From: IAAI-01 Proceedings. Copyright © 2001, AAAI (www.aaai.org). All rights reserved.

Scaling Up Context-Sensitive Text Correction

Andrew J. Carlson **Jeffrey Rosen** **Dan Roth**

Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, IL 61801
[ajcarlso, jrosen, danr@uiuc.edu]

Abstract

The main challenge in an effort to build a realistic system with context-sensitive inference capabilities, beyond accuracy, is scalability. This paper studies this problem in the context of a learning-based approach to context sensitive text correction – the task of fixing spelling errors that result in valid words, such as substituting *to* for *too*, *casual* for *causal*, and so on. Research papers on this problem have developed algorithms that can achieve fairly high accuracy, in many cases over 90%. However, this level of performance is not sufficient for a large coverage practical system since it implies a low sentence level performance.

We examine and offer solutions to several issues relating to scaling up a context sensitive text correction system. In particular, we suggest methods to reduce the memory requirements while maintaining a high level of performance and show that this can still allow the system to adapt to new domains. Most important, we show how to significantly increase the coverage of the system to realistic levels, while providing a very high level of performance, at the 99% level.

Introduction

Virtually all current day software systems that perform text processing provide some spell checking facility. Word processors, email and news readers, and even operating systems provide tools to verify that the text contains valid words. When an invalid word is discovered some form of distance measure is used to select candidate correct words from a dictionary. The shortcoming of all these spell checkers is that they fail to detect errors that result in a valid word, as in *I'd like a peace of cake*, where *peace* was typed when *piece* was intended, or *I took a walk it the park*, where *it* was typed instead of *in*, etc.

An earlier study (Kukich 1992) showed that errors that result in valid words account for anywhere from 25% to over 50% of observed spelling errors. Today, as our reliance on text processing tools increases while fewer resources are spent on editing published text - the Internet revolution has resulted in additional pressure to shorten the time from writing to publishing - this could be a significant undercount.

However, identifying and correcting mistakes that result in valid words requires awareness of the *context* in which different words, such as *piece* and *peace*, tend to occur. The problem of characterizing the linguistic context in which even a single word tends to occur is a difficult problem and a large scale one; it might depend on particular words near the target word, the pattern of parts of speech around the target word and so on. A "knowledge engineering" approach to this problem, therefore, is unlikely to succeed. Indeed, in recent years, machine learning techniques have begun to be applied to this problem and several of them were shown to do quite well (Golding & Roth 1996; Golding & Roth 1999; Mangu & Brill 1997).

Existing work along these lines has focused on developing learning methods that are appropriate for this problem and thus concentrated on a relatively small number of words. Restricting the problem this way (to 20–40 words, depending on the study) also allowed researchers to keep the experiments manageable, given the large scale of the problem.

However, in order to be useful as a practical tool, systems addressing this problem need to be able to offer wide word coverage with reasonable performance and resource requirements. This work offers solutions to several issues relating to the *scaling up* of these systems.

First, it is clear that almost every word in English could be mistaken for some other valid word, and therefore a practical system needs to have a coverage of thousands of words. Our first step in the current work is therefore to increase the coverage of our system to roughly five-hundred words. The approach taken by most of the existing work has been to form *confusion sets* and treat the problem as a disambiguation task over the members of a confusion set. Confusion sets consist of words that are likely to be misused in place of one another. In this paper we continue with this approach. We seek to handle a number of confusion sets closer to the scale of the real problem, but without having to fine-tune parameters for each set.

Second, given that the number of features that might be required to characterize the context of a word is very large, scaling up to realistic coverage might introduce resource problems - memory and evaluation time. We suggest a way to avoid that and show its minimal effect on the performance. A related issue involved in a practical approach to context sensitive text correction is that different genres of text might have different characteristics and might use different vocabulary; this could require different characteriza-

Copyright © 2001, American Association for Artificial Intelligence (www.aaai.org). All rights reserved.

tion of the context. We show that our approach can adapt to new texts quickly and reliably.

Finally, the most important issue is performance. Research papers on context sensitive text correction have shown different algorithms that can achieve fairly high accuracy, in many cases over 90%. However, this level of performance is not sufficient for a large coverage practical system. Performing at the 90% level in a wide coverage system means that the system will make, on average, one mistake per sentence, and this would be unacceptable for most users. We suggest a way to significantly increase the performance of a wide coverage system by automatically reducing the willingness of the system to alert the user for mistakes in which it is less confident. This solution relies on the ability of the algorithm to reliably assess its confidence in the prediction, and, as we show, our approach can do that, yielding an average performance of over 99% over a large corpus, with prediction willingness of 85%.

Our algorithmic approach builds on one of the most successful approaches studied for this problem (Golding & Roth 1999), based on the SNoW learning architecture (Carlson *et al.* 1999; Roth 1998). We briefly describe how SNoW is used here, discuss some methodological issues and then interleave the scaling up discussion with the experiments performed to exhibit the performance of the system.

Context-Sensitive Text Correction

Given a body of text, possibly like this paper, we would like to scan it and locate errors resulting from the improper usage of *real words*. This task has typically been referred to as *context-sensitive spelling correction* in earlier research, but here we refer to it as text correction rather than spelling since the techniques are not limited to simple single word substitutions. Context-Sensitive Text Correction is the task of fixing spelling errors that happen to result in valid words, such as substituting *to* for *too*, *casual* for *causal* or simple word usage errors like in *"There could be any amount of reasons he didn't show up."*, where *amount* was used instead of *number*. Our definition of the task includes correcting not only "classic" types of spelling mistakes, such as homophone errors, (e.g., *peace* and *piece*) and typographic errors, as in *"I'll be ready in five minuets."* (where *minuets* was typed when *minutes* was intended), or when *from* is replaced by *form*. We can also fix mistakes that are more commonly regarded as grammatical errors (e.g., "among" and "between"), incorrect forms of pronouns, as in *"I had a great time with his."*, where *his* was typed instead of *him* or errors that cross word boundaries (e.g., *maybe* and *may be*).

Problem Formulation

We cast context-sensitive text correction as a disambiguation task (Roth 1998). Given an input sentence and a distinguished word sequence (usually of size 1) - which we call the *target* - within the sentence, we wish to predict whether the target is correct, or whether it should be replaced by some other word sequence. The ambiguity among words (or word sequences) is modeled by *confusion sets*. A confusion set $C = \{W_1, \ldots, W_n\}$ means that each word W_i in the set is ambiguous with each other word. Thus if $C = \{hear, here\}$, then when we see an occurrence of either *hear* or *here* in the target document, we take it to be ambiguous between *hear* and *here*; the task is to decide from the context which one was actually intended.

Applying SNoW to Context-Sensitive Text Correction

Our study makes use of one of the more successful learning approaches tried on the problem of context sensitive text correction (Golding & Roth 1999). SNoW (Roth 1998; Carlson *et al.* 1999) is a multi-class classifier that is specifically tailored for large scale learning tasks. The SNoW learning architecture learns a sparse network of linear functions, in which the targets (elements in confusion sets, in this case) are represented as linear functions over a common feature space. Several update rules can be used within SNoW. The most successful update rule, and the only one used here, is a variant of Littlestone's Winnow update rule (Littlestone 1988), a multiplicative update rule that we tailored to the situation in which the set of input features is not known a priori. SNoW has already been used successfully for a variety of tasks in natural language and visual processing (Golding & Roth 1999; Roth, Yang, & Ahuja 2000; Punyakanok & Roth 2001). We refer the reader to these for a detailed description of SNoW; here we briefly describe how it is applied to context-sensitive text correction and the modifications made relative to (Golding & Roth 1999).

When SNoW is applied to context-sensitive text correction a target node is allocated to each word sequence that is a member of a confusion set. Thus, each word sequence is learned as a function of the context in which it correctly appears. A SNoW unit corresponds to a confusion set; in training, elements belonging to a unit are trained together in the sense that they compete with each other - given a confusion set element, it is viewed as a positive example to its corresponding target and as negative to the targets in its unit. At evaluation time, an element of one of the confusion sets is identified in the text, and the competition is between the targets' corresponding elements in the confusion set.

In principle, a more general approach could use a single confusion set containing all words. However, this is not practical for a general text correction system. If we ignore the confusion sets and present all examples to all targets for training, and then have all targets compete at evaluation time, we see great decreases in both computational efficiency and performance (Even-Zohar & Roth 2000).

The key difference in the architecture used here from the one used in (Golding & Roth 1999) is the fact that we use only a single layer architecture without the notion of the "clouds" used there. While, as shown there, the use of clouds improves the performance somewhat, the simplified architecture used here greatly reduces the learning time and memory requirement. We get the performance level back up in other ways, using a larger training corpus and, mainly, using the confidence level enhancements described later. In particular, that implies that we explicitly use the activation level output by SNoW, rather than only the prediction.

Experimental Methodology

This work makes used of the concept of *confusion sets* and treats the problem as a task of disambiguating the correct set member. The confusion sets acquired for this work were generated automatically by using simple edit distance in both the character space and phoneme space. We later pruned and edited the list manually. Overall, the experiments used a set of 265 different confusion sets (previous works have used between 10 to 21). 244 of the confusion sets were of size 2, 20 were of size 3, and 1 was of size 4.

The experiments were performed on data from the TDT2 English corpus that is available via the Penn treebank (Marcus, Santorini, & Marcinkiewicz 1993). The corpus includes text taken from six English news sources, which aids in the generality of our system. It includes about 1,000,000 English sentences, providing a good amount of data for most of the confusion sets we are interested in. Each experiment was run using five-fold cross-validation, where in each case 80% of the corpus was used for training and the remaining 20% was used for testing.

Clearly, determining the type of features used by the learning algorithm is crucial to its performance. The feature space needs to be expressive enough to allow good approximation of the target functions using linear functions but without excessively increasing the resource requirements. We use the type of features identified in previous research on this problem - collocations: small conjunctions (size 2) of words and part of speech (POS) tags around the target word (up to three words away from the target) and context words in a small window (five words away from the target) around the target word. POS information is added to the text using a SNoW-based POS tagger (Roth & Zelenko 1998).

To avoid a vast amount of rare features we used an eligibility mechanism during the feature extraction process, which eliminated those that occurred less than 4 times. Overall, the feature space had 647,217 features, of which 495 were labels, 549,650 were collocations, and 97,072 were context words. All of the experiments were performed using the Winnow update rule within SNoW, with the following parameters: $\alpha = 1.3$, $\beta = 0.8$, $\theta = 1.0$, and a initial weight of 0.2. Two full epochs (passes through the training sample) were used.

Scaling Up

We describe several suggestions for handling issues that arise in scaling up context sensitive text correction, along with experiments exhibiting their effectiveness.

Network Pruning

Previous work on context sensitive text correction (Golding & Roth 1999; Mangu & Brill 1997) has clearly shown that learning with a larger number of features improves the performance. We describe a method for selectively pruning the effective number of features used, as part of the learning process, and show its effectiveness in reducing the memory requirements while minimally affecting the performance. It also reduces the evaluation time, which scales linearly with the number of active features in the example.

Confusion Set	Train WSJ Test WSJ	Train TDT2 Test WSJ	Train both Test WSJ
accept, except	90.6	94.5	93.2
affect, effect	96.7	96.1	96.4
among, between	87.3	89.5	90.1
amount, number	84.8	79.0	88.5
cite, sight, site	85.1	90.1	90.1
country, county	93.8	95.2	96.1
fewer, less	90.6	91.8	92.6
I, me	98.1	98.9	98.9
it's, its	98.8	99.0	99.2
lay, lie	74.3	85.1	83.8
passed, past	95.9	97.6	97.5
peace, piece	88.7	91.7	91.7
principal, principle	91.7	93.4	94.7
quiet, quite, quit	83.7	90.7	90.4
raise, rise	94.3	93.0	95.1
than, then	97.7	98.6	98.5
their, there, they're	97.2	98.5	98.6
weather, whether	97.0	98.0	98.0
you're, your	94.5	98.3	98.3
Set Average	96.0	96.7	97.2
All Sets Average	94.5	94.6	95.7

Table 2: Adaptation Results for Specific Confusion Sets: The WSJ train / WSJ test column gives performance from using the WSJ corpus only using 80-20% splits. The TDT2 train / WSJ test column gives performance for training on TDT2 and testing on the same 20% splits of the WSJ corpus. The "train both" column gives performance for training on TDT2, then training on the same 80% of WSJ as in the first experiment, then testing on the remaining 20%. These experiments were done using 5-fold cross-validation, with a 10% eligibility ratio.

The approach is based on the intuition that we need not rely on a feature that is observed with a given target very rarely[1]. We refer to this method as *eligibility*. The key issue, we found, is that this method of pruning needs to be done on a *per target* basis, and has to be relative to the sparseness of the target representation. We define an *eligibility ratio*, such that only a specified percentage of the most active features observed have a weight assigned and participate in predictions. This is done by making a first training pass through the data, creating a histogram of feature occurrences for each target, and then eliminating the least active features until we are left with the proper number of features. Another epoch of training is then performed with the remaining features.

The experiments use eligibility ratios of 100% (no pruning), 10%, and 1%. In each experiment, we used five-fold cross-validation, running five 80%-20% splits for each confusion set, so that each example for a given confusion set appeared as a test example once and a training example four times. Overall effects of eligibility on all our confusions sets as well as the details for 19 different confusion sets are

[1] A second, also intuitive, option – to prune based on the *weight* of the feature is not as effective for reasons we will not address here.

Confusion Set	Examples	Eligibility Ratio 1.0 Perf	1.0 Links	0.1 Perf	0.1 Links	0.01 Perf	0.01 Links
accept-except	2910	95.7	8169	96.7	1174	94.4	85
affect-effect	3278	94.2	8202	95.3	1264	94.1	84
among-between	20458	88.5	33279	90.0	4241	85.1	340
amount-number	9134	90.0	14248	89.9	1773	86.0	147
cite-sight-site	3983	91.9	5634	92.9	866	88.7	59
county-country	11897	95.6	16242	95.4	2196	93.4	166
fewer-less	7445	92.0	14986	92.9	1861	92.0	156
I-me	74022	99.0	45906	99.2	5077	98.4	463
it's-its	57843	97.3	62437	97.8	7321	96.8	633
lay-lie	1620	84.1	4479	85.8	692	77.7	48
passed-past	8772	95.6	15328	96.0	2011	92.9	158
peace-piece	7646	95.4	12018	96.3	1476	91.7	122
principal-principle	1270	87.2	4007	86.1	721	83.0	42
quiet-quite-quit	3836	89.6	7320	91.7	1203	87.7	78
raise-rise	3773	93.6	9120	93.8	1326	90.4	94
than-then	46651	97.0	53626	98.3	6692	97.5	544
their-there-they're	85407	97.4	57138	98.2	6790	96.7	582
weather-whether	12234	98.6	17698	98.4	2081	96.7	180
you're-your	14071	97.2	13668	97.8	2616	96.1	217
Set Average	376250	96.4	21639	97.1	2704	95.4	221
All Sets Average	6117483	95.5	17932	95.2	2233	89.5	183

Table 1: Effect of Eligibility on Specific Confusion Sets: We show results for 19 confusion sets for three eligibility ratios: 100% (no pruning), 10%, and 1%. Examples indicates the total number of examples for each confusion set. Each example was presented using 5-fold cross-validation with an 80%-20% split of the data. For each ratio value, Perf indicates the accuracy for the set, and Links are the average number of links (features) per target word.

shown in Table 1. We found that the size of the networks could be reduced greatly without a significant decrease in performance. Using a ratio of around 10% seemed to be a good cutoff point for the tradeoff between performance and size. This gives us an average across all confusion sets of 2,233 features per target, a substantial reduction from the original 17,932, with a slight drop in accuracy.

Adaptation

In order for a general text correction system to be useful, it needs to be able to perform well in domains other than the one on which it was trained. This is clearly an important issue in natural language processing given the diversity of text genres in terms of vocabulary and style. For a learning system there is an additional issue. There is a clear tradeoff between pruning the feature base of the hypothesis and its ability to adapt to new text (Herbster & Warmuth 1998). Intuitively, features which are rarely present in one domain could be important in another, but if they are pruned when trained on the first domain, the hypothesis will not be able to adapt its weights given the new texts.

We implemented an adaptation mechanism that is based on suggestions in (Golding & Roth 1999) and performed several experiments in order to examine the adaptation properties of our system in the presence of significant pruning (10%) and a large and diverse training corpus, the TDT2.

As a baseline, we ran experiments using 5-fold cross-validation on the Wall Street Journal corpus, using 80-20% splits. This gave us an overall performance of 94.5% for the weighted average across all 265 confusion sets. The WSJ corpus is rather small compared to the TDT2 corpus, and so we wondered if the extra data might help a network trained on the TDT2 corpus perform better on the WSJ test data. We found that the system was able to adapt even after significant pruning of features. Using all 265 confusions sets and 5-fold cross-validation, we trained on only the TDT2 corpus and tested on the same 20% slices of the Wall Street Journal as before. This gave overall accuracy of 94.6%, which was slightly better than the 94.5% obtained by training on WSJ only. This suggests that training on a large corpus such as TDT2 countered the effects of testing outside of the training domain. Finally, we tried to boost performance on the WSJ test data by adapting our already trained system to the new corpus by training it on the other WSJ data. When we trained on the TDT2 corpus as before, then trained on 80% of WSJ, and then tested on the leftover 20% of WSJ (the same test data as before), we reached 95.7% performance over all 265 confusion sets – a significant improvement over the results obtained when just WSJ is used in the training, which are 94.5%. The results are summarized in table 2.

These results indicate that even in the presence of significant feature pruning, the system can adapt well to new domains. Moreover, it suggests that in order to enhance performance on specific domains, it is beneficial to "fine-tune" it to this domain. We emphasize that this is costless, since context-sensitive text correction requires no annotation of the text - it assumes that the text is correct and uses this

Confusion Set	Examples	Prediction Threshold 0.05 Perf	0.05 Will	0.125 Perf	0.125 Will	0.2 Perf	0.2 Will
accept-except	2910	97.9	96.4	99.1	88.1	99.6	80.1
affect-effect	3278	96.7	95.9	97.4	88.3	97.6	75.9
among-between	20458	93.0	91.1	96.3	77.6	98.2	63.1
amount-number	9134	93.0	91.2	96.2	76.8	98.0	61.8
cite-sight-site	3983	95.3	93.2	97.4	81.5	98.6	66.8
county-country	11897	96.6	96.4	97.6	89.7	98.4	77.3
fewer-less	7445	89.9	94.6	96.7	85.9	98.0	73.7
I-me	74022	99.5	99.2	99.7	97.7	99.9	95.2
it's-its	57843	98.6	98.0	99.2	94.2	99.4	87.9
lay-lie	1620	89.9	88.8	93.4	71.3	96.8	57.0
passed-past	8772	97.6	95.7	99.0	88.6	99.5	80.0
peace-piece	7646	97.9	96.4	99.1	90.3	99.5	82.4
principal-principle	1270	90.3	87.5	94.1	68.4	96.3	51.5
quiet-quite-quit	3836	94.7	92.4	98.0	78.2	99.5	63.1
raise-rise	3773	96.0	94.7	97.7	84.4	98.7	72.4
than-then	46651	98.9	98.3	99.4	94.8	99.7	88.9
their-there-they're	85407	99.0	98.2	99.5	94.8	99.8	89.3
weather-whether	12234	98.9	98.6	99.4	95.8	99.7	91.7
you're-your	14071	98.5	98.1	99.2	93.8	99.6	87.4
Set Average	376250	98.1	97.3	99.0	92.6	99.5	85.9
All Sets Average	6117483	97.3	94.6	99.0	86.2	99.6	77.0

Table 3: Confidence Results for Specific Confusion Sets: Here we see results for three specific prediction thresholds. For each prediction threshold, Perf refers to the overall accuracy for predictions, and Will gives the Willingness of the system to make a prediction. Set Average refers to the average for the 19 sets shown here, and All Sets Average refers to the average across all 265 sets. All experiments were run using 5-fold cross-validation and a 10% eligibility ratio.

to label its training examples. And, as we show, it yields significantly better performance if the system is previously trained on the diverse corpus.

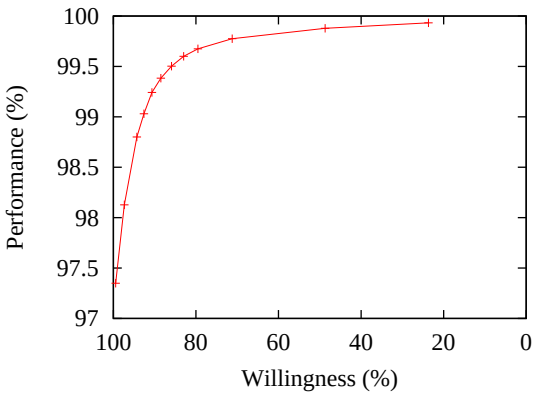

Figure 1: Performance vs. Willingness for 19 Confusion Sets

Prediction Confidence

We have seen that we can perform context-sensitive text correction with an accuracy of greater than 90%, and maintain that accuracy while scaling up to hundreds of confusion sets, and while pruning our networks to compact representations. However, performing at the 90–95% level is not sufficient for a practical system with wide coverage (that is, where many of the words in each sentence are in one of the confusion sets). In this case, if we make only 5 predictions per sentence, then our sentence level performance is only 50–75%. Even the most tolerant user would object to a system that makes a mistake every couple of sentences. Until we develop methods with basic performance in the range of 98–99%, our solution is to assign a confidence to predictions and make a prediction only when our confidence in that prediction is high. This approach requires that the learning approach assign a robust measure of confidence to its predictions so that this can be done reliably. Given that, our hope is that we can improve performance if we sacrifice some coverage; but, this will only be in cases in which we are not confident enough to voice our prediction. This will not annoy users but rather serve to increase their confidence in the system. An orthogonal benefit of this is that it also provides a mechanism for the user to adjust the confidence threshold at prediction-time. Users can adjust the behavior of the system to suit their personal preferences and abilities. Also, in most practical applications the user's word choice will be correct more often than not, and so abstaining from uncertain predictions will slightly favor the correct choice.

In order to explore this notion of confidence, we note that the activation of a target node is computed using a sigmoid function over the linear sum of *active* weights. Specifically,

the activation of the target t is given by

$$a_t = \frac{1}{1 + \exp\{\sum_{i \in \mathcal{A}_t} w_i^t - \theta\}}, \quad (1)$$

where $\mathcal{A}_t = \{i_1, \ldots, i_m\}$ is the set of features that are active in an example and are linked to the target node t, w_i^t is the weight on the edge connecting the ith feature to the target node t, and θ_t is the threshold for the target node t. With this, one can verify that the output behaves as a distribution function. A prediction is made only when

$$|a_1 - a_2| > C,$$

where a_1 and a_2 are the two highest activations in the confusion set, and C is the confidence threshold. If the confidence function does not exceed the threshold then no prediction is made. In a practical system, this is the equivalent of leaving the text as is – if we are not certain of our prediction, we leave the user's original word choice there.

For the experiment we used the same subset of 19 confusion sets presented in the previous experiments. The results are shown in figure 1. The *performance* axis is the percentage of predictions the system actually makes that are correct and the *willingness* is defined as the percentage of queries (occurrences of confusion set members) on which the system makes a prediction. So for example, a willingness of 80% means that the system is passive on 20% of the queries. The actual threshold used (C) is held fixed for all confusion sets. The experiment use five-fold cross validation as before and a 10% eligibility ratio.

We see that for the subset of 19 confusion sets, the performance rises above 99% when the willingness is around 92% (that is, by abstaining in only 8% of predictions).

Table 3 gives the results for both the 19 confusion sets and the average for all 265 sets. Each column represents a different value for the prediction threshold. Some sets which tend to do well in general (for example, $\{I, me\}$), have high accuracy and tend to have higher willingness than other sets for a given prediction threshold. In general, though, we see each set gaining substantially in accuracy as its willingness decreases. The averages for all 265 confusion sets show that we reach accuracy of 99% with willingness above 85%. These confidence experiments were all performed using a 10% eligibility ratio, demonstrating that we can effectively boost performance while cutting down on our resource requirements at the same time.

Conclusions

Intelligent human-machine interaction relies heavily on the ability to perform context-sensitive inferences. These are knowledge intensive tasks that are hard to perform without a significant learning component. The main challenge in an effort to build a realistic system with context-sensitive inference capabilities, beyond accuracy, is scalability.

In this work we study a learning approach to context sensitive text correction and directly address the crucial issue of scalability. While we have chosen to use a proven learning approach tailored towards large scale processes, significant enhancements in terms of both data and computation are still required before this can support a practical approach.

This paper has explored several issues relating to the scaling up of this task to provide wide word coverage while limiting resource requirements to reasonable levels and increasing the performance levels to those that are acceptable to users. The most significant finding is that a robust prediction confidence can be used to trade coverage for performance and a moderate reduction in willingness can increase the overall performance to over 99% – a level usable in a real-world system.

Acknowledgments

This research is supported by NSF grants IIS-9801638 and IIS-9984168 and a gift from IBM Research.

References

Carlson, A.; Cumby, C.; Rosen, J.; and Roth, D. 1999. The SNoW learning architecture. Technical Report UIUCDCS-R-99-2101, UIUC Computer Science Department.

Even-Zohar, Y., and Roth, D. 2000. A classification approach to word prediction. In *NAACL-2000, The 1st North American Conference on Computational Linguistics*, 124–131.

Golding, A. R., and Roth, D. 1996. Applying Winnow to context-sensitive spelling correction. In *Proc. of the International Conference on Machine Learning*, 182–190.

Golding, A. R., and Roth, D. 1999. A Winnow based approach to context-sensitive spelling correction. *Machine Learning* 34(1-3):107–130. Special Issue on Machine Learning and Natural Language.

Herbster, M., and Warmuth, M. K. 1998. Tracking the best regressor. In *Proc. 11th Annu. Conf. on Comput. Learning Theory*, 24–31. ACM Press, New York, NY.

Kukich, K. 1992. Techniques for automatically correcting words in text. *ACM Computing Surveys* 24(4):377–439.

Littlestone, N. 1988. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning* 2:285–318.

Mangu, L., and Brill, E. 1997. Automatic rule acquisition for spelling correction. In *Proc. 14th International Conference on Machine Learning*. Morgan Kaufmann.

Marcus, M. P.; Santorini, B.; and Marcinkiewicz, M. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics* 19(2):313–330.

Punyakanok, V., and Roth, D. 2001. The use of classifiers in sequential inference. In *NIPS-13; The 2000 Conference on Advances in Neural Information Processing Systems*. MIT Press.

Roth, D., and Zelenko, D. 1998. Part of speech tagging using a network of linear separators. In *COLING-ACL 98, The 17th International Conference on Computational Linguistics*, 1136–1142.

Roth, D.; Yang, M.-H.; and Ahuja, N. 2000. Learning to recognize objects. In *CVPR'00, The IEEE Conference on Computer Vision and Pattern Recognition*, 724–731.

Roth, D. 1998. Learning to resolve natural language ambiguities: A unified approach. In *Proc. National Conference on Artificial Intelligence*, 806–813.