IMPROVING TRIGRAM LANGUAGE MODELING WITH THE WORLD WIDE WEB

Xiaojin Zhu and Ronald Rosenfeld

School of Computer Science, Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh, PA 15213, USA
{zhuxj, roni}@cs.cmu.edu

ABSTRACT

We propose a novel method for using the World Wide Web to acquire trigram estimates for statistical language modeling. We submit an N-gram as a phrase query to web search engines. The search engines return the number of web pages containing the phrase, from which the N-gram count is estimated. The N-gram counts are then used to form web-based trigram probability estimates. We discuss the properties of such estimates, and methods to interpolate them with traditional corpus based trigram estimates. We show that the interpolated models improve speech recognition word error rate significantly over a small test set.

1. INTRODUCTION

A language model (LM) is a critical component for many applications, including speech recognition. Enormous effort has been spent on building and improving language models. Broadly speaking, this effort develops along two orthogonal directions: The first direction is to apply increasingly sophisticated estimation methods to a fixed training data set (corpus) to achieve better estimation. Examples include various interpolation and backoff schemes for smoothing, variable length N-grams, vocabulary clustering, decision trees, probabilistic context free grammar, maximum entropy models, etc [1]. We can view these methods as trying to "squeeze out" more benefit from a fixed corpus. The second direction is to acquire more training data. However, automatically collecting and incorporating new training data is non-trivial, and there has been relatively little research in this direction. An example is a cache LM, which uses recent utterances as additional training data to create better N-gram estimates. The recent rapid development of the World Wide Web (WWW) makes it an extremely large and valuable data source. Just-in-time language modeling [2] submits previous user utterances as queries to WWW search engines, and uses the retrieved web pages as unigram adaptation data. In this paper, we propose a novel method for using the WWW and its search engines to derive additional training data for N-gram language modeling, and show significant improvements in terms of speech recognition word error rate. An extended version of this paper can be found in [3].

The authors are grateful to Stanley Chen, Matthew Siegler, Chris Paciorek and Kevin Lenzo for their help. The first author was supported in part by NSF LIS under grant REC-9720374 and Microsoft Research Graduate Fellowship.

2. THE WWW AS TRIGRAM TRAINING DATA

The main idea of our method is to obtain the counts of "$w_1 w_2 w_3$" and "$w_1 w_2$" as they appear on the WWW, to estimate

$$\hat{p}_{web}(w_3|w_1, w_2) = \frac{c_{web}(w_1 w_2 w_3)}{c_{web}(w_1 w_2)}$$

and combine $\hat{p}_{web}$ with the estimates from a traditional corpus (here and elsewhere, when $c_{web}(w_1 w_2) = 0$, we regard $\hat{p}_{web}$ as unavailable). Essentially, we are using the searchable web as additional training data for trigram language modeling. There are several questions to be addressed: How to obtain the counts from the web? What is the quality of these web estimates? How could they be used to improve language modeling? We will examine these questions in the following sections, in the context of N-best list rescoring for speech recognition.

2.1. Obtaining N-gram counts from the WWW

To obtain the count of an N-gram "$w_1 \ldots w_n$" from the web, we send it as a single quoted phrase query to a search engine. We want the search engine to perform exact phrase search (i.e. don't use a stopword list or stemming), and return phrase counts or web page counts (from which we can estimate phrase counts, see below). We experimented with a dozen popular search engines, and found three that meet our criteria: AltaVista [4] advanced search mode, Lycos [5], and FAST [6] [1]. They all report web page counts.

One brute force method to get the phrase counts is to actually download all the web pages the search engine finds. However, queries of common words typically result in tens of thousands of web pages, and this method is clearly infeasible. Fortunately at the time of our experiment AltaVista had a simple search mode, which reported both the phrase count and the web page count for a query. Figure 1(a) shows the phrase count vs. web page count for 430 bigram queries (phrases consisting of two consecutive words). Trigram queries and unigram queries have similar behavior. There are horizontal branches in the bigram and trigram plots that don't make sense (more web pages than total phrase counts). We regard these as outliers due to idiosyncrasies of the search engine, and exclude them from further consideration. Since the plots are largely log-linear, we perform log-linear regression $c = \alpha_0 * pg^{\alpha_1}$ separately for trigrams, bigrams, and unigrams, where c is the phrase count, and pg the web page count. We found for unigram $\alpha_0 = 2.427, \alpha_1 = 1.019$; for bigram $\alpha_0 = 1.209, \alpha_1 = 1.014$;

[1] Our selection is admittedly incomplete. In addition, since search engines develop and change rapidly, all our comments are only valid during the period of this experiment

0-7803-7041-4/01/$10.00 ©2001 IEEE

and for trigram $\alpha_0 = 1.174, \alpha_1 = 1.025$. The bigram regression function is also plotted in Figure 1(a). We assume these functions apply to other search engines as well. In the rest of the paper, all web N-gram counts are estimated by applying the corresponding regression function to the web page counts reported by search engines.

2.2. The quality of web estimates

To investigate the quality of web estimates, we needed a baseline corpus for comparison. The baseline we used is a 103 million word Broadcast News corpus.

Web N-gram coverage on unseen test text: We show that the web covers many more N-grams than the baseline corpus. Note that by 'the web' we mean the searchable portion of the web as indexed by the search engines we chose. A test text consisted of 24 randomly chosen sentences from 4 web news sources (CNN, ABC, Fox, BBC) and 6 categories (world, domestic, technology, health, entertainment, politics) was created from the day's news stories, on the day the experiment was carried out. This was to make sure that the search engines hadn't had the time to index the web pages containing these sentences. After the experiment was completed, we checked each sentence, and indeed none of them were found by the search engines yet. Therefore the test text is truly unseen to both the web search engines and the baseline corpus. (The test text is of written news style though, which might be slightly different from the broadcast news style in the baseline corpus.)

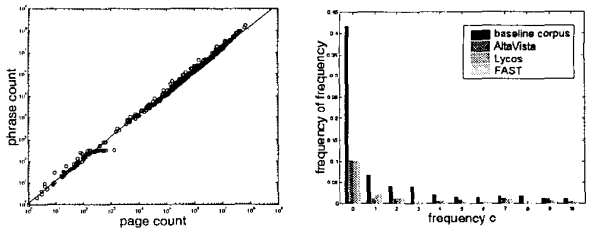
	unique	Not Covered By			
	types	AltaVista	Lycos	FAST	Corpus
1g	327	0	0	0	8
2g	462	4	5	5	68
3g	453	46	46	46	189

Table 1. Novel N-gram types in 24 news sentences

There are 327 unigram types (i.e. unique words), 462 bigram types and 453 trigram types in the test text. Table 1 lists the number of N-gram types not covered by the different search engines and the baseline corpus, respectively. Clearly, the web's coverage, under any of the search engines, is much better than that of the baseline corpus. It is also worth noting that for this test text, any N-gram not covered by the web was also not covered by the baseline corpus.

Next we asked the question "if one randomly picks a trigram from the test text, what's the chance the trigram has appeared c times in the training data?" Figure 1(b) shows the comparison, with the training data being the baseline corpus and the web through the different search engines respectively. This figure is also known as a "frequency-of-frequency" plot. According to this figure, a trigram from the test text has more than 40% chance of being absent in the baseline corpus, but the chance goes down to about 10% on the web regardless of the search engine. This is consistent with Table 1. Moreover, the trigram has a much larger chance in having a small count in the baseline corpus than on the web. Since small counts usually mean unreliable estimates, resorting to the web could be beneficial.

The effective size of the web: The web is large, and we would like to estimate the effective size of the web as if it were a LM training corpus. Let's assume that the web and the baseline corpus



(a) (b)

Fig. 1. (a) Web page count vs. phrase count for bigrams (b) Empirical frequency-of-frequency plot

are homogeneous (which is patently false, but we will ignore this for the time being), i.e. $p_{corpus}(\text{N-gram}) = p_{web}(\text{N-gram})$. Approximating the probabilities by their respective frequencies, we can estimate the size of the web in words. Each N-gram with a large count in the test text will gave us an estimate, and we took the median of all estimates for robustness. We found for AltaVista, Lycos and FAST, the effective size is 108, 79 and 83 billion words respectively. Note these are very rough estimates defined relative to the specific baseline corpus and specific test set we happened to choose. They should not be used to rank the performance of individual search engines.

Normalization of the web counts: A sanity check is to see if $c_{web}(w_1 w_2) = \sum_{w_3 \in V} c_{web}(w_1 w_2 w_3)$ holds for any bigram "$w_1 w_2$". If true, $\hat{p}_{web}(w_3|w_1, w_2)$ would already be normalized. We randomly selected six "$w_1 w_2$" pairs and performed the check. The right hand side ranges between 0.45 to 1.17 times the left hand side. Therefore the web counts are not perfectly normalized, and they should be used with caution.

Web trigram estimates compared to a traditional LM: We created a baseline trigram language model LM_0 from the 103 million word baseline corpus. We used modified Kneser-Ney smoothing [7] [8] which, according to [8], is one of the best smoothing methods available. In building LM_0, we discarded all singleton trigrams in the baseline corpus, a common practice to reduce language model size. We denote LM_0's probability estimates by p_0. We compare $\hat{p}_{web}(w_3|w_1, w_2)$ with $p_0(w_3|w_1, w_2)$for the trigrams in the test text. We found that the two agree in most cases, while the web estimates tend to be larger for small count trigrams. This is of course good news, as it suggests that this is where the web estimates tend to improve on the corpus estimates.

3. COMBINING WEB ESTIMATES WITH EXISTING LANGUAGE MODEL

In the previous section we saw the potential of the web: it is huge, it has better trigram coverage, and its trigram estimates are largely consistent with the corpus-based estimates. But to query each and every N-gram on the web is infeasible; web estimates are not well normalized; and the content of the web is heterogeneous and usually doesn't coincide with our domain of interest. Based on these considerations, we decided not to build a full fledged LM from the web. Instead we will start from a traditional language model LM_0, and interpolate its least reliable trigram estimates with the appropriate estimates from the web, since unreliable trigram estimates, especially those involving backing off to lower order N-grams,

have been shown to be correlated with increased speech recognition errors [9] [10]. We define a trigram estimate $p_0(w_3|w_1,w_2)$ unreliable if $c_{corpus}(w_1w_2w_3) \leq \tau$, where τ is the reliability threshold.

Even so there are still too many unreliable trigrams to query the web for. Since we were interested in N-best list rescoring, we further restricted our attention to those unreliable trigrams that appeared in the particular N-best list being processed. This greatly reduces the number of web queries at the price of some further bias. Let $U_{w_1w_2}$ be the set of w_3's in the current N-best list that form unreliable trigrams with history "w_1w_2". We would aquire $\hat{p}_{web}(u|w_1,w_2), u \in U_{w_1w_2}$ from the web, and interpolate them with $p_0(u|w_1,w_2)$ to form the final interpolated estimates, denoted as $p^*(u|w_1,w_2)$. We would like to have a tunable interpolation parameter so that on one extreme $p^*(u|w_1,w_2)$ goes to $p_0(u|w_1,w_2)$, while on the other extreme it goes to $\hat{p}_{web}(u|w_1,w_2)$. We now present three different methods for doing this.

In *exponential models with Gaussian priors* we define a set of binary functions, or 'features', as follows:

$$f_{w_1,w_2,u}(w_3) = \begin{cases} 1 & \text{if } w_3 = u \\ 0 & \text{otherwise} \end{cases}$$

for all $w_1, w_2, u \in U_{w_1w_2}$ in the N-best list. We define a conditional exponential model [11] [12] p^*_E with these features:

$$p^*_E(w_3|w_1,w_2) = \frac{1}{Z_{w_1w_2}} p_0(w_3|w_1,w_2) \exp\left(\sum_{u \in U_{w_1w_2}} \lambda_u f_{w_1,w_2,u}(w_3)\right) \quad (1)$$

Let Λ denote the set of parameters. The likelihood with respect to the web counts is:

$$L(\Lambda) = \prod_{w_1,w_2,w_3} p^*_E(w_3|w_1,w_2)^{c_{web}(w_1w_2w_3)}$$

We introduce a Gaussian prior with mean 0 and variance σ^2 over Λ:

$$p(\Lambda) = \prod_i \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{-\lambda_i^2}{2\sigma^2}\right)$$

and seek the maximum a posteriori (MAP) solution that maximizes $L(\Lambda) * p(\Lambda)$. This can be done by slightly modifying the Generalized Iterative Scaling algorithm [13], as described in [14].

We can control the degree of interpolation by choosing the prior variance $\sigma^2 \in (0, +\infty)$. If $\sigma^2 \to +\infty$, the Gaussian prior is flat and has virtually no restriction on the values of the λ's. Thus the λ's can reach their ME/MDI solutions that satisfies the following constraints:

$$p^*_E(u|w_1,w_2) = \hat{p}_{web}(u|w_1,w_2), \forall u \in U_{w_1w_2} \quad (2)$$

On the other hand if $\sigma^2 \to 0$, the Gaussian prior forces λ's to be close 0. From (1) we know in this case $p^*_E \to p_0$, the other extreme of the interpolation. A σ^2 between 0 and $+\infty$ results in an intermediate p^*_E distribution.

In *linear interpolation*, we have

$$p^*_L(w_3|w_1,w_2) = \begin{cases} (1-\alpha)p_0(w_3|w_1,w_2) + \alpha\hat{p}_{web}(w_3|w_1,w_2) \\ \qquad \text{, if } w_3 \in U_{w_1w_2} \\ \frac{1-\sum_{u \in U_{w_1w_2}} p^*_L(u|w_1,w_2)}{1-\sum_{u \in U_{w_1w_2}} p_0(u|w_1,w_2)} p_0(w_3|w_1,w_2) \\ \qquad \text{, otherwise} \end{cases} \quad (3)$$

In this case, $\alpha \in [0,1]$ is the tuning parameter. If $\alpha = 0$, $p^*_L = p_0$. If $\alpha = 1$, p^*_L satisfies (2). An α in between results in an intermediate p^*_L.

In *geometric interpolation*, we have

$$p^*_G(w_3|w_1,w_2) = \begin{cases} p_0(w_3|w_1,w_2)^{(1-\beta)} \left[\frac{c_{web}(w_1w_2w_3)+\epsilon}{c_{web}(w_1w_2)+|V|\epsilon}\right]^\beta \\ \qquad \text{, if } w_3 \in U_{w_1w_2} \\ \frac{1-\sum_{u \in U_{w_1w_2}} p^*_G(u|w_1,w_2)}{1-\sum_{u \in U_{w_1w_2}} p_0(u|w_1,w_2)} p_0(w_3|w_1,w_2) \\ \qquad \text{, otherwise} \end{cases} \quad (4)$$

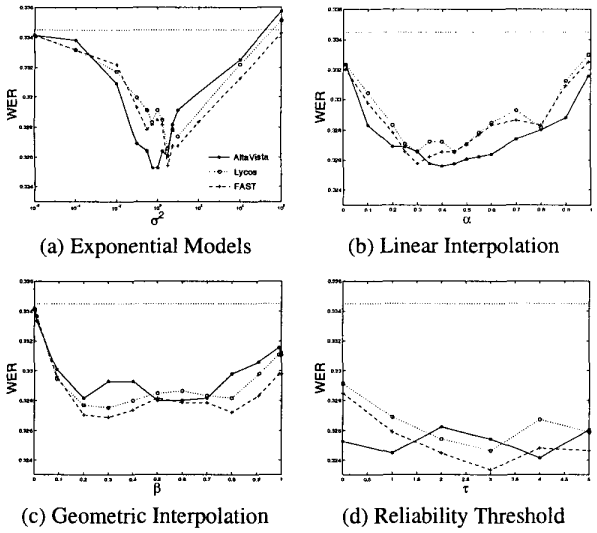
Note that here we have to smooth the web estimates to avoid zeros (which is not a problem in the previous two methods). We add a small positive value ϵ to every web count, a method known as additive smoothing [8]. ϵ is determined by minimizing perplexity when $\beta = 1$. $\beta \in [0,1]$ is the interpolation parameter. If $\beta = 0$, $p^*_G = p_0$. If $\beta = 1$, p^*_G satisfies the smoothed web estimates. A β in between results in an intermediate p^*_G.

4. EXPERIMENTAL RESULT

We randomly selected 200 utterance segments from the TREC-7 Spoken Document Retrieval track data [15] as our test set for this experiment. For each utterance we have its correct transcript and an N-best list with $N = 1000$, i.e. 1000 decoding hypotheses. We performed N-best list rescoring to measure the word error rate (WER) improvement. If we rescore the N-best lists with LM_0 and pick the top hypotheses, the WER is 33.45%. This is our baseline WER. For each N-best list, we queried the unreliable trigrams (and associated bigrams) in it, and computed p^* with the three different interpolation methods. For geometric interpolation we chose $\epsilon = 0.01$ because this minimized perplexity when $\beta = 1$. We then used p^* to rescore the N-best list and calculated the WER of the top hypothesis after rescoring respectively.

Figure 2(a)(b)(c) show the WER with exponential models, linear interpolation, and geometric interpolation respectively, with reliability threshold $\tau = 0$. The three curves stand for different search engines, which turn out to be very similar. The horizontal dashed line is the baseline WER. As the interpolation parameters take some intermediate values, all three models reach their minimum WER respectively. The exponential model reaches minimum WER 32.53% with AltaVista around $\sigma^2 = 1$, the linear interpolation model reaches 32.56% with AltaVista at $\alpha = 0.4$, and the geometric interpolation model reaches 32.69% with FAST when $\beta = 0.3$.

Figure 2(d) shows the effect of reliability threshold τ on WER. The interpolation method used here is the exponential model with Gaussian prior and $\sigma^2 = 1$. We varied τ from 0 to 5. With larger threshold, more trigrams are regarded as unreliable, and hence more web queries had to be issued. There is a slight but significant improvement when we increase τ from 0 to 1. For example, The WER with AltaVista at $\tau = 1$ is 32.45%. Note that LM_0, the language model we are incorporating web estimate into, was built after excluding all singleton trigrams in the corpus. This may explain why $\tau = 1$ is better since trigrams with counts 0 or 1 in the corpus are indeed unreliable: in LM_0 they must backoff to bigram or unigram. Further increase in τ doesn't bring as significant improvement.



(a) Exponential Models (b) Linear Interpolation

(c) Geometric Interpolation (d) Reliability Threshold

Fig. 2. Word Error Rates of web-improved LMs as function of the smoothing parameter for several different interpolation schemes, based on N-best rescoring

More analysis of the WER improvement can be found in [3]. Perplexity cannot be computed because the models are not properly normalized.

5. DISCUSSIONS

We demonstrated that estimates obtained from the web can improve WER. We believe the improvement largely comes from better trigram coverage due to the sheer size of the web, which acts as a 'general English' knowledge source. Interestingly the choice of particular search engine or interpolation method doesn't seem to matter much. Our method has certain advantages. Besides having better N-gram coverage, the content of the web is constantly changing, enabling automatic up-to-date language modeling. However there are also disadvantages. The most severe one is the large number of web queries: We needed 340 queries on average for each utterance. This results in heavy web traffic and load on the search engines, and hence slow rescoring. Another concern is privacy: one may be sending fragments of potentially sensitive utterances to the web. Both problems, however, can be partly solved by using a web-in-a-box setting, i.e. if we have a snapshot of the text content of the whole WWW on local storage. Yet another problem is the lack of focus on specific domains. This might be solved by querying specific domain hosts instead of the whole web, although by doing so the N-gram coverage may deteriorate.

The method proposed in this paper is only one crude way of exploiting the web as a knowledge source for language modeling. One could also look for more complex phenomena, e.g. semantic coherence [16] among content words in a hypothesis. Intuitively, if a hypothesis has content words that 'go with each other', it is more likely than one whose content words seldom appear together in a large training text set. The web search engine approach seems well suited for this purpose. We are currently pursing this direction.

6. REFERENCES

[1] Ronald Rosenfeld, "Two decades of statistical language modeling: Where do we go from here?," *Proceedings of the IEEE*, vol. 88, no. 8, 2000.

[2] Adam Berger and Robert Miller, "Just-in-time language modeling," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, Seattle, Washington, 1998, vol. II, pp. 705–708.

[3] Xiaojin Zhu and Ronald Rosenfeld, "Improving trigram langauge modeling with the world wide web," Tech. Rep. CMU-CS-00-171, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, 2000.

[4] AltaVista. http://www.altavista.com/.

[5] Lycos. http://www.lycos.com/.

[6] FAST Search. http://www.alltheweb.com/.

[7] 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*, Detroit, Michigan, May 1995, vol. I, pp. 181–184.

[8] Stanley F. Chen and Joshua Goodman, "An empirical study of smoothing techniques for language modeling," Tech. Rep. TR-10-98, Harvard University, 1998, Available from `ftp://ftp.das.harvard.edu/techreports/tr-10-98.ps.gz`.

[9] Lin Chase, Ronald Rosenfeld, and Wayne Ward, "Error-responsive modifications to speech recognizers: Negative N-grams," in *Proceedings of the ICSLP*, 1994.

[10] Stanley F. Chen, Douglas Beeferman, and Ronald Rosenfeld, "Evaluation metrics for language models," in *Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop*, 1998, pp. 275–280.

[11] S. Della Pietra, V. Della Pietra, R.L. Mercer, and S. Roukos, "Adaptive language modeling using minimum discriminant estimation," in *Proceedings of the Speech and Natural Language DARPA Workshop*, February 1992.

[12] Adam Berger, Stephen Della Pietra, and Vincent Della Pietra, "A maximum entropy approach to natural language processing," *Computational Linguistics*, vol. 22, no. 1, pp. 39–71, 1996.

[13] J.N. Darroch and D. Ratcliff, "Generalized iterative scaling for log-linear models," *The Annals of Mathematical Statistics*, vol. 43, pp. 1470–1480, 1972.

[14] Stanley F. Chen and Ronald Rosenfeld, "A Gaussian prior for smoothing maximum entropy models," Tech. Rep. CMU-CS-99-108, Computer Science Department, Carnegie Mellon University, Pittsburgh, PA, 1999.

[15] John S. Garofolo, Ellen M. Voorhees, Cedric G. P. Auzanne, Vincent M. Stanford, and Bruce A. Lund, "1998 TREC-7 spoken document retrieval track overview and results," in *Proceedings of TREC-7: The Seventh Text Retrieval Conference*, 1998.

[16] Can Cai, Larry Wasserman, and Roni Rosenfeld, "Exponential language models, logistic regression, and semantic coherence," in *Proceedings of the NIST/DARPA Speech Transcription Workshop*, May 2000.