups in a roughly lemmatized dictionary and preservation of word forms through random walks, which we discuss now. Among its various capabilities, the CELEX database (Baayen et al., 1995) provides interfaces for mapping arbitrary English words to their lemmas, querying for lemma syntactic (sub)classes, and discovering the morphological inflectional features of arbitrary words. We use these capabilities in conjunction with unigram frequencies from our language model and a standard stop word filter to build abridged sets of random walk candidates as in Algorithm 1.

Algorithm 1 Build an abridged set of random walk candidates C for an observed word w s.t. each $c_i \in C$ has syntactic and morphological characteristics similar to w and is in the top m such candidates as sorted by word frequency.

Let $G = (V, E)$ be the undirected dictionary graph, m the max candidates per word, B the set of stop words, I the set of inflectional features of w, and C the set of random walk candidates for w, initially $\{\}$
if $w \in B$ **then**
  return $\{\}$
end if
for lemmas l of w **do**
  Let S be the set of syntactic classes of l
  for l' generated from a random walk of length 2 in G from l **do**
    if $S \cap \{\text{syntactic classes of } l'\} \neq \{\}$ **then**
      for words w' related to l' **do**
        if $I \cap \{\text{inflectional features of } w'\} \neq \{\} \wedge w' \notin B$ **then**
          $C \leftarrow C \cup \{w'\}$
        end if
      end for
    end if
  end for
end for
if $|C| > m$ **then**
  $C \leftarrow$ top m members of C by word frequency
end if
return C

3.2.4 Sentence Models

Sentences are simply identity transducers, i.e. wFSTs with $n + 1$ states for a sentence of length n and a single arc between each state $0 \leq i < n$ and state $i+1$ labeled with input and output token i from

[1]The maximum degree of the dictionary graph is 515, meaning that the upper bound on the number of paths in a random walk of length 2 is $515^2 = 265,225$!

the sentence and weight 1.

3.2.5 Training and Decoding

For training, we hold language model parameters constant and use expectation maximization (Dempster et al., 1977) to learn noise model parameters as follows. We replace language model input symbols and sentence model output symbols with the empty symbol ϵ and use the V-expectation semiring of Eisner (2002) to annotate noise model arcs with initial parameter values. This is our M-step. Then, we compose the language, noise, and sentence models, which produces a transducer with only ϵ-labeled arcs, and use ϵ-removal to move expectation information into a single state from which we can easily read off expected noise model parameter counts thanks to the V-expectation semiring's bookkeeping (Eisner, 2002; Mohri, 2001). We repeat this process over a batch of training sentences and add the results together to yield a final vector of expected counts. This is our E-step. Finally, we normalize the expected parameter counts to recompute our parameters and rebuild the noise model in a repetition of the M-step. This process goes back and forth from E- to M-step until the parameters converge within some threshold.

The decoding or inference process is performed in a similar fashion, the main difference being that we use the negative log Viterbi semiring for computing shortest paths instead of the V-expectation semiring. We first build a new noise model for each sentence using the parameter values learned during training. Then, the language, noise, and sentence models (sans ϵ substitutions) are composed together, and the shortest path is computed.

3.2.6 wFST Implementation

All wFST manipulation is performed using OpenFST (Allauzen et al., 2007), an open source weighted finite-state transducer library written in C++. Additionally, we use the V-expectation semiring code of Dreyer et al. (2008) for training.

3.2.7 Evaluation

The most probable unobserved sentence $\boldsymbol{w'}$ from which the observed sentence $\boldsymbol{w}$ was generated under our model, $\text{argmax}_{\boldsymbol{w'}}\, p(\boldsymbol{w'}|\theta)p(\boldsymbol{w}|\boldsymbol{w'}, f, \theta)$, can be read off from the input of the transducer produced during the decoding process. In order to evaluate its quality versus the observed ESL sentence, we use the METEOR[2] and BLEU evaluation metrics for machine translation (Lavie and Agarwal, 2007; Papineni et al., 2002). This evaluation is performed using a set of human-corrected sentences gathered via Amazon Mechanical Turk, an online service where workers are paid to perform a short task, and further filtered for correctness by an undergraduate research assistant. 8 workers were assigned to correct each sentence from the development and evaluation sets described in Section 3.1, and so after filtering we had 8 or fewer unique corrected versions per sentence available for evaluation. We note that the use of METEOR and BLEU is justified inasmuch as the process of grammar correction is translation from an ungrammatical "language" to a grammatical one (Park and Levy, 2011). However, it is far from perfect, as we shall see shortly.

While human evaluation is far too costly to attempt at every step during development, it is very worthwhile to examine our corrections through a human eye for final evaluation, especially given the somewhat tenuous suitability of METEOR and BLEU for our evaluation task. In order to facilitate this, we designed a simple task, again using Amazon Mechanical Turk, where native English speakers are presented with side-by-side ESL and corrected sentences and asked to choose which is more correct. Workers are instructed to "judge whether the corrected sentence improves the grammaticality and/or fluency of the ESL sentence without changing the ESL sentence's basic meaning." They are then presented with two questions per sentence pair:

1. *Question:* "Between the two sentences listed above, which is more correct?"
   Answer choices: "ESL sentence is more correct," "Corrected sentence is more correct," "Both are equally correct," and, "The sentences are identical."

[2] Although the METEOR "synonymy" module may initially seem appropriate to our evaluation task, we find that it does little to improve or clarify evaluation results. For that reason, and moreover since we do not wish for differing forms of the same lemma to be given equal weight in a grammar correction task, we instead use the "exact" module for all evaluation in this paper.

2. *Question:* "Is the meaning of the corrected sentence significantly different from that of the ESL sentence?"
   Answer choices: "Yes, the two sentences do not mean the same thing," and, "No, the two sentences have roughly the same meaning."

Each task is 10 sentences long, 3 of which are identical filler sentences. When a worker mislabels more than one sentence as identical in any single task, the results for that task are thrown out and resubmitted for another worker to complete. We additionally require that each sentence pair be judged by 5 unique, U.S.-based workers.

3.3 Experiments

3.3.1 Experiment 1

Motivation and Noise Model For our first experiment, we assume that the probability of arriving at some word $w' \neq w$ after a random walk of length 2 from an observed word w is uniform across all w. This is perhaps not the most plausible model, but it serves as a baseline by which we can evaluate more complex models.

More concretely, we use a single parameter λ modeling the probability of walking two steps along the dictionary graph from an observed English word w to its Korean definition(s), and then back to some other English word $w' \neq w$. Since we treat unobserved words as transducer input and observed words as output, λ is normalized by $|\{w|w \neq w'\}|$, i.e. the number of edges with different input and output per input word, and $p(w|w) = 1 - \lambda$ such that $\sum_w p(w|w') = 1$.

Initialization and Other Settings We train two variations on the same model, setting m from Algorithm 1, i.e. the maximum number of allowed random walk candidates per word, to 5 and 10. We initialize λ to 0.01 for each.

Results We find that both variations converge after roughly 10 iterations[3]. The parameters learned are slightly lower than the initialization value ($\lambda = 0.01$), 0.007246 for the 5 candidate variation and 0.009528 for the 10 candidate variation. We interpret the parameter value disparity between the two model variations as follows. The larger the number of random walk candidates available for each observed word, the more likely that at least one of the candidates has a high probability in the sentence context, so it makes sense that the 10 candidate variation would yield a higher value for λ. Moreover, recalling that λ is normalized by the number of observed words $|\{w|w \neq w'\}|$ reachable from each unobserved candidate word w', it is reasonable that a higher value of λ would need to be learned in order to distribute enough probability mass to candidates that are highly probable in the sentence context.

The METEOR and BLEU scores for this Experiment are summarized in Table 1, and the final parameter values after 10 iterations are listed in Table 2. We discuss these in greater detail in Section 4.

Table 1: METEOR and BLEU scores for all experiments.

	METEOR	BLEU
ESL baseline	0.820802	0.715625
Exp. 1, 5 candidates	0.816055	0.708871
Exp. 1, 10 candidates	0.815703	0.708284
Exp. 2, 5 candidates	0.815162	0.707549
Exp. 2, 10 candidates	0.814533	0.706587

Table 2: Final parameter values after 10 iterations for Experiment 1 with 5 and 10 word random walk candidate limits.

	Max 5 Candidates	Max 10 Candidates
λ	0.007246	0.009528

3.3.2 Experiment 2

Motivation and Noise Model For our second experiment, we hypothesize that there is an inverse relationship between unobserved word frequency and random walk path probability. We motivate this by observing that when a language learner produces a common word, it is likely that she either meant to use that word or used it in place of a rarer word that she did not know. Likewise, when she uses a rare word, it is likely that she chose it above any of the

[3]Running on a Linux server with two quad-core Intel Xeon processors and 72GB of memory, training for all models in this paper takes around 4 hours per model. Note that decoding is a much quicker process, requiring less than one second per sentence.

common words that she knows. If the word that she chose was erroneous, then, it is most likely that she did not mean to use a common word but could have meant to use a different rare word with a subtle semantic difference. Hence, we should always prefer to replace observed words, regardless of their frequency, with rare words unless the language model overwhelmingly prefers a common word.

In order to model this hypothesis, we introduce a second parameter $\alpha < 0$ to which power the unigram frequency of each unobserved word w', $\text{freq}(w')$, is raised. The resulting full model is $p(w|w')_{w \neq w'} = \frac{\text{freq}(w')^\alpha \lambda}{|\{w|w \neq w'\}|}$ and $p(w|w) = 1 - \text{freq}(w)^\alpha \lambda$. We approximate the full model to simple coin flips by bucketing the unique word frequencies from the language model and initializing each bucket using its average frequency and some appropriate initial values of α and λ, leaving us with a number of parameters equal to the number of frequency buckets.

Initialization and Other Settings We train two variations on the same model, setting m from Algorithm 1 to 5 and 10. We initialize λ to 0.01 and α to -0.1 for each and use 10 frequency buckets.

Results As in Experiment 1, we find that both model variations converge after roughly 10 iterations. The random walk parameters learned for both variations in the highest frequency bucket, $\text{freq}(w')^\alpha \lambda \approx 0.004803$ and 0.004845 for 5 and 10 candidates, respectively, seem to validate our hypothesis that we should prefer rare unobserved words. However, the parameters learned for the proceeding buckets do not indicate the smooth positive slope that we might have hoped for, which we discuss further in Section 4. The 10 candidate variation learns consistently higher parameter values than the 5 candidate variation, and we interpret this disparity in the same way as in Experiment 1.

The METEOR and BLEU scores for this Experiment are summarized in Table 1, and the final parameter values after 10 iterations are listed in Table 3. We discuss these in greater detail in Section 4.

Table 3: Final parameter values after 10 iterations for Experiment 2 with 5 and 10 word random walk candidate limits.

Word Frequency (high to low)	Max 5 Candidates	Max 10 Candidates
Bucket 1	0.004803	0.004845
Bucket 2	0.031505	0.052706
Bucket 3	0.019211	0.036479
Bucket 4	0.006871	0.013130
Bucket 5	0.002603	0.005024
Bucket 6	0.000032	0.000599
Bucket 7	0.001908	0.003336
Bucket 8	0.000609	0.002771
Bucket 9	0.001256	0.002014
Bucket 10	0.006085	0.006828

4 Discussion

At first glance, the experimental results are less than satisfactory. However, METEOR and BLEU do not tell the whole story. At a high level, these metrics work by computing the level of agreement, e.g. unigram and bigram precision, between the sentence being evaluated and a pool of "correct" sentences (Lavie and Agarwal, 2007; Papineni et al., 2002). When the correct sentences agree strongly with each other, the evaluated sentence is heavily penalized for any departures from the correct sentence pool. This sort of penalization can occur even when the model-corrected sentence is a perfectly valid correction that just had the misfortune of choosing a different replacement word than the majority of the human workers. For example, one ESL sentence in our evaluation set reads, *progress of medical science helps human live longer.* All four of our models correct this to *progress of medical science helps people live longer*, but none of the workers correct to "people," instead opting for "humans." This issue is exacerbated by the fact that Mechanical Turk workers were instructed to change each ESL sentence as little as possible, which helps their consistency but hurts these particular models' evaluation scores.

With the exception of some mostly harmless but ultimately useless exchanges, e.g. changing "reduce mistakes" to "reduce errors," the models actually do fairly well when they correct ungrammatical words and phrases. As we alluded to in Section 1, all four model variations correct the sentence *to begin with, i'd rather not room with someone who is a entire stranger to me* from our development set to *to be-*

gin with, i'd rather not room with someone who is a complete stranger to me. But only 2 out of 5 human workers make this correction, 2 retain "entire," and 1 removes it altogether. As another example, all model variations correct *however, depending merely on luck is very dangerous* from our evaluation set to *however, depending solely on luck is very dangerous*. However, only 1 worker corrects "merely" to "solely," with the others either preferring to retain "merely" or leaving it out entirely.

None of this is to say that the models suffer only from an unfortunate difference in correction bias relative to the workers, or even that the models make good corrections a majority of the time. In fact, they make a range of false-positive corrections as well[4]. These seem to fall into three major categories: slight preferences for similar words that don't fit in the overall context of the sentence or change its meaning in an undesired way, e.g. changing "roommate" to "lodger" in *you and your roommate must devide [sic] the housework*, strong preferences for very common words in the local context that render the corrected sentence ungrammatical, e.g. changing "compose" to "take" in *first, during childhood years, we compose our personality*, and misinterpretations of ambiguous parts of speech that cause nouns to be replaced with verbs, etc., e.g. changing "circumstance" to "go" in . . . *that help you look abound your circumstance and find out* . . . .

Many of these issues can be blamed at least partially on the myopia of the language model, which, for example, vastly prefers "go and find" to "circumstance and find." However, they can also be attributed to the motivational intuition for Experiment 2, which states that we should avoid replacing observed words with common alternatives. While Table 3 does demonstrate that the models in Experiment 2 learn this preference to a degree for the highest frequency bucket, the proceeding buckets do not exhibit a smooth upwards slope analogous to the function being approximated. Indeed, the words in bucket 2 are preferred an order of magnitude more than those in bucket 1. This can be traced to the truncation policy of Algorithm 1, which selects only the highest frequency words from an over-sized set of random walk candidates. While it is unclear how to intelligently select a good candidate set of manageable size, a policy that butts heads with our intuition about which words we should be correcting is clearly not the right one.

The differences between the models themselves are somewhat more difficult to interpret. The 5 and 10 candidate variations of Experiment 1 and those of Experiment 2 correct 103, 108, 115, and 130 sentences out of 1017, respectively, and at least one model differs from the others on 123 of those sentences (they all agree on 42 sentences). These disagreements are of all types: sometimes only a single model corrects or vice versa, sometimes two models are pitted against the other two, and occasionally all four will choose a different word, but none of these inconsistencies seem to follow any sort of pattern, e.g. the two five candidate models agreeing more often than the other two or the like.

Interestingly, however, the models tend to be in agreement on the sentences that they correct the most effectively. We explore this more concretely in Table 4, in which we manually judge the quality of sentence corrections versus the agreement between models. Specifically, we judge a set of sentence corrections as *Good* if all of the corrections made between models improve sentence grammaticality, *Harmless* if the corrections do not significantly improve or reduce grammaticality, and *Bad* if at least one of the corrections is either ungrammatical or changes the sentence meaning. We note that *Bad* corrections for the most part do not take grammatical sentences and make them ungrammatical, only perturb them in some other erroneous fashion. Clearly, there is a strong correlation between corrected sentence quality and model agreement. We conclude from this observation that the models are all learning to correct the most unambiguously incorrect sentences in a consistent way, but where some deal of ambiguity remains, they are subject to random differences inherent in each's construction.

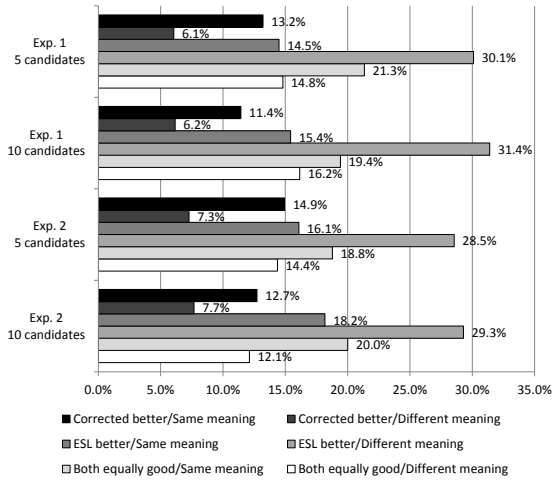
To round out our evaluation of correction quality, we presented the corrected sentences from all 4 model variations to human workers for judgment using the task detailed in Section 3.2.7. The results

[4]Although Type I errors are of course undesirable, Gamon et al. (2009) suggest that learners are able to effectively distinguish between good and bad corrections when presented with possible error locations and scored alternatives. Such an interactive system is beyond the scope of this paper but nonetheless feasible without significant model modification.

Table 4: Manual judgments of model-corrected sentence quality between experiments. If all models are in agreement, a sentence is marked as *Same*, and *Different* otherwise. We judge a set of sentence corrections as *Good* if all of the corrections made between models improve sentence grammaticality, *Harmless* if the corrections do not significantly improve or reduce grammaticality, and *Bad* if at least one of the corrections is either ungrammatical or changes the sentence meaning. Only corrected sentences are listed.

Model Agreement	Judgment	# of Sentences	% of Total
Same	Good	6	14.3%
	Harmless	11	26.2%
	Bad	25	59.5%
	Total	42	–
Different	Good	4	3.3%
	Harmless	34	27.6%
	Bad	85	69.1%
	Total	123	–

Figure 3: Human judgments of corrected sentences gathered using Mechanical Turk. The items listed in the legend are answers to the questions *Between the [original (ESL) and corrected] sentences, which is more correct?* / *Is the meaning of the corrected sentence significantly different from that of the ESL sentence?* See Section 3.2.7 for methodological details and Section 4 for results discussion.



of this effort are detailed in Figure 3. The workers are perhaps a bit more generous with their judgments than we are, but overall, they tend towards the same results that we do in our manual evaluation. Aside from the conclusions already presented, the worker judgments do expose one interesting finding: When the corrected sentence is judged to be at least as grammatical as the ESL sentence, it also tends to preserve the ESL sentence's meaning. However, when the ESL sentence is judged more correct, the meaning preservation trend is reversed. This observation leads us to believe that incorporating some measure of semantic distance into our random walk function f might prove effective.

5 Conclusion and Future Work

We have presented a novel noisy channel model for correcting a broad class of language learner production errors. Although our experimental results are mixed, we believe that our model constitutes an interesting and potentially very fruitful approach to ESL grammar correction. There are a number of opportunities for improvement available. Using a richer language model, such as a PCFG, would undoubtedly improve our results. Noting that ESL errors tend to occur in groups within sentences and are often interdependent, the addition of other noise models, such as those detailed in Park and Levy (2011), would further improve things by allowing the language model to consider a wider range of corrected contexts around each word. Our random walk model itself could also be improved by incorporating observed word frequency information or some notion of semantic difference between observed and unobserved words, or by learning separate parameters for different word classes. Somewhat counterintuitively, a structured reduction of dictionary richness could also yield better results by limiting the breadth of random walk candidates. Finally, a more intelligent heuristic for truncating large sets of random walk candidates would likely foster improvement.

Acknowledgments

We would like to thank three anonymous reviewers for their insightful comments and suggestions, and Markus Dreyer for providing us with his expectation semiring code. Additionally, we are grateful to the San Diego Supercomputer Center for allowing us access to DASH.

References

Cyril Allauzen, Michael Riley, Johan Schalkwyk, Wojciech Skut, and Mehryar Mohri. 2007. Openfst: a general and efficient weighted finite-state transducer library. In *Proceedings of the 12th international conference on Implementation and application of automata*, CIAA'07, pages 11–23, Berlin, Heidelberg. Springer-Verlag.

Harald R. Baayen, Richard Piepenbrock, and Leon Gulikers. 1995. *The CELEX Lexical Database. Release 2 (CD-ROM)*. Linguistic Data Consortium, University of Pennsylvania, Philadelphia, Pennsylvania.

Chris Brockett, William B. Dolan, and Michael Gamon. 2006. Correcting esl errors using phrasal smt techniques. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, ACL-44, pages 249–256, Stroudsburg, PA, USA. Association for Computational Linguistics.

Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: parameter estimation. *Comput. Linguist.*, 19:263–311, June.

Lou Burnard. 1995. *Users Reference Guide British National Corpus Version 1.0*. Oxford University Computing Services, UK.

Stanley F. Chen and Joshua Goodman. 1996. An empirical study of smoothing techniques for language modeling. In *Proceedings of the 34th annual meeting on Association for Computational Linguistics*, ACL '96, pages 310–318, Stroudsburg, PA, USA. Association for Computational Linguistics.

Arthur P. Dempster, Nan M. Laird, and Donald B. Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm (with discussion). *Journal of the Royal Statistical Society B*, 39:1–38.

Alain Désilets and Matthieu Hermet. 2009. Using automatic roundtrip translation to repair general errors in second language writing. pages 198–206. MT Summit XII.

Markus Dreyer, Jason R. Smith, and Jason Eisner. 2008. Latent-variable modeling of string transductions with finite-state methods. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 1080–1089, Stroudsburg, PA, USA. Association for Computational Linguistics.

Jason Eisner. 2002. Parameter estimation for probabilistic finite-state transducers. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 1–8, Stroudsburg, PA, USA. Association for Computational Linguistics.

Michael Gamon, Claudia Leacock, Chris Brockett, William B Dolan, Jianfeng Gao, Dmitriy Belenko, and Alexandre Klementiev. 2009. Using statistical techniques and web search to correct esl errors. *CALICO Journal*, 26:491–511.

Stanley Kok and Chris Brockett. 2010. Hitting the right paraphrases in good time. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 145–153, Stroudsburg, PA, USA. Association for Computational Linguistics.

Alon Lavie and Abhaya Agarwal. 2007. Meteor: an automatic metric for mt evaluation with high levels of correlation with human judgments. In *Proceedings of the Second Workshop on Statistical Machine Translation*, StatMT '07, pages 228–231, Stroudsburg, PA, USA. Association for Computational Linguistics.

John Lee and Stephanie Seneff. 2006. Automatic grammar correction for second-language learners. ICSLP.

Mehryar Mohri. 2001. Generic ϵ-removal algorithm for weighted automata. In Shen Yu and Andrei Paun, editors, *Implementation and Application of Automata*, volume 2088 of *Lecture Notes in Computer Science*, pages 230–242. Springer Berlin / Heidelberg.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, Stroudsburg, PA, USA. Association for Computational Linguistics.

Y. Albert Park and Roger Levy. 2011. Automated whole sentence grammar correction using a noisy channel model. ACL '11. Association for Computational Linguistics. In Press.

C. E. Shannon. 1948. A mathematical theory of communication. *Bell Systems Technical Journal*, 27:623–656.