# EVALUATION METRICS FOR LANGUAGE MODELS

*Stanley Chen, Douglas Beeferman, Ronald Rosenfeld**

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
{sfc,dougb,roni}@cs.cmu.edu

## ABSTRACT

The most widely-used evaluation metric for language models for speech recognition is the *perplexity* of test data. While perplexities can be calculated efficiently and without access to a speech recognizer, they often do not correlate well with speech recognition word-error rates. In this research, we attempt to find a measure that like perplexity is easily calculated but which better predicts speech recognition performance.

We investigate two approaches; first, we attempt to extend perplexity by using similar measures that utilize information about language models that perplexity ignores. Second, we attempt to imitate the word-error calculation without using a speech recognizer by artificially generating speech recognition lattices. To test our new metrics, we have built over thirty varied language models. We find that perplexity correlates with word-error rate remarkably well when only considering $n$-gram models trained on in-domain data. When considering other types of models, our novel metrics are superior to perplexity for predicting speech recognition performance. However, we conclude that none of these measures predict word-error rate sufficiently accurately to be effective tools for language model evaluation in speech recognition.

## 1. INTRODUCTION

In the literature, two primary metrics are used to estimate the performance of language models in speech recognition systems. First, they are evaluated by the word-error rate (WER) yielded when placed in a speech recognition system. Second, and more commonly, they are evaluated through their *perplexity* on test data, an information-theoretic assessment of their predictive power.

While word-error rate is currently the most popular method for rating speech recognition performance, it is computationally expensive to calculate. Furthermore, its calculation generally requires access to the innards of a speech recognition system, few of which are publically available. Finally, word-error rate is speech-recognizer-dependent, which makes it difficult for different research sites to compare language models with this measure.

Perplexity, on the other hand, can be computed trivially and in isolation; the perplexity $\mathrm{PP}(p_M)$ of a language model $p_M(\text{next word } w|\text{history } h)$ on a test set $T = \{w_1, \ldots, w_t\}$ is just

$$\mathrm{PP}_T(p_M) = \frac{1}{\left(\prod_{i=1}^{t} p_M(w_i|w_1 \cdots w_{i-1})\right)^{\frac{1}{t}}} \tag{1}$$

or the inverse of the (geometric) average probability assigned to each word in the test set by the model. Perplexity is theoretically elegant as its logarithm is an upper bound on the number of bits per word expected in compressing (in-domain) text employing the measured model. Unfortunately, while language models with lower perplexities tend to have lower word-error rates, there have been numerous examples in the literature where language models providing a large improvement in perplexity over a baseline model have yielded little or no improvement in word-error rate [1, 2]. In addition, perplexity is inapplicable to *unnormalized* language models (*i.e.*, models that are not true probability distributions that sum to 1), and perplexity is not comparable between language models with different vocabularies. In this research, we attempt to find a measure for evaluating language models that is applicable to unnormalized models and that predicts word-error rate more accurately than perplexity but which, like perplexity, is computationally inexpensive and can be computed separately from a speech recognition system. We consider two different approaches to this task.

Our first approach involves extending perplexity to utilize information that it previously ignores. As can be seen from equation (1), perplexity depends only on the probabilities assigned to actual text. However, word-error rate depends on the probabilities assigned to *all* transcriptions hypothesized by a speech recognizer; errors occur when an incorrect hypothesis has a higher score than the correct hypothesis. We consider metrics that harness this information.

Our second approach involves an attempt to mimic the process of calculating word-error rate through lattice rescoring, without actually using a speech recognition system to construct lattices. Instead, we artificially generate lattices and evaluate language models through their word-error rates on these artificial lattices.

To evaluate our novel language model measures, we have constructed over thirty language models of varying types, including class $n$-gram[3, 4], trigger[5], and cache[6] language models. We find that perplexity correlates with word-error rate remarkably well when only considering $n$-gram models trained on in-domain data. When considering other types of models, our novel metrics are superior to perplexity for predicting speech recognition performance. However, we conclude that none of these measures predict word-error rate sufficiently accurately to be effective tools for language model evaluation in speech recognition.

*This work was supported by the National Security Agency under grants MDA904-96-1-0113 and MDA904-97-1-0006 and by the DARPA AASERT award DAAH04-95-1-0475. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. government.

## 1.1. Previous Work

Iyer *et al.*[2] investigate the prediction of speech recognition performance for language models in the Switchboard domain, for trigram models built on differing amounts of in-domain and out-of-domain training data. Over the ten models they constructed, they find that perplexity predicts word-error rate well when only in-domain training data is used, but poorly when out-of-domain text is added. They find that trigram coverage, or the fraction of trigrams in the test data present in the training data, is a better predictor of word-error rate than perplexity. However, it is unclear how to extend $n$-gram coverage to comparing other types of models, such as class models or $n$-gram models of different order. In addition, this measure cannot distinguish between different models trained on the same data.

They also present techniques for building a decision tree that predicts the relative performance of two models on each word in a test set. Using this decision tree, they are able to predict with high accuracy the relative performance of pairs of trigram models. While this technique seems promising, the features used to build the tree include lexical information such as part-of-speech information and the phonetic lengths of words. In this work, we would like to investigate what is possible with measures like perplexity that ignore detailed lexical information.

## 1.2. Methodology

In this research, we investigate speech recognition performance in the Broadcast News domain. We generated narrow-beam lattices with the Sphinx-III recognition system[7] using a trigram model trained on 130M words of Broadcast News text; trigrams occurring only once were excluded from the model. The word-error rates reported in this work were calculated by rescoring these lattices with the given language model.

We created 35 language models, which we divided into two sets. Set A contains only $n$-gram models built on Broadcast News training data. The training set size, smoothing, $n$-gram order, and $n$-gram cutoffs were varied. Set B contains various kinds of models, including $n$-gram class models, trigram models enhanced with a cache or triggers, $n$-gram models built on out-of-domain data, and models that are an interpolation of $n$-gram models built on in-domain and out-of-domain data. In Table 1, we list the language models in each set. The held-out and test sets consist of 22,000 and 28,000 words, respectively, of Broadcast News data.

set A

| $n$ | data (wds) | smooth alg |
|---|---|---|
| 1 | 5M | K-N[8] |
| 2 | 5M | K-N |
| 3 | 5M | K-N |
| 4 | 5M | K-N |
| 5 | 5M | K-N |
| 3 | 5M | Katz[9] |
| 3 | 5M | poor |
| 3 | 10M | poor |
| 3 | 25M | poor |
| 3 | 5M | K-N (i) |
| 3 | 5M | K-N (ii) |
| 3 | 1M | K-N |
| 3 | 25M | K-N |
| 3 | 130M | K-N |
| 2 | 10M | K-N |
| 2 | 25M | K-N |
| 2 | 130M | K-N |

set B

| $n$ | description |
|---|---|
| 2 | class $n$-gram model |
| 3 | class $n$-gram model |
| 4 | class $n$-gram model |
| 3 | trigram model + cache 1 |
| 3 | trigram model + cache 2 |
| 3 | trigram model, Katz |
| 3 | Katz model + triggers 1 |
| 3 | Katz model + triggers 2 |
| 2 | AP news training data |
| 3 | AP news training data |
| 4 | AP news training data |
| 2 | Switchboard (SWB) data |
| 3 | Switchboard data |
| 4 | Switchboard data |
| 3 | AP and BN models mixed |
| 4 | AP and BN models mixed |
| 3 | SWB and BN models mixed |
| 4 | SWB and BN models mixed |

Table 1: Language models in sets A and B. The $n$ column describes the order of the $n$-gram model (*e.g.*, unigram or bigram). The *data* column describes the size of the training set used. In set A, the model labeled (i) excludes all bigrams and trigrams with only one count; the model labeled (ii) excludes all bigrams and trigrams with two or fewer counts. The abbreviation *K-N* stands for Kneser-Ney. The smoothing method *poor* is an algorithm specially designed to perform poorly. In set B, all models are trained on 5M words of data, have no $n$-gram cutoffs, and are smoothed with Kneser-Ney smoothing except where otherwise specified.

# 2. PERPLEXITY AND WORD-ERROR RATE

In Figure 1, we display a graph of word-error rate versus log perplexity for each of the models in sets A and B. The linear correlation between word-error rate and log perplexity seems remarkably strong for the models in set A, which consists of only $n$-gram models built on in-domain data, but less so for the models in set B, which is a more disparate collection of models. This indicates that log perplexity may be a good predictor of speech recognition performance when considering only particular types of models.

It seems somewhat surprising that log perplexity, which is measured in *bits* (recall the information theoretic interpretation of perplexity mentioned in Section 1), is correlated with the very different unit of *word errors*. To attempt to shed light on why these two apparently unrelated quantities are related, in Figure 2 we graph the relationship between the language model probability assigned to a word in a test set and the chance that word is transcribed correctly in speech recognition. The dotted lines represent curves for each of the individual models in sets A and B. To generate each curve, we first calculated the probability assigned by the given model to each word in our held-out set, and placed these words in logarithmically-spaced buckets based on these probabilities. Then, from the corresponding speech recognition run we used NIST's *sclite* software to mark each word in the held-out set as correct or incorrect. Finally, we calculated the fraction of words in each bucket that are correct or incorrect.

To relate log perplexity and word-error rate, consider approximating the curves in Figure 2 as a straight line, *i.e.*,

$$p_M(w \text{ is correct}) \approx a_1 \log p_M(w|h) + a_2$$

for all models $M$ for some constants $a_1$ and $a_2$, where $p_M(w|h)$ denotes the language model probability assigned to word $w$ by model $M$ given history $h$. Then, for a test set $T = \{w_1, \ldots, w_t\}$ the expected word accuracy is

$$\begin{aligned} \frac{1}{t}\sum_{i=1}^{t} p_M(w_i \text{ is correct}) &= \frac{1}{t}\sum_{i=1}^{t}\left[a_1 \log p_M(w_i|h_i) + a_2\right] \\ &= -a_1 \log \mathrm{PP}_T(p_M) + a_2 \end{aligned}$$

*i.e.*, the expected word accuracy is a linear function of the perplexity. If we make the approximation that word-error rate is a linear function

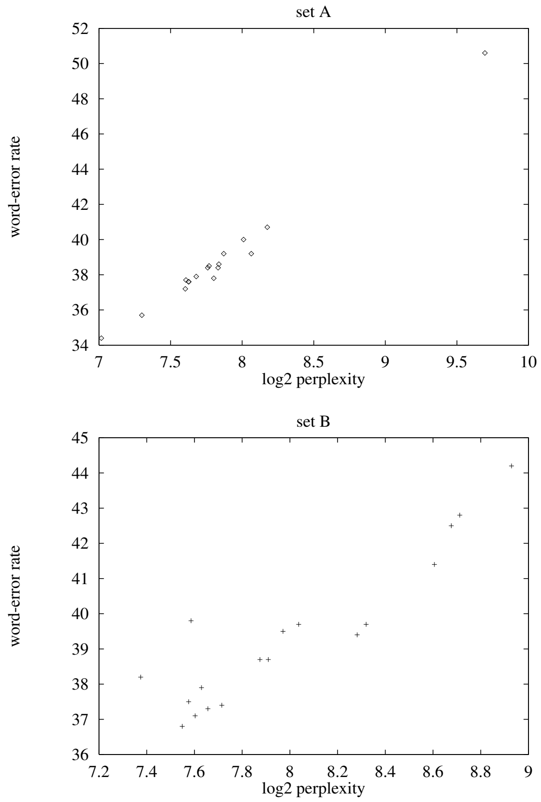


Figure 1: Word-error rate *vs.* log perplexity

of word accuracy, then we have that word-error rate is also a linear function of perplexity.

This analysis, while very rough, does lend some insight as to why perplexity and word-error rate are at all related, and suggests where perplexity might be improved and where the perplexity-WER relationship might break down. For example, it is clear that the linear approximation is poor for very low probabilities, where the probability of correctness is predicted to be less than zero.

# 3. EXTENDING PERPLEXITY

## 3.1. Modeling the Relation between Language Model Probability and Word Accuracy

One natural technique to try given the analysis in Section 2 is to use the functions displayed in Figure 2 to estimate word-error rate. That is, since our use of log perplexity to predict word-error rate can be viewed as being based on a hypothesis that these functions are linear, we might do better with an empirically-estimated function. To implement this technique, for each model we calculated the probability assigned to each word in our test set and placed these words into log-spaced buckets based on these probabilities. We calculated the average over all curves in Figure 2 to estimate the fraction of words correct in each bucket, and collated results over all buckets to get a final estimate of word accuracy. We subtract from 1 to produce an estimate of word-error rate, and call this measure *M-ref*. We graph this value versus real word-error rate in Figure 3 for set B.

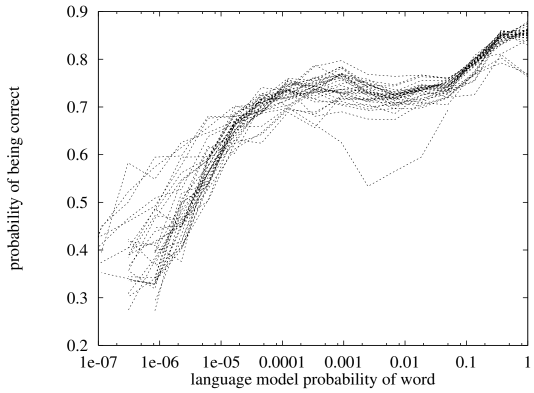


Figure 2: Probability of a word being correct in speech recognition given its language model probability. Each line represents one of the language models in sets A and B.

To quantify the correlation between different metrics with word-error rate, we calculate the *linear correlation coefficient* (or *Pearson's* $r$) measuring the degree of linear correlation; the *Spearman rank-order correlation coefficient* measuring how well the *ranks* of models linearly correlate; and *Kendall's* $\tau$ measuring how well the relative performance of pairs of models is predicted. In Table 2, we display these correlations for perplexity and *M-ref* versus word-error rate. For set A, perplexity correlates with word-error rate better than measure *M-ref* according to all three measures, while for set B measure *M-ref* is marginally better.

## 3.2. Using Additional Information

Perplexity and *M-ref* depend only on the probabilities of words in the test set, which in speech recognition is simply the reference transcript. However, word-error rate depends also on the probabilities assigned to *incorrect* hypotheses; in particular, errors occur when an incorrect hypothesis outscores the correct hypothesis. For example, it seems intuitive that errors are more likely to occur when many incorrect words are assigned large language model probabilities.

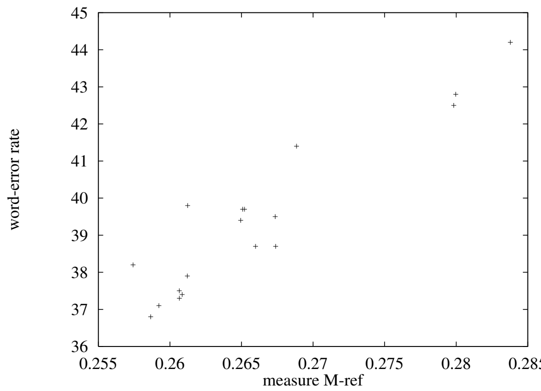


Figure 3: Word-error rate *vs.* measure *M-ref* on set B

| | set A | | | set B | | |
|---|---|---|---|---|---|---|
| | linear | rank | pair | linear | rank | pair |
| PP | 0.99 | 0.97 | 0.88 | 0.92 | 0.80 | 0.69 |
| *M-ref* | 0.94 | 0.80 | 0.68 | 0.93 | 0.86 | 0.69 |

Table 2: Correlations of perplexity and measure *M-ref* with word-error rate

We considered two methods for estimating the effect of overall language model probabilities on word-error rate: first, we examined the relationship between the *absolute* language model probability assigned to a word and the frequency with which that word occurs as an error in speech recognition; and secondly, we examine this relationship except using the *relative* language model probability of a word as compared to the probability assigned to the correct word. When we say a word occurs as an error, we mean that the word occurred in the transcription hypothesized by the speech recognizer but was marked as incorrect in word-error rate scoring. It is likely that both absolute and relative probabilities are relevant in determining how frequently a word occurs as an error: if the correct hypothesis has a very high score, then relative probability is probably more important; otherwise, absolute probability may play a larger role.

To estimate the relation between absolute probability and error frequency, we calculated the language model probability assigned to each word in the hypothesis for each utterance in our held-out set. We placed each word deemed incorrect by *sclite* in logarithmically-spaced buckets according to language model probability, to find the frequency of errors in each bucket. To estimate the frequency of words occurring in each bucket in the language model, we evaluated the given language model over all words in the vocabulary over our held-out set; *i.e.*, for held-out data $T = \{w_1, \ldots, w_t\}$ we evaluated probabilities of the form $p(w|w_1 \cdots w_{i-1})$ for all $i \in \{1, \ldots, t\}$ and all words $w$. Dividing the errors per bucket by the total number of words in each bucket yields an estimate of the probability of a word occurring as an error given its language model probability; this

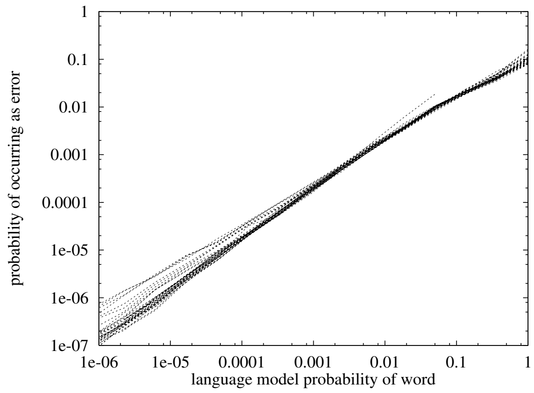


Figure 4: Relation between language model probability of a word and the frequency with which the word occurs as an error. Each line represents one of the language models in sets A and B.

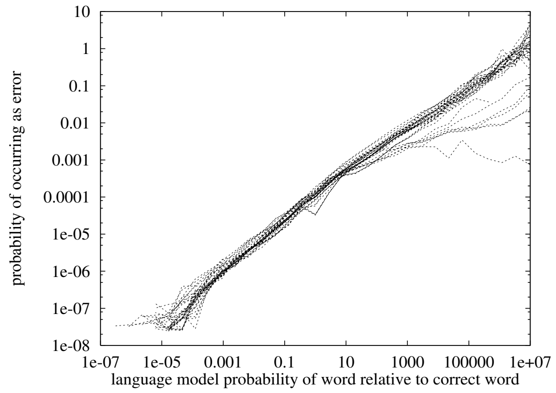


Figure 5: Relation between language model probability of a word relative to the correct word and the frequency with which the word occurs as an error

quantity is graphed in Figure 4.[1] The different lines correspond to each individual model. It is interesting to note the small variation between the curves for each model, as well as the linearity of the curves as plotted in log-log scale.

To estimate the relation between *relative* probability and error frequency, we used a similar procedure as for absolute probability except that in each step, instead of bucketing by absolute probability we bucket by the ratio between the probability of the given word and the "correct" word. In order to determine the "correct" word, we only consider substitution errors in this analysis. In calculating the language model probability of the correct word, we use the same history as was used to calculate the language model probability of the given word. Then, using a similar procedure as was described above, we produce the graph displayed in Figure 5. Again, the curves are quite linear (in log-log space) and tightly packed, though not as tightly as in the previous graph.

We can use these graphs to create new metrics that approximate word-error rate. Since this information is largely orthogonal with perplexity, it may be possible to combine the two to achieve a stronger metric. We have yet to explore this avenue.

# 4. ARTIFICIAL LATTICES

Instead of predicting speech recognition performance by examining basic features of a language model such as perplexity, another approach is to attempt to mimic the process of calculating word-error rate, except without using a speech recognizer. In this section, we discuss methods for artificially generating speech recognition lattices. Word-error rates calculated on these artificial lattices can be used to evaluate language models, and we describe a method for constructing lattices such that these *artificial word-error rates* correlate well with word-error rates calculated on genuine lattices. In addition, the lattices constructed are very narrow, so that artificial

[1] It is unclear how to count how often a word *occurs* in each bucket; *e.g.*, during speech recognition, language model probabilities for a word may be estimated multiple times at each position in the utterance with different histories. For the purposes of this calculation, we pretend that a total of $|V|$ words "occur" at each word position in an utterance where $V$ is the vocabulary used, and normalize accordingly.

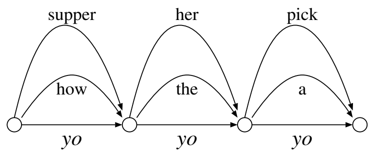


Figure 6: An example artificial lattice for the utterance *yo yo yo*

word-error rates can be calculated quickly.

In generating lattices, we have made several simplifying assumptions, and have found that the method still works well. First, we assume that the correct hypothesis is always in the lattice. Secondly, we assume that all words in a lattice are perfectly time-aligned with the correct hypothesis; *i.e.*, all words in a lattice have the same begin and end times as a word in the correct hypothesis — only substitution errors are considered. One advantage of this assumption is that all hypotheses are the same length in words, and an insertion penalty has no effect and can be ignored. Thirdly, we assume that there will be a few words that will be acoustically confusable with each word in the correct hypothesis, and that these words will have the same acoustic score as the correct word. This is equivalent to only including "acoustically confusable" words at each position in the lattice, and setting all acoustic scores to zero. With this assumption, the language weight becomes irrelevant since all hypotheses have the same acoustic score.

Our algorithm for generating a lattice on a test-set utterance is as follows. We begin with a lattice that just contains the correct path. The start frames and end frames of each word are unimportant, since all words in the lattice will be time-aligned. Then, for each word in the utterance, we randomly generate (according to a distribution to be specified) $k$ words that occur in the same position (*i.e.*, have the same begin and end times). Typically, we have taken $k$ to be about 9. All acoustic scores are set to zero. In Figure 6, we show an artificial lattice for the utterance *yo yo yo* with $k = 2$.

To generate the words that are "acoustically confusable" with each word in the utterance, one possibility is to determine which words are acoustically nearby. However, we make the assumption that whether we choose random words or genuinely acoustically confusable words will not affect word-error rate, and use a single probability distribution to generate alternatives for all words. One distribution that seems reasonable to use is the unigram distribution $p_U(w)$, which just reflects the frequency of words $w$ in the training text. We have found empirically that distributions of the form $p_U(w)^\alpha$ produce lattices that do well in predicting actual word-error rate, where the value $\alpha = 0.5$ has worked well in both Broadcast News and Switchboard experiments.

Using the value $k = 9$, we generated artificial lattices over our entire test set. We calculated word-error rates on these artificial lattices for all of our models in sets A and B, and in Figure 7 we display a graph of artificial word-error rate *vs.* actual word-error rate over these models. In Table 3, we display the correlation between artificial word-error rate and actual word-error rate. Perplexity is marginally better on set A, but artificial word-error rate is substantially superior on set B, the motley mix of models.

We have also performed experiments on the Switchboard task using lattices generated by the Janus speech recognition system[10].

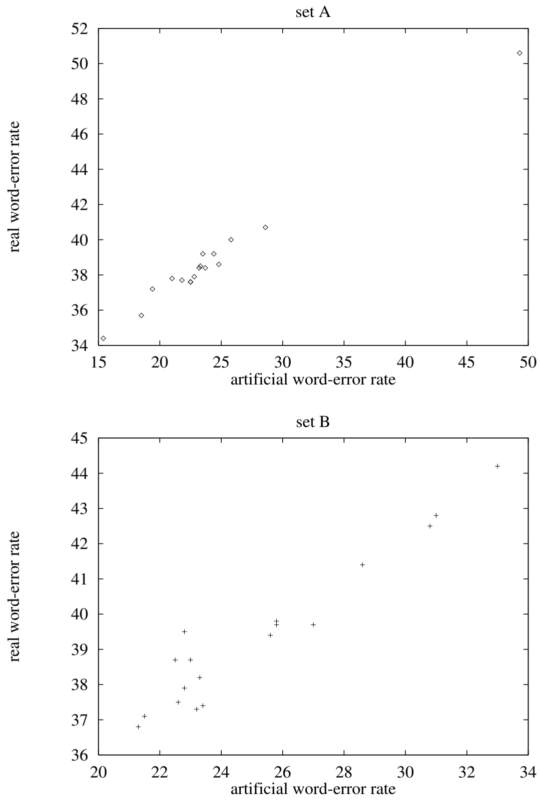


Figure 7: Actual word-error rate *vs.* artificial word-error rate for models in sets A and B

Generating artificial lattices with the values $k = 3$ and $\alpha = 0.5$, we compared the correlation between perplexity and artificial word-error rate with actual word-error rate over nine $n$-gram models. The $n$-gram models were built with varying training data sizes, count cutoffs, smoothing, and $n$-gram order. In Table 3, we display the correlations for perplexity and artificial word-error rate; artificial word-error rate is superior on this data set.

In terms of computation, we compare the different metrics through the language model probability evaluations required per word in the test set. Perplexity requires only one language model evaluation per

Broadcast News

| | set A | | | set B | | |
|---|---|---|---|---|---|---|
| | linear | rank | pair | linear | rank | pair |
| PP | 0.99 | 0.97 | 0.88 | 0.92 | 0.80 | 0.69 |
| AWER | 0.99 | 0.96 | 0.86 | 0.96 | 0.86 | 0.74 |

Switchboard

| | linear | rank | pair |
|---|---|---|---|
| PP | 0.85 | 0.73 | 0.56 |
| AWER | 0.93 | 0.83 | 0.67 |

Table 3: Correlations of perplexity and artificial word-error rate with actual word-error rate

word, and is by far the most efficient. For a trigram model, artificial word-error rate requires at most $k^3$ language model evaluations per word; in practice, the actual value was about 300 for $k = 9$. The time required to rescore artificial lattices on our 22,000 word held-out set on a 300 Mhz Pentium II machine ranged from 2 minutes for a trigram model to 33 minutes for a trigram model with triggers. Rescoring actual lattices with a trigram model required about 3600 language model evaluations per word. The computation time required varied from 1.6 hours for a trigram model to 18.2 hours for a trigram model with triggers. Thus, calculating artificial word-error rate, while significantly more expensive than calculating perplexity, is still much less expensive than rescoring genuine lattices and the absolute times involved are quite reasonable.

# 5. DISCUSSION

In this work, we have shown that perplexity can predict word-error rate quite well for conventional $n$-gram models trained on in-domain data. However, for models of a more disparate nature, perplexity is a poorer predictor. We have developed a measure, *M-ref*, that extends perplexity and better predicts word-error rate for complex language models. We have also described a technique for generating artificial lattices such that word-error rates calculated on these lattices correlate with actual error rates better than perplexity. The error-rate calculation over these lattices is quite inexpensive.

Despite this work, it is still unclear whether perplexity or our novel evaluation metrics are effective tools for language modeling researchers. Perplexity has been a popular comparison measure historically because it allows language model research to develop in isolation from speech recognizers, and it has many theoretically elegant properties. Unfortunately, this modularization of language modeling is justified only if our isolated measures can predict application performance accurately enough. While perplexity is an indication of performance in the application of text compression, it has been shown to be inadequate in predicting speech recognition performance. For example, one basic criterion of a language model evaluation metric is that it can distinguish between language models whose application performances are significantly different. A word-error rate difference of 0.5% or 1.0% absolute is often considered significant; if we refer to Figure 1, we find models with essentially the same perplexity that differ by more than 1.0% in error rate. This property is also true of the novel evaluation metrics that we have described. In practice, during language model development for the Hub 4 evaluations we have discontinued calculating perplexities and instead calculate word-error rates directly to decide whether any changes are useful. Experience has dictated that this is the most effective course of action.

We consider it unlikely that any accurate measure can be developed that, like perplexity, is based only on language model features. This is because a great many factors affect speech recognition performance: the values of the language weight and insertion penalty; the search algorithm used (search algorithms for long-distance models tend to be less effective); the stage at which the language model is applied (decoding, lattice rescoring, or $n$-best list rescoring); the language models used in the other stages; and the interaction of the language model with the acoustic model. All of these factors significantly impact recognition performance, and it is unclear how any metric that is blind to these factors could compensate for their effects.

Measures that imitate the speech-recognition process can abstract over many of these issues. For example, in artificial lattice generation, the search algorithm is not an issue if we assume different search algorithms over artificial lattices cause the same variation in performance as in real lattices. If we have acoustic scores in our artificial lattices, then we can optimize language weights over artificial lattices just as in real lattices. However, as measures become more complex and expensive to compute, calculating word-error rates directly will become a more attractive alternative.

In conclusion, existing measures such as perplexity or our novel measures are not accurate enough to be effective tools in language model development for speech recognition, and it is unclear how useful it is to continue to compare language models for speech recognition using perplexity. While this leaves researchers with the unpleasant requirement that they compare language models only with respect to the same speech recognizer, it does not seem there is a reasonable alternative unless more effective measures are developed. There are techniques for making word-error rate computation less expensive, such as $n$-best list rescoring or lattice rescoring with narrow-beam lattices, and such techniques are in common use in practice. Indeed, to move solely to word-error rate reporting just mirrors the decision made long ago in acoustic modeling, that acoustic models can only be accurately judged in the context of a speech recognition system.

# References


1. S.C. Martin, J. Liermann, and H. Ney. Adaptive topic-dependent language modelling using word-based varigrams. In *Proceedings of Eurospeech '97*, 1997.
2. R. Iyer, M. Ostendorf, and M. Meteer. Analyzing and predicting language model improvements. In *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding*, 1997.
3. Peter F. Brown, Vincent J. Della Pietra, Peter V. deSouza, Jennifer C. Lai, and Robert L. Mercer. Class-based n-gram models of natural language. *Computational Linguistics*, 18(4):467–479, December 1992.
4. Hermann Ney, Ute Essen, and Reinhard Kneser. On structuring probabilistic dependences in stochastic language modeling. *Computer, Speech, and Language*, 8:1–38, 1994.
5. D. Beeferman, A. Berger, and J. Lafferty. A model of lexical attraction and repulsion. In *Proceedings of the ACL*, Madrid, Spain, 1997.
6. R. Kuhn and R. De Mori. A cache-based natural language model for speech reproduction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(6):570–583, 1990.
7. P. Placeway, S. Chen, M. Eskenazi, U. Jain, V. Parikh, B. Raj, M. Ravishankar, R. Rosenfeld, K. Seymore, M. Siegler, R. Stern, and E. Thayer. The 1996 Hub-4 Sphinx-3 system. In *Proceedings of the DARPA Speech Recognition Workshop*, February 1997.
8. Reinhard Kneser and Hermann Ney. Improved backing-off for m-gram language modeling. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 1, pages 181–184, 1995.
9. Slava M. Katz. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Transactions on Acoustics, Speech and Signal Processing*, ASSP-35(3):400–401, March 1987.
10. Ivica Rogina and Alex Waibel. The Janus speech recognizer. In *ARPA SLT Workshop*, 1995.