One Billion Word Benchmark for Measuring Progress in Statistical Language Modeling

Ciprian Chelba, Tomas Mikolov,
Mike Schuster, **Qi Ge**, **Thorsten Brants**
Google, 1600 Amphitheatre Parkway
Mountain View, CA 94043, USA

Phillipp Koehn
University of Edinburgh
10 Crichton Street, Room 4.19
Edinburgh, EH8 9AB, United Kingdom

Abstract

We propose a new benchmark corpus to be used for measuring progress in statistical language modeling. With almost one billion words of training data, we hope this benchmark will be useful to quickly evaluate novel language modeling techniques, and to compare their contribution when combined with other advanced techniques. We show performance of several well-known types of language models, with the best results achieved with a recurrent neural network based language model. The baseline unpruned Kneser-Ney 5-gram model achieves perplexity 74.4. A combination of techniques leads to 37% reduction in perplexity, or 11% reduction in cross-entropy (bits), over that baseline.

The benchmark is available as a code.google.com project; besides the scripts needed to rebuild the training/held-out data, it also makes available log-probability values for each word in each of ten held-out data sets, for each of the baseline n-gram models.

1 Introduction

Statistical language modeling has been applied to a wide range of applications and domains with great success. To name a few, automatic speech recognition, machine translation, spelling correction, touch-screen "soft" keyboards and many natural language processing applications depend on the quality of language models (LMs).

The performance of LMs is determined mostly by several factors: the amount of training data, quality and match of the training data to the test data, and choice of modeling technique for estimation from the data. It is widely accepted that the amount of data, and the ability of a given estimation algorithm to accomodate large amounts of training are very important in providing a solution that competes successfully with the entrenched n-gram LMs. At the same time, scaling up a novel algorithm to a large amount of data involves a large amount of work, and provides a significant barrier to entry for new modeling techniques. By choosing one billion words as the amount of training data we hope to strike a balance between the relevance of the benchmark in the world of abundant data, and the ease with which any researcher can evaluate a given modeling approach. This follows the work of Goodman (2001a), who explored performance of various language modeling techniques when applied to large data sets. One of the key contributions of our work is that the experiments presented in this paper can be reproduced by virtually anybody with an interest in LM, as we use a data set that is freely available on the web.

Another contribution is that we provide strong baseline results with the currently very popular neural network LM (Bengio et al., 2003). This should allow researchers who work on competitive techniques to quickly compare their results to the current state of the art.

The paper is organized as follows: Section 2 describes how the training data was obtained; Section 3 provides a short overview of the language modeling techniques evaluated; finally, Section 4 presents results obtained and Section 5 concludes the paper.

Model	Num. Params [billions]	Training Time [hours]	[CPUs]	Perplexity
Interpolated KN 5-gram, 1.1B n-grams	1.76	3	100	74.4
Katz 5-gram, 1.1B n-grams	1.74	2	100	87.9
Stupid Backoff 5-gram	1.13	0.8	200	98.4
Interpolated KN 5-gram, 15M n-grams	0.03	3	100	246.3
Katz 5-gram, 15M n-grams	0.03	2	100	135.8
Binary MaxEnt 5-gram (n-gram features)	1.13	1	5000	122.7
Binary MaxEnt 5-gram (n-gram + skip-1 features)	1.88	1.25	5000	114.5
Hierarchical Softmax MaxEnt 4-gram	7	3	1	107.3
Recurrent NN-600 + MaxEnt 9-gram	20	168	16	58.3
Recurrent NN-800 + MaxEnt 9-gram	20	336	16	54.4

Table 1: Results on the 1B Word Benchmark test set with various types of language models.

2 Description of the Benchmark Data

In the following experiments, we used text data obtained from the WMT11 website[1]. The data preparation process was performed as follows:

- All training monolingual/English corpora were selected
- Normalization and tokenization was performed using scripts distributed from the WMT11 site, slightly augmented to normalize various UTF-8 variants for common punctuation, e.g. ’
- Duplicate sentences were removed, dropping the number of words from about 2.9 billion to about 0.8 billion (825422561, more exactly, counting sentence boundary markers <S>, <\S>)
- Vocabulary (791888 words including sentence boundary markers <S>, <\S>) was constructed by discarding all words with count below 3
- Words outside of the vocabulary were mapped to <UNK> token, also part of the vocabulary
- Sentence order was randomized, and the data was split into 100 disjoint partitions
- One such partition (1%) of the data was chosen as the held-out set
- The held-out set was then randomly shuffled and split again into 50 disjoint partitions to be used as development/test data
- One such resulting partition (2%, amounting to 158539 words without counting the sentence beginning marker <S> which is never predicted by the language model) of the held-out data were used as test data in our experiments; the remaining partitions are reserved for future experiments
- The out-of-vocabulary (OoV) rate on the test set was 0.30%.

The benchmark is available as a code.google.com project: https://code.google.com/p/1-billion-word-langu Besides the scripts needed to rebuild the training/held-out data, it also makes available log-probability values for each word in each of ten held-out data sets, for each of the baseline n-gram models.

Because the original data had already randomized sentence order, the benchmark is not useful for experiments with models that capture long context dependencies across sentence boundaries.

3 Baseline Language Models

As baselines we chose to use (Katz, 1995), and Interpolated (Kneser and Ney, 1995) (KN) 5-gram LMs, as they are the most prevalent. Since in practice these models are pruned, often quite aggressivley, we also illustrate the negative effect of (Stolcke, 1998) entropy pruning on both models, similar to (Chelba et al., 2010). In particular KN smoothing degrades much more rapidly than Katz,

[1] http://statmt.org/wmt11/training-monolingual.tgz

calling for a discerning choice in a given application.

4 Advanced Language Modeling Techniques

The number of advanced techniques for statistical language modeling is very large. It is out of scope of this paper to provide their detailed description, but we mention some of the most popular ones:

- N-grams with Modified Kneser-Ney smoothing (Chen and Goodman, 1996)
- Cache (Jelinek et al., 1991)
- Class-based (Brown et al., 1992)
- Maximum entropy (Rosenfeld, 1994)
- Structured (Chelba and Jelinek, 2000)
- Neural net based (Bengio et al., 2003)
- Discriminative (Roark et al., 2004)
- Random forest (Xu, 2005)
- Bayesian (Teh, 2006)

Below, we provide a short description of models that we used in our comparison using the benchmark data.

4.1 Normalized Stupid Backoff

The Stupid Backoff LM was proposed in (Brants et al., 2007) as a simplified version of backoff LM, suited to client-server architectures in a distributed computing environment.It does not apply any discounting to relative frequencies, and it uses a single backoff weight instead of context-dependent backoff weights. As a result, the Stupid Backoff model does not generate normalized probabilities. For the purpose of computing perplexity as reported in Table 1, values output by the model were normalized over the entire LM vocabulary.

4.2 Binary Maximum Entropy Language Model

The Binary MaxEnt model was proposed in (Xu et al., 2011) and aims to avoid the expensive probability normalization during training by using independent binary predictors. Each predictor is trained using all the positive examples, but the negative examples are dramatically down-sampled. This type of model is attractive for parallel training, thus we explored its performance further.

We trained two models with a sampling rate of 0.001 for negative examples, one uses n-gram features only and the other uses n-gram and skip-1 n-gram features. We separated the phases of generating negative examples and tuning model parameters such that the output of the first phase can be shared by two models. The generation of the negative examples took 9.25 hours using 500 machines, while tuning the parameters took 1 hour and 1.25 hours for two models using 5000 machines.

4.3 Maximum Entropy Language Model with Hierarchical Softmax

Another option to reduce training complexity of the MaxEnt models is to use a hierarchical softmax (Goodman, 2001b; Morin and Bengio, 2005). The idea is to estimate probabilities of groups of words, like in a class based model – only the classes that contain the positive examples need to be evaluated. In our case, we explored a binary Huffman tree representation of the vocabulary, such that evaluation of frequent words takes less time. The idea of using frequencies of words for a hierarchical softmax was presented previously in (Mikolov et al., 2011a).

4.4 Recurrent Neural Network Language Model

The Recurrent Neural Network (RNN) based LM have recently achieved outstanding performance on a number of tasks (Mikolov, 2012). It was shown that RNN LM significantly outperforms many other language modeling techniques on the Penn Treebank data set (Mikolov et al., 2011b). It was also shown that RNN models scale very well to data sets with hundreds of millions of words (Mikolov et al., 2011c), although the reported training times for the largest models were in the order of weeks.

We cut down training times by a factor of 20-50 for large problems using a number of techniques, which allow RNN training in typically 1-10 days with billions of words, $> 1M$ vocabularies and up to 20B parameters on a single standard machine without GPUs.

These techniques were in order of importance:

Model	Perplexity	Interpolation Weight
Interpolated KN 5-gram, 1.1B n-grams	74.4	0.16
Combination of 8 RNN-MaxEnt models	47.9	0.84
All models	**46.8**	—

Table 2: Model combination on the 1B Word Benchmark test set. The weights were tuned to minimize perplexity on held-out data. Only models whose interpolation weight is non-zero are listed.

a) Parallelization of training across available CPU threads, b) Making use of SIMD instructions where possible, c) Reducing number of output parameters by 90%, d) Running a Maximum Entropy model in parallel to the RNN. Because of space limitations we cannot describe the exact details of the speed-ups here – they will be reported in an upcoming paper.

We trained several models with 600 and 800 recurrent neurons (Table 1) using regular SGD with a learning rate of 0.05 to 0.001 using 10 iterations over the data. The MaxEnt models running in parallel to the RNN capture a history of 9 previous words, and the 800 neuron model uses as additional features the previous 15 words independently of order. While training times approach 2 weeks for the most complex model, slightly worse models can be trained in a few days. Note that we didn't optimize for model size nor training speed, only test performance.

5 Results

5.1 Performance of Individual Models

Results achieved on the benchmark data with various types of LM are reported in Table 1. The test set consists of 152360 words (5976 sentences). We focused on minimizing the perplexity when choosing hyper-parameters, however we also report the time required to train the models. Training times are not necessarily comparable as they depend on the underlying implementation. Mapreduces can potentially process larger data sets than single-machine implementations, but come with a large overhead of communication and file I/O. Discussing details of the implementations is outside the scope as this paper.

5.2 Model Combination

The best perplexity results were achieved by linearly interpolating together probabilities from all models. However, only some models had significant weight in the combination; the weights were tuned on the held-out data. As can be seen in Table 2, the best perplexity is about 37% lower than the baseline - the modified Kneser-Ney smoothed 5-gram model with no count cutoffs. This corresponds to about 11% reduction of cross-entropy (bits).

6 Conclusion

We introduced a new data set for measuring research progress in statistical language modeling. The benchmark data set is based on resources that are freely available on the web, thus fair comparison of various techniques is easily possible. The importance of such effort is unquestionable: it has been seen many times in the history of research that significant progress can be achieved when various approaches are measurable, reproducible, and the barrier to entry is low.

The choice of approximately one billion words might seem somewhat restrictive. Indeed, it can be hardly expected that new techniques will be immediately competitive on a large data set. Computationally expensive techniques can still be compared using for example just the first or the first 10 partitions of this new dataset, corresponding to approx. 10 million and 100 million words. However, to achieve impactful results in domains such as speech recognition and machine translation, the language modeling techniques need to be scaled up to large data sets.

Another contribution of this paper is the comparison of a few novel modeling approaches when trained on a large data set. As far as we know, we were able to train the largest recurrent neural network language model ever reported. The performance gain is very promising; the perplexity reduction of 37% is large enough to let us hope for significant improvements in various applications.

In the future, we would like to encourage other researchers to participate in our efforts to make language modeling research more transparent. This

would greatly help to transfer the latest discoveries into real-world applications. In the spirit of a benchmark our first goal was to achieve the best possible test perplexities regardless of model sizes or training time. However, this was a relatively limited collaborative effort, and some well known techniques are still missing. We invite other researchers to complete the picture by evaluating new, and well-known techniques on this corpus. Ideally the benchmark would also contain ASR or SMT lattices/N-best lists, such that one can evaluate application specific performance as well.

References

[Bengio et al.2003] Y. Bengio, R. Ducharme, and P. Vincent. 2003. *A neural probabilistic language model*. Journal of Machine Learning Research, 3:1137-1155.

[Brants et al.2007] T. Brants, A. C. Popat, P. Xu, F. J. Och, and J. Dean. 2007. *Large language models in machine translation*. In Proceedings of EMNLP.

[Brown et al.1992] P. F. Brown, P. V. deSouza, R. L. Mercer, V. J. Della Pietra, and J. C. Lai. 1992. *Class-Based n-gram Models of Natural Language*. Computational Linguistics, 18, 467-479.

[Elman1990] J. Elman. 1990. *Finding Structure in Time*. Cognitive Science, 14, 179-211.

[Emami2006] A. Emami. 2006. *A Neural Syntactic Language Model*. Ph.D. thesis, Johns Hopkins University.

[Goodman2001a] J. T. Goodman. 2001a. *A bit of progress in language modeling, extended version*. Technical report MSR-TR-2001-72.

[Goodman2001b] J. T. Goodman. 2001b. *Classes for fast maximum entropy training*. In Proceedings of ICASSP.

[Jelinek et al.1991] F. Jelinek, B. Merialdo, S. Roukos, and M. Strauss. 1991. *A Dynamic Language Model for Speech Recognition*. In Proceedings of the DARPA Workshop on Speech and Natural Language.

[Chelba and Jelinek2000] C. Chelba and F. Jelinek. 2000. *Structured language modeling*. Computer Speech & Language.

[Chelba et al.2010] C. Chelba, T. Brants, W. Neveitt, and P. Xu. 2010. *Study on Interaction between Entropy Pruning and Kneser-Ney Smoothing*. In Proceedings of Interspeech.

[Chen and Goodman1996] S. F. Chen and J. T. Goodman. 1996. *An empirical study of smoothing techniques for language modeling*. In Proceedings of ACL.

[Chen2009] S. F. Chen. 2009. *Shrinking exponential language models*. In Proceedings of NAACL-HLT.

[Katz1995] S. Katz. 1987. *Estimation of probabilities from sparse data for the language model component of a speech recognizer*. In IEEE Transactions on Acoustics, Speech and Signal Processing.

[Kneser and Ney1995] R. Kneser and H. Ney. 1995. *Improved Backing-Off For M-Gram Language Modeling*. In Proceedings of ICASSP.

[Mikolov et al.2010] T. Mikolov, M. Karafiát, L. Burget, J. Černocký, and S. Khudanpur. 2010. *Recurrent neural network based language model*. In Proceedings of Interspeech.

[Mikolov et al.2011a] T. Mikolov, S. Kombrink, L. Burget, J. Černocký, and S. Khudanpur. 2011. *Extensions ofl Recurrent Neural Network Language Model*. In Proceedings of ICASSP.

[Mikolov et al.2011b] T. Mikolov, A. Deoras, S. Kombrink, L. Burget, and J. Černocký. 2011a. *Empirical Evaluation and Combination of Advanced Language Modeling Techniques*. In Proceedings of Interspeech.

[Mikolov et al.2011c] T. Mikolov, A. Deoras, D. Povey, L. Burget, and J. Černocký. 2011b. *Strategies for Training Large Scale Neural Network Language Models*. In Proceedings of ASRU.

[Mikolov2012] T. Mikolov. 2012. *Statistical Language Models based on Neural Networks*. Ph.D. thesis, Brno University of Technology.

[Mnih and Hinton2007] A. Mnih and G. Hinton. 2007. *Three new graphical models for statistical language modelling*. In Proceedings of ICML.

[Morin and Bengio2005] F. Morin and Y. Bengio. 2005. *Hierarchical Probabilistic Neural Network Language Model*. In Proceedings of AISTATS.

[Roark et al.2004] B. Roark, M. Saralar, M. Collins, and M. Johnson. 2004. *Discriminative language modeling with conditional random fields and the perceptron algorithm*. In Proceedings of ACL.

[Rosenfeld1994] R. Rosenfeld. 1994. *Adaptive Statistical Language Modeling: A Maximum Entropy Approach*. Ph.D. thesis, Carnegie Mellon University.

[Rumelhart et al.1986] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. 1986. *Learning internal representations by back-propagating errors*. Nature, 323:533-536.

[Schwenk2007] H. Schwenk. 2007. *Continuous space language models*. Computer Speech and Language, vol. 21.

[Stolcke1998] A. Stolcke. 1998. *Entropy-based Pruning of Back-off Language Models*. In Proceedings of News Transcription and Understanding Workshop.

[Sundermeyer et al.2012] M. Sundermeyer, R. Schluter, and H. Ney. 2012. *LSTM Neural Networks for Language Modeling*. In Proceedings of Interspeech.

[Teh2006] Y. W. Teh. 2006. *A hierarchical Bayesian language model based on PitmanYor processes*. In Proceedings of Coling/ACL.

[Wu et al.2012] Y. Wu, H. Yamamoto, X. Lu, S. Matsuda, C. Hori, and H. Kashioka. 2012. *Factored Recurrent Neural Network Language Model in TED Lecture Transcription*. In Proceedings of IWSLT.

[Xu2005] Peng Xu. 2005. *Random forests and the data sparseness problem in language modeling*. Ph.D. thesis, Johns Hopkins University.

[Xu et al.2011] Puyang Xu, A. Gunawardana, and S. Khudanpur. 2011. *Efficient Subsampling for Training Complex Language Models*. In Proceedings of EMNLP.

[Zweig and Makarychev2013] G. Zweig and K. Makarychev. 2013. *Speed Regularization and Optimality in Word Classing*. In Proceedings of ICASSP.