Mining Wikipedia's Article Revision History for Training Computational Linguistics Algorithms

Rani Nelken and **Elif Yamangil**
School of Engineering and Applied Sciences
Harvard University
Cambridge, MA 02138, USA
{nelken,elif}@eecs.harvard.edu

Abstract

We present a novel paradigm for obtaining large amounts of training data for computational linguistics tasks by mining Wikipedia's article revision history. By comparing adjacent versions of the same article, we extract voluminous training data for tasks for which data is usually scarce or costly to obtain. We illustrate this paradigm by applying it to three separate text processing tasks at various levels of linguistic granularity. We first apply this approach to the collection of textual errors and their correction, focusing on the specific type of lexical errors known as "eggcorns". Second, moving up to the sentential level, we show how to mine Wikipedia revisions for training sentence compression algorithms. By dramatically increasing the size of the available training data, we are able to create more discerning lexicalized models, providing improved compression results. Finally, moving up to the document level, we present some preliminary ideas on how to use the Wikipedia data to bootstrap text summarization systems. We propose to use a sentence's persistence throughout a document's evolution as an indicator of its fitness as part of an extractive summary.

Introduction

Much recent progress in natural language processing stems from successfully leveraging large-scale document corpora as a source of training data. Text documents are almost invariably found in fixed final form, a form which hides an often rich history of the documents' evolution from inception as a first draft to final published form. If we could somehow gain access to this information for a large document corpus, we could learn invaluable information from it.

Fortunately, Wikipedia provides just such a resource. Through Wikipedia's collaborative editing process, articles are iteratively amended and refined by multiple Web users. Wikipedia offers periodic snapshots of all of this historical data for its more than 7 million articles, thus providing a virtual paper trail of this collaborative editing process. It would not be an exaggeration to state that in the entire history of writing there has never been such a comprehensive resource containing the full history of so many documents.

We present a new paradigm for leveraging this data for training language processing algorithms. By comparing different versions of the same document, and repeating the process over a large collection of documents, we propose to collect users' editorial choices. We illustrate this process at different levels of linguistic granularity ranging from the level of the single word, through the sentence, to the document, using the data for three different tasks.

First, consider the problem of automated text correction. Text is often fraught with errors in spelling, style, and grammar, requiring subsequent correction and editing. Collecting common errors and their corrections is of obvious practical interest for text-correction algorithms as well as theoretical linguistic studies. Although modern word processors provide support for automated and semi-automated text correction, including context-sensitive spelling correction and grammar-checking, even the most sophisticated tools still fall short of catching all errors.[1] Given a supervised collection of typical errors, text correction algorithms can be accurately trained to identify and correct errors. For instance, Carlson, Rosen, and Roth (2001) presented a Winnow-based algorithm achieving accuracy levels at the 99% range for 265 lexical confusion sets, but how can we effectively collect such confusion sets automatically?

We propose that such errors and their corrections can be automatically harvested from Wikipedia article revisions. Since Wikipedia articles are viewed by many pairs of eyes, errors inadvertently added by one user are likely to be identified and corrected by subsequent users. By comparing temporally adjacent versions of the same article, we can metaphorically peer over users' shoulders just as they are making corrections. As a proof of principle, we illustrate this approach here on a very specific form of lexical error, known as "Eggcorns", a lexical error that retains both phonetic similarity and some level of semantic coherence. We show how such errors, which are of great interest to linguists—in addition to their practical interest for context-sensitive spelling correction—can be mined automatically from Wikipedia's revision history.

As a second application, we move up from the lexical level to the sentence level, presenting a new algorithm for sentence compression, the problem of shortening sentences

[1] See http://faculty.washington.edu/sandeep/check/ for a critique of a popular commercial text editor's correction capabilities.

by dropping words in a way that preserves the most pertinent information and does not harm grammaticality. Such shortening is useful for instance for subtitle or caption generation or as part of a larger machine translation or summarization system. This problem has received much attention in the literature, but has suffered from a severe dearth of training data. We show how we can obtain training data up to 3 orders of magnitude larger than ever used for this task. Using only a fraction of this data, we train a novel statistical noisy channel model for sentence compression, showing improved compression rates and grammaticality with only a slight decrease in the importance of the retained information.

Finally, moving to the document level, we describe some early experiments with a new approach to taking advantage of Wikipedia revision data for training text summarization systems. Inspired by biological evolution, we hypothesize that the temporal persistence of a sentence throughout the revision history is a good indicator of its importance. We present some preliminary experiments lending credence to the idea that this approach can be used to help train text summarization systems using a corpus of unprecedented size.

Throughout the presentation of these three different tasks, we maintain the same theme of using Wikipedia's revision history to provide significant amounts of training data, which was previously either extremely sparse or very costly to acquire manually.

All our experiments were performed on the July 2006 full history version of the English Wikipedia, consisting at the time of 1.4 million articles. To efficiently process a dataset of such size (hundreds of GBs), we split it into multiple smaller chunks, and distribute all the processing among multiple processors. Note that at present Wikipedia includes almost 2.3 million English articles, significantly increasing the available data.

Harvesting Eggcorns

The term "eggcorn" was coined by Geoff Pullum on the popular "Language Log" Blog[2] for a particular type of error in English language usage. This error occurs when an expression or part of it is substituted by a homophone such that the resulting expression still makes sense semantically, even while deviating from common usage. The word "eggcorn" is itself an eggcorn for "acorn". The idea is that the error is not merely phonetic, but there are also semantic reasons behind the confusion, however misguided it is. For instance, an acorn and an egg share the same shape, and a grain of corn and an acorn are both seeds. Such usage errors are a source of interest (as well as undeniable amusement) for linguists, who may use them to learn interesting variations in common usage and their underlying semantics.

We denote eggcorns here as ordered pairs of the form ⟨*incorrect, correct⟩, using the linguists' star notation for the incorrect form. As a canonical running example we use the pair ⟨***full**proof, **fool**proof⟩. We are interested in automatically identifying occurrences of such eggcorns from Wikipedia revisions. A useful reference database of eggcorns which we use is the Eggcorn Database,[3] which documents 596 eggcorns. For simplicity, we focus here only on eggcorns consisting of a pair of single words (thus ignoring eggcorns of the form ⟨*a posable, opposable⟩ (as in "*a posable thumb"). We also exclude any eggcorns where one of the words is a stop-word, e.g., ⟨*and, ad⟩ (as in "*and hoc"), leading to a total of 348 reference eggcorns.[4]

Eggcorns are defined by a combination of semantic and phonetic similarity. The notion of semantic similarity underlying eggcorns is extremely nebulous, and non-uniform. For instance, we can detect and retrospectively rationalize some notions of semantic similarity in eggcorn pairs such as ⟨*crutch, crux⟩ and ⟨*isle, aisle⟩, or indeed in ⟨*eggcorn, acorn⟩, but it is much more difficult to formulate a general criterion that captures such similarity, much less automate it. For instance, it is clear that this type of similarity goes well beyond standard thesaural relations, and is unlikely to be easily amenable to co-occurrence based measures.

Phonetic similarity, on the other hand, is much more straightforward. We experimented with two measures:

- The classic Soundex algorithm, developed by Odell and Russell and patented as early as 1918 (Hall and Dowling 1980), which is based on mapping a word's consonants to one of 6 classes. Each word is then mapped to a concise 4-character code which consists of the first letter of the word and the first three distinct class indices of its letters. Two words are considered similar if they are mapped to the same code.
- Editex (Zobel and Dart 1996), which uses a similar though distinct method of classifying the letters into classes. Given two words, it then applies an edit-distance computation on the full encoding. Two words are considered similar if the edit-distance is below some threshold.

We tested these algorithms by applying them to the reference eggcorns, with the results in Table 1. Best results for Editex were achieved with an edit-distance threshold of 0.5. We therefore also used it for finding new eggcorns.

Table 1: Phonetic similarity of reference eggcorns

Algorithm	Precision	Recall
Soundex	0.95	0.64
Editex (threshold = 0.5)	0.90	0.96

Finding eggcorns

We are interested in all corrections of the form $\langle *w_1, w_2\rangle$, where the words are phonetically similar, are not morphologically related, and are not synonyms.

It is thus insufficient to search just for occurrences of w_1 or of w_2; we need to find genuine instances where w_1 is changed to w_2. We are interested in finding instances of

[2] http://itre.cis.upenn.edu/~myl/languagelog/

[3] http://eggcorns.lascribe.net/

[4] In addition to the officially sanctioned eggcorns, the user forums on the Eggcorn Database site include hundreds of additional user-contributed eggcorn candidates. For simplicity, we did not look for these.

the reference eggcorns as well as new, previously unreported eggcorns. To limit the search space, we first indexed all the articles that contain an occurrence of w_2, where w_2 is the correct member of one of the reference eggcorns. Over these articles we then searched for all cases where some word, w_1, is modified to a phonetically similar word, w_2. This approach ensures that we will traverse all the articles containing a potential occurrence of a reference eggcorn, and has the added potential of finding additional eggcorns.

We split each article into its set of revisions, $(r_1, r_2, \ldots, r_N)$ in chronological order. We find all pairs of adjacent revisions (r_n, r_{n+1}) where r_{n+1} included w_2, but r_n did not. Where did w_2 come from? Either it was typed correctly by the editor of r_{n+1}, or it already appeared in a possible incorrect form, w_1, in r_n, and was modified to w_2 in r_{n+1}, which is precisely the case we are interested in.

We seek the word w_1 that was w_2's predecessor in r_n by performing a double edit-distance operation. Using lines as a simple proxy for sentences, we split r_n and r_{n+1} into lines, and run an edit-distance operation, treating each line as an atomic token. We then examine where the line containing w_2 in r_{n+1} came from. If this line is a replacement of another line in r_n, then we run a second edit-distance on these two lines, this time at the resolution of words, to find where w_2 came from. If it is a replacement of a word, w_1, we check whether w_1 is phonetically similar to w_2. We further use WordNet (Fellbaum 1998) to filter out any pairs that share a morphological stem, or are potential synonyms.[5] This WordNet filter eliminates two reference eggcorns, ⟨*font, fount⟩, and ⟨*flare, flair⟩. In both cases, the words are synonyms under a dispreferred disambiguation. We output the resulting pairs $\langle *w_1, w_2 \rangle$ as eggcorn candidates.

We successfully found 31% of the reference eggcorns, such as ⟨*dandruff, dander⟩, ⟨*curtsey, courtesy⟩, and ⟨*isle, aisle⟩. Of course, there is no guarantee that the remaining eggcorns have ever occurred in Wikipedia. In addition, the procedure was able to identify many new and interesting eggcorns not previously listed in the Eggcorn Database. We list some examples in Table 2. Not surprisingly, less common words are more prone to error.

Table 2: Sampling of new eggcorns identified from Wikipedia

⟨*funder, founder⟩	⟨*rectify, ratify⟩
⟨*heaven, haven⟩	⟨*absorb, adsorb⟩
⟨*acerbate, exacerbate⟩	⟨*arrogate, abrogate⟩
⟨*birth, berth⟩	⟨*citing, sighting⟩
⟨*siege, seize⟩	⟨*ripe, rife⟩
⟨*rigid, rugged⟩	⟨*reverse, revert⟩
⟨*assume, resume⟩	⟨*restrain, refrain⟩

This procedure is geared towards high recall, and thus unsurprisingly has poor precision. False positives include typos ("bight" instead of "blight") pairs of words that are not in fact phonetically confusable (e.g., "ability" and "agility"), or not semantically related (e.g., "bacon" and "beacon"), or profanity. Future work is needed to filter these out effectively.

These results provide an encouraging proof of principle for mining such corrections automatically from Wikipedia revisions. We are currently working on a more comprehensive context-sensitive text-correction system trained on more general Wikipedia revision corrections.

Sentence compression

Moving from the lexical to the sentential level, we next explore sentence compression. We summarize the main contribution here, with fuller details presented in (Yamangil and Nelken 2008).

With the increasing success of machine translation in recent years, several researchers have suggested transferring similar methods for monolingual text rewriting tasks. In particular, Knight and Marcu (2000) (KM) applied a noisy channel model to the task of *sentence compression* — dropping words from an individual sentence while retaining its important information, and without sacrificing its grammaticality.

A well-recognized problem of sentence compression is data sparseness. While bilingual parallel corpora are relatively easy to obtain, collections of sentence compressions are quite rare. Indeed, most work on sentence compression has used the Ziff-Davis corpus (Knight and Marcu 2000), which consists of a mere 1067 sentence pairs. While data sparseness is a common problem of many computational linguistics tasks, the dearth of sentence compression data is a well recognized problem (Turner and Charniak 2005).

In recent years, there has been a growing interest in the related and more general tasks of sentence paraphrasing (Dolan, Quirk, and Brockett 2004), and textual entailment (Dagan, Glickman, and Magnini 2005), together with concentrated efforts to create common datasets for these problems. Theoretically, we could use this data for sentence compression as well, by filtering just those sentence pairs that are true compressions, as opposed to more general paraphrases. Unfortunately, however, these datasets are still only on the order of hundreds or thousands of sentences, only a fraction of which would be directly useful for compression, and hence they do not solve the sparseness problem.

Mining Wikipedia revisions from 250,000 articles, we have extracted over 380,000 full/compressed sentence pairs, 2 orders of magnitude more than the number of pairs in the Ziff-Davis corpus. Since Wikipedia currently has almost 2.3 million English articles and is constantly expanding, we can expect an increase of another order of magnitude. We thus can afford to be extremely selective of the sentence pairs we use. We make the simplifying assumption that *all* such sentence pairs also retain the core meaning, and are therefore valid training data for our purposes. This assumption is of course patently naïve, as there are many cases in which such revisions reverse sentence meaning, add or drop essential information, are part of a flame war, etc. Classifying these edits is an interesting task which we relegate to future work.

As with the eggcorns, we first extract all revisions, for each article. Here splitting into lines is no longer sufficient,

[5]We thank one of the reviewers for suggesting the WordNet filter.

so we split each revision into its sentences using a rule-based sentence splitter. [6]

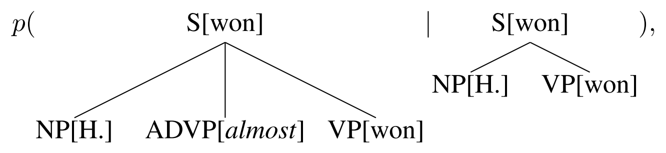
For each article, we run an edit-distance comparison between each temporally adjacent pair of revisions, treating each sentence as an atomic token. We look for all replacement operations of one sentence by another, and check whether one sentence is a compression of the other. We filter out ungrammatical sentences during preprocessing in which we run Collins' parser (1997), filtering out any sentences that score below a certain threshold.

We wish to apply KM's generative noisy channel model for the problem. Under this model, sentences start their life in short form, s, are ranked by a source language model, $p(s)$, and then probabilistically expanded to form the long sentence, $p(l|s)$. During decoding, given a long sentence, we seek the most likely short sentence that could have generated it. Using Bayes' rule, this is equivalent to seeking the short sentence s that maximizes $p(s) \cdot p(l|s)$.

This huge increase in training data enables not only a quantitive improvement in existing models, but a qualitative improvement as well. Whereas KM's original model was purely grammatical, we take advantage of the huge increase in data to lexicalize the model. Thus, sentence compression decisions can be made not only on the basis of grammatical information, but also taking into account the values of the lexical items. To illustrate why this is useful, consider the following sentence pair:

1. Hillary *barely* won the primaries.
2. Hillary *almost* won the primaries.

The validity of dropping the adverbial here clearly depends on the lexical value of the adverb. It is much more acceptable to drop the adverb in Sentence 1, since dropping it in Sentence 2 reverses the meaning. We learn probabilities of the form:



where we percolate the lexical head for each phrase up the tree. Our model makes compression decisions based on lexical dependencies between the compressed and retained parts of the parse tree.

We use Witten-Bell discounting (1991) to gradually back off from fully lexicalized probabilities to the purely grammatical probabilities, in cases where there is not enough lexical information. In addition to the lexicalized channel model, we also use a lexicalized probabilistic syntax-based source model, which we train from the parser's output on the short sentences of each sentence pair. Finally, decoding is done using the statistical sentence generator of Langkilde (2000), from which we extract the best scoring compression.

[6]We used a sentence splitter by Paul Clough, from `http://ir.shef.ac.uk/cloughie/software.html`

Evaluation

We evaluated our system using the same method as KM, using the same 32 sentences taken from the Ziff-Davis corpus.[7] We solicited judgments of *importance* (the value of the retained information), and *grammaticality* for our compression, the KM results, and human compressions from 8 native English speaking judges on a scale of 1 (worst) to 5 (best). Mean and standard deviation are shown in Table 3. Our method achieves an increase in compression rate as well as grammaticality over KM's method, with a slight decrease in importance.

Text summarization

As a final illustration of this new paradigm, we now move on to the document level, and in particular focus on text summarization, a task which has received much attention in the literature. The dominant approach to summarization is sentence extraction, in which a summary is created by extracting sentences from the full document, optimizing sentence salience and coherence, while avoiding redundancy.

We suggest that information in the revision history may be useful for summarizing Wikipedia articles. There are two motivations for this approach. First, there is independent interest in summarizing Wikipedia articles, due to their ubiquitous use. Although many Wikipedia articles include short abstracts, these abstracts have a fixed short length, and it would be useful to create summaries of flexible length.

More interestingly, if features extracted from the revision history are indeed significant indicators of importance, we could use these features to bootstrap further summarization research as follows. First, we would train a supervised or semi-supervised Wikipedia summarization system based in large part on the revision history. Second, we would apply the trained summarizer to Wikipedia articles, obtaining a huge collection of pairs of articles and their automatically generated summaries. Finally, we would use these pairs as training data for further summarization systems, which would no longer require the revision history.

What can we learn from the revision history about the importance of sentences for summaries? Following a biological intuition, we hypothesize that a sentence's persistence in an article is a good measure of its "fitness", i.e., the importance of it being included in the article. For all sentences in the most recent version of the article, we wish to track their lifespan from the moment they first appear in the document, through whatever changes they undergo, until the final version. Our hypothesis is that sentences that have persisted longer are more likely to be important than sentences that are only more recent additions, and are thus better candidates for extractive summaries.

Since sentences evolve, we have to be careful in tracking sentences. Viégas, Wattenberg, and Dave (2004) presented striking visualizations of sentence-level history of Wikipedia articles, using a notion of sentence identity up to the character level. This notion is too strict for our purposes (and arguable also for visualization). We therefore define a new notion of weak sentence identity up to an edit-distance

[7]We thank KM for sharing their dataset with us.

	KM	Our model	Humans
Compression	72.91%	67.38%	53.33%
Grammaticality	4.02±1.03	4.31±0.78	4.78±0.17
Importance	3.86±1.09	3.65±1.07	3.90±0.58

Table 3: Compression evaluation results

threshold, computed pairwise between the revisions. [8] For each sentence in the final version, we define its persistence at revision r_n to be the percentage of revisions in which it has survived.

$$\text{persistence}_n(s) = \frac{\#\text{revisions until } r_n \text{ including} s}{n}$$

We define the (final) persistence of a sentence as its persistence in the final revision. We discount large-scale spam revisions such as wholesale deletions. We also maintain the identity of sentences that are deleted and disappear from up to 50 revisions only to reappear in a subsequent revision.

To illustrate the effect of sentence identity on persistence, the average and standard deviation for persistence using both types of identity for a particular Wikipedia article, "World of Warcraft" (WoW), are as follows: 0.470 ± 0.365 (using weak identity), 0.162 ± 0.210 (using strict identity). As expected, the average using weak identity is higher, indicating that we can track sentences along a larger number of revisions, discounting small changes. Moreover, the standard deviation is higher, indicating more diversity in persistence, and thus a stronger differentiation between sentences. The Pearson correlation coefficient between the two persistence score vectors is 0.418, indicating a large qualitative difference.

To gauge the efficacy of this approach, we scored the persistence of sentences of two Wikipedia articles, "WoW" and "William Shakespeare" (WS) and plotted them shown in the heat-maps of Figures 1 and 2. In these figures, the (n, k) pixel corresponds to $\text{persistence}_n(s_k)$, where s_k is the k'th sentence of the final version. The color indicates the level of persistence, on a scale from blue (0) to red (1). Each sentence contributes a horizontal stripe of growing persistence, with highest persistence sentences tending towards the red end of the spectrum. Discontinuities indicate that the sentence was dropped from a small number of revisions only to return later on.

Examining the preserved sentences, we see that they often correspond to the first sentences in each section. We have marked these correspondences on the Y-axis of the figures. Since the lead sentences of a section are often the most important, this graph lends credence to our hypothesis that sentence persistence correlates with importance. Of course, much more extensive evaluation is required to validate this hypothesis. In addition, we see that structural Wikipedia markup is often maintained as well. This is illustrated in Figure 2 towards the bottom of the document. This document, WS, includes a structured section containing a collection of links, and these, not surprisingly well-preserved.

Obviously, our preliminary observation of the correlation between persistence and importance in no way constitutes a full summarization system. A true summarization system would not use this feature in isolation, but as part of a larger collection of features. Indeed, we plan to examine additional features of the revision history, such as user confidence, the number and extent of edits that a sentence underwent, number of revisions two sentences were adjacent, etc. The hope is to use such automatically extracted features to replace or augment human annotation in a method such as (Kupiec, Pederson, and Chen 1995). If successful, this approach can be applied to millions of Wikipedia articles, which could then be used as training data for further summarization algorithms.

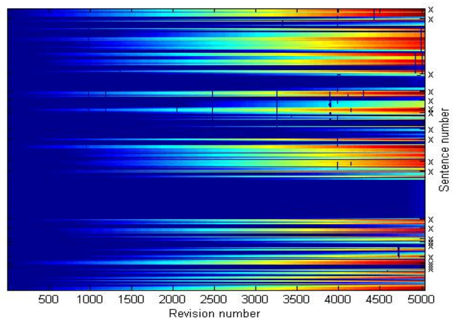


Figure 1: Persistence heat map for WoW

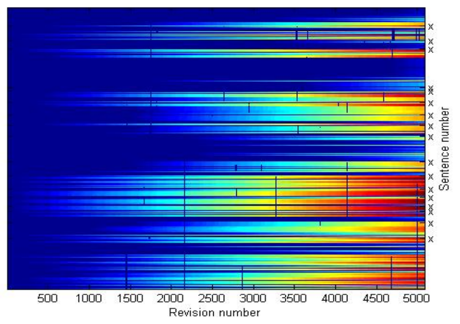


Figure 2: Persistence heat map for WS

[8] We chose the edit-distance threshold manually. In future work, we will train this threshold using cross-validation on a small hand-labeled training set. We thank one of the reviewers for this suggestion.

Conclusion

We have presented a new paradigm for obtaining large scale training data for training natural language processing algorithms from Wikipedia's article revision history, illustrating the approach on three specific tasks. Additional applications of this method abound at all levels of linguistic granularity. For instance, at the lexical level, our focus on eggcorns was more illustrative than exhaustive. For a full text correction application, one could mine the revisions for other types of lexical or phrasal replacements. One of the challenges of such an endeavor is filtering out the content-based replacements that would be meaningful only in the context of a particular article from the more general corrections that can be applied across the board. Hopefully, since content-related corrections would be specific to a particular document or domain context, the more general corrections would recur much more frequently in the corpus.

In addition to correction, one particularly interesting form of lexical substitution is the use of anaphoric pronouns. The revision history contains many instances of a noun phrase being replaced by an anaphoric pronoun or vice versa. If we make the plausible assumption that such replacements retain sentence meaning, we can use this data to train anaphora resolution systems using millions of articles.

Moving up to the sentence level, much work remains on extending the sentence compression method we have described. In particular, we are interested in learning better predictors of importance by classifying different types of sentence compressions/expansions appearing in the Wikipedia data, and moving from a generative to a discriminative model. Furthermore, we plan to use the data to learn other types of sentence rewriting such as more flexible paraphrases and grammatical reordering.

Finally, at the document level, our initial experiments on summarization are an encouraging first step of a fuller study of how to take advantage of Wikipedia's article revisions for training summarization algorithms.

Acknowledgments

This work was partially supported by a Google research award. We thank Stuart Shieber and the anonymous reviewers for their suggestions.

References

Carlson, A. J.; Rosen, J.; and Roth, D. 2001. Scaling up context sensitive text correction. In *IAAI2001*, 45–50.

Collins, M. 1997. Three generative, lexicalized models for statistical parsing. In Cohen, P. R., and Wahlster, W., eds., *Proceedings of the Thirty-Fifth Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics*, 16–23. Somerset, New Jersey: Association for Computational Linguistics.

Dagan, I.; Glickman, O.; and Magnini, B. 2005. The PASCAL recognising textual entailment challenge. In Candela, J. Q.; Dagan, I.; Magnini, B.; and d'Alché Buc, F., eds., *MLCW*, volume 3944 of *Lecture Notes in Computer Science*, 177–190. Springer.

Dolan, B.; Quirk, C.; and Brockett, C. 2004. Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING-2004)*.

Fellbaum, C. 1998. *WordNet: An Electronic Lexical Database*. MIT Press.

Hall, P. A. V., and Dowling, G. R. 1980. Approximate string matching. *ACM Comput. Surv.* 12(4):381–402.

Knight, K., and Marcu, D. 2000. Statistics-based summarization - step one: Sentence compression. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*, 703–710. AAAI Press / The MIT Press.

Kupiec, J.; Pederson, J.; and Chen, F. 1995. A Trainable Document Summarizer. In *SIGIR 1995*, 68–73. Washington, WA, USA: ACM.

Langkilde, I. 2000. Forest-based statistical sentence generation. In *Proceedings of the first conference on North American chapter of the Association for Computational Linguistics*, 170–177. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Turner, J., and Charniak, E. 2005. Supervised and unsupervised learning for sentence compression. In *ACL '05: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, 290–297. Morristown, NJ, USA: Association for Computational Linguistics.

Viégas, F. B.; Wattenberg, M.; and Dave, K. 2004. Studying cooperation and conflict between authors with *history flow* visualizations. In Dykstra-Erickson, E., and Tscheligi, M., eds., *CHI*, 575–582. ACM.

Witten, I., and Bell, T. 1991. The zero-frequency problem: Estimating the probabilities of novel events in adaptive text compression. *IEEE Transactions on Information Theory* 37(4).

Yamangil, E., and Nelken, R. 2008. Mining wikipedia revision histories for improving sentence compression. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*.

Zobel, J., and Dart, P. W. 1996. Phonetic string matching: Lessons from information retrieval. In Frei, H.-P.; Harman, D.; Schäble, P.; and Wilkinson, R., eds., *Proceedings of the 19th International Conference on Research and Development in Information Retrieval*, 166–172. Zurich, Switzerland: ACM Press.