INVESTIGATION OF SAMPLING TECHNIQUES FOR MAXIMUM ENTROPY LANGUAGE MODELING TRAINING

Xie Chen, Jun Zhang, Tasos Anastasakos, Fil Alleva

Microsoft, USA

{xieche, junzh, tasos.anastasakos, fil}@microsoft.com

ABSTRACT

Maximum entropy language models (MaxEnt LMs) are log-linear models which are able to incorporate various hand-crafted features and non-linguistic information. Standard MaxEnt LMs are computationally heavy for tasks with a large vocabulary size due to the expensive normalization computation in the denominator. To address this issue, most recent works on MaxEnt LMs have used class based MaxEnt LMs. However, the performance of class based MaxEnt LMs might be sensitive to word clustering and it is also time-consuming to generate high-quality word classes. Motivated by the recent success of sampling techniques in accelerating the training of neural network language models, in this paper, three widely used sampling techniques, importance sampling, noise contrastive estimation (NCE) and sampled softmax, are investigated for the MaxEnt LM training. Experimental results on the Google One Billion corpus and an internal speech recognition system demonstrate the effectiveness of sampled softmax and NCE on MaxEnt LM training. However, importance sampling is not effective for MaxEnt LM training despite its similarity to sampled softmax. To our knowledge, this is the first work applying sampling techniques on MaxEnt LM training.

Index Terms— Maximum entropy language model, importance sampling, noise contrastive estimation, sampled softmax

1. INTRODUCTION

The language model is an important component in a range of applications including automatic speech recognition (ASR). N-gram and neural network language models are the two most popular models used in modern ASR systems. However, it is difficult to incorporate various informative features, such as skip-gram, trigger words, time-of-day and geographic location.

There is another type of powerful language model, maximum entropy language models (MaxEnt LMs), which are able to incorporate hand-crafted features and non-linguistic information into a unified log-linear framework. Standard n-gram features and other features, such as skip-grams and geographic location, are easily taken into account in MaxEnt LMs.

These days, it is common to use billions of words for the training of language models. It is crucial to improve the scalability of language model training for large amounts of data. The standard MaxEnt LMs shown in Equation 1 require summing over the whole vocabulary. This is computationally expensive for large vocabulary tasks. Furthermore, the computation in MaxEnt LMs is not as structured as neural network language models, where GPUs can be used to optimize the matrix related operations. However, for the training of MaxEnt LMs, CPUs are normally used. To improve the computational efficiency, previous approaches used class based MaxEnt LMs when large data corpora were processed. Unfortunately, the model performance can be affected by the choice of word classes [1]. In addition, high-quality word clustering on scalable data is time-consuming. For example, in [2], almost one third of total training time was spent on word clustering.

Motivated by the recent success of sampling techniques in the training of neural network language models [3, 4, 5, 6], we aim to explore the use of sampling techniques for MaxEnt LM training. Three widely used sampling techniques, importance sampling [7], noise contrastive estimation (NCE) [8, 4] and sampled softmax [5], were applied to the MaxEnt LM training. To our knowledge, this is the first published work investigating sampling techniques for the training of MaxEnt LMs. Our initial experiments demonstrated that sampled softmax and NCE works well for training of MaxEnt LMs. In contrast, importance sampling is not suitable for the training of MaxEnt LMs despite its similarity to sampled softmax.

This paper is organized as follows, Section 2 gives a brief review of maximum entropy language models. The three sampling algorithms investigated in this paper are described in Section 3, followed by the implementation details of MaxEnt LMs on multiple machines. Experimental results are reported in Section 5. Finally, the conclusion and discussion of future work can be found in Section 6.

2. REVIEW OF MAXIMUM ENTROPY LMS

MaxEnt LMs [9] were originally derived from information theory by maximizing the entropy of the model distribution given some constraints observed in the training data. It can be proven that the unique solution of MaxEnt LMs can be expressed in the form of a log-linear model, and the probability of word w_d with history h can be computed as,

$$P(w_d|h) = \frac{\exp(\sum_{k=1}^{|F|} \lambda_{f_k(w_d,h)} f_k(w_d,h))}{\sum_{j\in\mathcal{V}} \exp(\sum_{k=1}^{|F|} \lambda_{f_k(w_j,h)} f_k(w_j,h))} \quad (1)$$

where $f(w,h)$ defines the map from any hand-crafted feature extracted from the target word w and its history h to $\{0,1\}$ binary value. Its value is 1 when some property of (w,h) defined by $f(w,h)$ is true. For example, this feature can be n-gram or skip-gram features. $\lambda_{f_k(w,h)}$ is the model parameter of the corresponding feature. $|F|$ is the total number of MaxEnt feature types. The model parameters of MaxEnt LMs can be estimated using generalized iterative scale (GIS) [10] or stochastic gradient descent (SGD). According to the literature, when the size of training data is small, it is more common to choose GIS as it is faster to converge [11, 12]. In a large corpus, it is difficult to store the statistics of the whole data and SGD is used instead [13, 14, 15].

The computation of MaxEnt LMs can be viewed as the softmax function of $s(w,h) = \sum_{k=1}^{|F|} \lambda_{f_k(w,h)} f_k(w,h))$. In order to get a valid probability, Equation 1 requires computing the denominator over the whole vocabulary. It is computationally heavy for tasks with a large vocabulary. The authors in [16] proposed an efficient way for training MaxEnt LMs with n-gram features by utilizing the hierarchical structure in the different orders of n-gram features. However, this technique cannot be applied to other features such as skip-gram and non-linguistic features. In order to address this issue, recent work has introduced word classes to improve the training efficiency [11] and performance [1] of MaxEnt LMs. In MaxEnt LMs with word classes, the word probability can be computed as,

$$P(w_i|h) = P(C(w_i)|h)P(w_i|C(w_i),h) \tag{2}$$

It can be seen that the word probability is factored into the product of two separate probabilities: one is the class probability and the other is the word probability within class. These two probabilities can be modeled by two MaxEnt LMs respectively. The MaxEnt LM $P(C(w_i)|h)$ for class only needs to sum over classes C and the MaxEnt LM for word $P(w_i|C(w_i),h)$ computes the sum over the class $C(w_i)$ which w_i belongs to. The computation is significantly smaller than that of word based MaxEnt LMs. Ideally, the computation can be accelerated up to $\sqrt{|\mathcal{V}|}/2$ times compared to the word based MaxEnt LM shown in Equation 1, when $\sqrt{|\mathcal{V}|}$ word classes are clustered and each class contains $\sqrt{|\mathcal{V}|}$ words.

There have been many research efforts in MaxEnt LMs during in the last several years due to their flexibility of incorporating various information. Model M [1, 17] is a typical MaxEnt LM. It presented consistent performance improvement over the standard backoff n-gram LM using standard n-gram features. L1 and L2 regularization were applied with specific regularization coefficients. However, the MaxEnt LMs in Model M were normally trained on a relatively small corpora (less than 100M words). There are also a series of papers from Google working on scalable MaxEnt LMs [13, 14, 15, 2], where MaxEnt LMs were trained on the giga or even tera words scale. It was found that the model converged well without any regularization. The models were trained on multiple machines and a simple model average was applied after a specific number of updates. Our approach is more similar to Google's work by training MaxEnt LMs on a large amount of data with multiple machines.

3. SAMPLING TECHNIQUES FOR MAXENT LM TRAINING

Sampling techniques have achieved big success in the neural network language model training [3, 4, 5, 6] [1]. By applying sampling techniques, only a small fraction of noise samples and target samples, are needed for the likelihood estimation and gradient computation, instead of the whole vocabulary. This improves the computational efficiency significantly, especially for tasks with a large vocabulary. MaxEnt LMs encountered the same issue in the calculation of the denominator as is shown in Equation 1. The MaxEnt LM can be viewed as a neural network language model with direct connection from various MaxEnt features to the target word [18]. However, in neural network language models, the history is represented as a continuous hidden vector, and this hidden vector representation is shared by all words in the output layer. In contrast, MaxEnt LMs use discrete feature representation in the input layer and the connection between the input features and words in the output layer is very sparse, depending on whether the specific MaxEnt feature $f_k(w,h)$ exists or not. Therefore, it is still unknown whether sampled techniques are suitable for the training of MaxEnt LMs, especially for NCE, which implicitly constrains the variance of normalization term to be a constant. We aim to investigate sampling techniques for MaxEnt LMs in this paper. Three well-known sampling algorithms were explored, which are importance sampling (IS), noise contrastive estimation (NCE) and sampled softmax (SS).

[1] It is also known as candidate sampling in literatures: https://www.tensorflow.org/extras/candidate_sampling.pdf

3.1. Importance Sampling

Importance sampling is a general technique to estimate properties of a particular distribution when it is difficult to draw samples from the original distribution. Samples from another noise distribution are used for approximation. In previous work, importance sampling (IS) was applied for the training of neural network language models (NNLMs) to reduce computation in the output layer [7, 19]. In MaxEnt LMs, the same issue exists in the computation of the denominator in Equation 1. Hence, importance sampling can be applied. For the maximum likelihood based objective function, the gradient of MaxEnt LMs can be written as

$$\frac{\partial \log P(w_d|h)}{\partial \theta} = \frac{\partial s(w_d,h)}{\partial \theta} - \sum_{w \in \mathcal{V}} P(w|h) \frac{\partial s(w,h)}{\partial \theta} \tag{3}$$

where $\frac{\partial s(w,h)}{\partial \theta}$ is the gradient of logit $s(w,h)$ with respect to model parameters θ. The second term in the above equation requires summing over the whole vocabulary. Importance sampling can be applied to approximate the sum in the second term.

$$\frac{\partial \log P(w_d|h)}{\partial \theta} \approx \frac{\partial s(w_d,h)}{\partial \theta} - \frac{1}{K} \sum_{k=1}^{K} \frac{P(w_k|h)}{Q(w_k|h)} \frac{\partial s(w_k,h)}{\partial \theta} \tag{4}$$

where $Q(w|h)$ is the noise distribution and K is the number of samples drawn from $Q(w|h)$. Normally $Q(w|h)$ is chosen as history independent, i.e. $Q(w|h) = Q(w)$, such as the unigram distribution. With this approximation, the gradient computation only involves the target word and K sampled words. However, Equation 4 still needs to use the word probability $P(w|h)$ for the gradient computation, which is

$$P(w|h) = \frac{\exp(s(w,h))}{Z(h)} \tag{5}$$

The normalization term $Z(h)$ is computed by summing over the whole vocabulary. Again, importance sampling is applied to approximate the computation of normalization term $Z(h)$,

$$Z(h) = \sum_{w \in \mathcal{V}} \exp(s(w,h)) \approx \frac{1}{K} \sum_{k=1}^{K} \frac{\exp(s(w_k,h))}{Q(w_k)} \tag{6}$$

By applying Equations 5 and 6 into Equation 4, the gradient of IS can be written as

$$\frac{\partial \log P(w_d|h)}{\partial \theta} = \frac{\partial s(w_d,h)}{\partial \theta} - \sum_{k=1}^{K} \frac{\exp(r(w_k,h))}{\sum_{j=1}^{K} \exp(r(w_j,h))} \frac{\partial s(w_k,h)}{\partial \theta} \tag{7}$$

where $r(w,h) = \log(\frac{\exp(s(w,h))}{Q(w)})$. By applying importance sampling, the gradient of MaxEnt LMs is only related to the target word and K sampled words from the distribution $Q(w)$, instead of the vocabulary size $|\mathcal{V}|$.

It is worth noting that, for importance sampling, the objective function is not changed. The importance sampling is applied to approximate the gradient computation of maximum likelihood objective function.

3.2. Noise Contrastive Estimation

Noise contrastive estimation (NCE) is an alternative objective function based on sampling to speedup both training and evaluation. NCE aims to discriminate samples generated from the noise distribution and the data distribution. Its objective function can be written as

$$\mathcal{J}_{nce} = \log \frac{P(w_d|h)}{P(w_d|h) + KQ(w_d)} + \sum_{k=1}^{K} \log \frac{KP(w_k|h)}{P(w_k|h) + KQ(w_k)} \tag{8}$$

The first term is the likelihood of target word w_d generated by model distribution and the second term is the likelihood of K noise words generated by noise distribution. K is the ratio of noise samples to target samples, which is set empirically. During NCE training, the model probability $P(w|h)$ can be approximated by using the unnormalized probability $\exp(s(w,h))$ as the normalization term $Z(h)$ is constrained to be constant. Therefore, the NCE objective function can be rewritten as

$$\begin{aligned} \mathcal{J}_{nce} &= \log \frac{\exp(g(w_d,h))}{1+\exp(g(w_d,h))} + \sum_{k=1}^{K} \log \frac{1}{1+\exp(g(w_k,h))} \\ &= \log \sigma(g(w_d,h)) + \sum_{k=1}^{K} \log(1-\sigma(g(w_k,h))) \end{aligned} \tag{9}$$

where $g(w,h) = \log(\frac{\exp(s(w,h)}{KQ(w)})$ and σ denotes the sigmoid function. The NCE objective function can be viewed as applying the binary classifier on each target and noise samples. At test time, unnormalized probabilities can be applied, as the normalization term can be approximated as a constant.

3.3. Sampled Softmax

Sampled softmax [5] is another sampling based objective function derived from NCE. As discussed above, NCE can be viewed as applying a binary classifier between target and noise samples using the sigmoid function. The softmax function is a natural extension of the sigmoid function from binary to multi-class classification. The objective function of sampled softmax can be written as,

$$\mathcal{J}_{ss} = \log \frac{\exp(g(w_0,h))}{\sum_{k=0}^{K} \exp(g(w_k,h))} = \log \frac{\exp(r(w_0,h))}{\sum_{k=0}^{K} \exp(r(w_k,h))} \tag{10}$$

For notation clarity, w_0 is the target word w_d , and $w_1, ..., w_K$ are K noise samples. Note that $g(w,h)$ is $\log(\frac{\exp(s(w,h))}{KQ(w)})$ used in NCE, and $r(w,h)$ is $\log(\frac{\exp(s(w,h))}{Q(w)})$ in IS. Hence softmax of $g(w,h)$ and $r(w,h)$ are the same. The gradient of the sampled softmax objective function can be computed by

$$\frac{\partial \mathcal{J}_{ss}}{\partial \theta} = \frac{\partial s(w_0,h)}{\partial \theta} - \sum_{k=0}^{K} \frac{\exp(r(w_k,h))}{\sum_{j=0}^{K} \exp(r(w_j,h)} \frac{\partial s(w_k,h)}{\partial \theta} \tag{11}$$

It can be seen that the gradient of importance sampling in Equation 7 is very similar to that of sampled softmax in Equation 11. The only difference is that the term $r(w_0,h)$ for the target word is used in the softmax (the second term) in sampled softmax.

In [5], the authors denoted sampled softmax as importance sampling. However, they are not equivalent according to the above discussion. Importance sampling is used to approximate the gradient computation in maximum likelihood. It does not change the objective function. On the other hand, sampled softmax defines a new objective function, which is an extension of NCE over multi-class classification. The experiments presented in Section 5 also shows that the behavior of importance sampling and sampled softmax are very different. Nevertheless, these two sampling techniques indeed have some connection. In importance sampling, if we always use the target word as one of the noise samples, the resulted gradient computation is mathematically the same as sampled softmax.

When sampling techniques are used for the training of MaxEnt LMs, the likelihood evaluation and gradient update computation is constant proportional to the K noise samples and independent of the vocabulary size $|\mathcal{V}|$. For a typical choice $K = 100$ noise words, the sampling techniques obtain the speedup factor of $|\mathcal{V}|/K$ which is considerably greater than $2\sqrt{|\mathcal{V}|}$ for class based MaxEnt LMs of large vocabulary size (e.g. $|\mathcal{V}| = 100K$ words).

4. IMPLEMENTATIONS OF MAXENT LMS

The main purpose of this paper is to investigate the use of sampling techniques for MaxEnt LM training. We limit the MaxEnt features to be n-gram and skip-gram features extracted from the target words and its previous 4 words. Features such as backoff features [13] and geographic features [20] are not used here. The CPU cluster, Cosmos [21]; was used to compute the frequency of various MaxEnt features and to prune the infrequent features efficiently. Differing from neural network language models, it is not easy to use GPUs for the training of MaxEnt LMs as the computation is not uniformly structured. Most computations happened in MaxEnt LMs are feature extraction, look-up operations in hash tables and the softmax calculation. For the sake of efficiency, the training of MaxEnt LMs was implemented with C++ on CPUs. Stochastic gradient descent (SGD) was applied to optimize the model parameters of MaxEnt LMs without any regularization. The MaxEnt LM was initialized using the log unigram probability derived from the training data to get a better initialization.

MaxEnt LMs are trained in a distributed fashion on multiple machines. In this paper, we used 30-50 machines (each machine has 16 cores) to train the MaxEnt models. Message Passing Interface (MPI) was used for communications between machines. The data was split evenly according to the number of machines. The iterative parameter mixtures (IPM) [22] algorithm was used to synchronize the training across machines. Model parameters on each machine were trained separately over the partition of data and averaged at the end of each epoch. The parallel training of MaxEnt LMs was similar to [13]. In each machine, multi-threading was used for minibatch based training, and in each thread, one training sample was processed within the minibatch. The minibatch size is set to 16 in this paper.

In this work, the unigram distribution is used as the noise distribution $Q(w)$. In neural network language models, in order to improve the convergence of NCE training [6, 23], the noise samples within the same minibatch are normally shared. This is computationally efficient in neural networks on GPUs. However, in MaxEnt LMs, each training word has different contexts and different MaxEnt features are extracted. Sharing noise samples won't introduce similar benefits as neural network language models. Therefore, the noise samples are not shared. As a result, sampling from unigram distribution is computationally heavy as each predicted word requires hundreds of different noise samples. In order to address this issue, we take the training file as the samples generated by the unigram distribution. Sampling noise samples from unigram distribution can be reduced to reading a specified number of words from the training file. This is very efficient. In this paper, the number of noise samples is fixed to be 100 for each training word.

5. EXPERIMENTS

In this section, the sampling techniques discussed in Section 3 are investigated for the training of MaxEnt LMs on two corpora. The first corpus is the public Google One Billion corpus [24], where the PPLs are reported and compared to demonstrate the effectiveness of different sampling techniques. The second corpus is an internal meeting transcription corpus and the performances of MaxEnt LMs on speech recognition were reported.

5.1. Experiments on Google One Billion corpus

The model parameters were pruned according to the frequencies of MaxEnt features. Two types of MaxEnt features were used to train the MaxEnt LMs, which are n-gram and skip-gram features. Only the target word and its previous 4 words were used to extract MaxEnt features. Table 1 shows the perplexities (PPLs) of MaxEnt LMs with different features (n-gram and skip-gram features) and different cutoff thresholds (at least 2 and 3 times). The MaxEnt LM was trained on 30 machines and the training speed was about 1M words per second. It takes 12 minutes for one epoch on average. The model converged in about 40 epochs. Importance sampling is very unstable and the PPL diverges after the first one or two epochs[2], which is similar to the observation reported in [7]. As discussed in Section 3, sampled softmax is quite similar to importance sampling. However, SS was much more stable and converged well. NCE gave comparable PPLs to sampled softmax. We did not implement the class based MaxEnt LMs in this paper. The PPL results of class based MaxEnt LMs on the same corpus can be found in [15] and used as references here. According to the Figure 1 of [15], the PPL of the MaxEnt LM with 100M parameters is about 105 and for 200M parameters it is about 98. It can be seen that, given the same number of parameters, the PPL is comparable to the class based MaxEnt LMs. It indicates that NCE and sampled softmax work well for the training of MaxEnt LMs. It is worth noting that MaxEnt LMs are convex models and have globally optimal solution. However, in NCE training, the normalization term is constrained to be a constant. So the NCE training probably converged to sub-optimal solution with this constrain. According to the results, the PPL of NCE is slightly worse than sampled softmax. However, the variance of the normalization term of NCE training is much smaller than that of that of sampled softmax. This allows the unnormalized probability to be used at test time for NCE training.

Freq. prune	MaxEnt feature	#Param	Sampling Method		
			IS	NCE	SS
2	n-gram	101M	N/A	109.2	103.2
1		204M		100.8	91.8
2	+skip-gram	404M	N/A	91.0	89.3
1		815M		84.9	83.5

Table 1. PPL results on Google One Billion corpus with different sampling techniques using n-gram and skip-gram features

The third and fourth lines in Table 1 show the PPL results of MaxEnt LMs with skip-gram features as well as n-gram features. The total number of parameters increases significantly, with slight but consistent PPL improvements.

[2]when the number of samples increases to more than 1000, the model is able to converge to a sensible PPL, but still higher than that of SS and NCE.

5.2. Experiments on Meeting data

An internal meeting transcription system was built for speech recognition experiments. A modified Kneser-Ney smoothed 5-gram LM with 37M n-grams was used for the first pass decoding to generate lattices. The vocabulary size for the 5-gram LM is about 1 Million. MaxEnt LMs were applied in the second pass rescoring on the 100-best. The MaxEnt LM was trained on about 3 billion words and 245k most frequent words were chosen as vocabulary for the MaxEnt LM. The probability of the MaxEnt LM was renormalized over the 1M vocabulary size of 5-gram LM by simply distributing the probability of "<unk>" evenly over all OOV words. The size of model parameters was pruned to 300M. Log-linear interpolation was applied to combine the MaxEnt and 5-gram LMs. As discussed in the previous experiment, importance sampling did not work for the training of MaxEnt LMs. Hence, only MaxEnt LMs with NCE and sampled softmax were trained.

Table 2 presents the WER results of sampled softmax and NCE. For the sampled softmax, the accurate probability was computed via explicitly normalization over the whole vocabulary. In this way, the n-best rescoring is time-consuming. While for NCE based models, the unnormalized probability was applied, which is very efficient to compute. According to Table 2, it can be seen that the WER can be reduced by 0.2% absolutely from MaxEnt LMs with n-gram features only and 0.4% absolutely from MaxEnt LMs with additional skip-gram features. NCE gave comparable WER performances to sampled softmax but with a much faster test speed. It demonstrates the effectiveness of the sampling techniques on MaxEnt LM training.

LM	5-gram LM	MaxEnt LM			
		n-gram fea.		+skip-gram fea.	
		NCE	SS	NCE	SS
WER	20.72	20.54	20.46	20.31	20.29

Table 2. WER results of MaxEnt LMs trained with NCE and SS, using n-gram and skip-gram features on meeting data

6. CONCLUSIONS

MaxEnt LMs are powerful language models able to incorporate various hand-crafted features. In order to train MaxEnt LMs efficiently, most previous work adopted class based MaxEnt LMs. In this paper, we explored the use of sampling techniques on the training of maximum entropy based language models (MaxEnt LMs). Three sampling techniques, importance sampling, noise contrastive estimation (NCE) and sampled softmax, were investigated. Experiments on the Google One Billion corpus and an internal ASR task demonstrated the effectiveness of sampled softmax and NCE for MaxEnt LM training. However, importance sampling is not suitable for MaxEnt LMs training. To our knowledge, this is the first work to apply sampling techniques for MaxEnt LMs training. In the future, we want to explore the use of sampling techniques for MaxEnt LMs with non-linguistic features, such as geographic features.

7. ACKNOWLEDGEMENT

The authors would like to thanks to Shawn Chang and Andreas Stolcke for fruitful discussions, also want to thanks to William Gale and Changliang Liu for providing the LM training corpus and pipelines for the ASR system.

8. REFERENCES

[1] Stanley F Chen and Stephen M Chu, "Enhanced word classing for model m," in *Eleventh Annual Conference of the International Speech Communication Association*, 2010.

[2] Ciprian Chelba, Diamantino Caseiro, and Fadi Biadsy, "Sparse non-negative matrix language modeling: Maximum entropy flexibility on the cheap," *Proc. Interspeech 2017*, pp. 2725–2729, 2017.

[3] Andriy Mnih and Yee Whye Teh, "A fast and simple algorithm for training neural probabilistic language models," *arXiv preprint arXiv:1206.6426*, 2012.

[4] Xie Chen, Xunying Liu, Mark John Gales, and Philip Charles Woodland, "Recurrent neural network language model training with noise contrastive estimation for speech recognition," 2015.

[5] Rafal Jozefowicz, Oriol Vinyals, Mike Schuster, Noam Shazeer, and Yonghui Wu, "Exploring the limits of language modeling," *arXiv preprint arXiv:1602.02410*, 2016.

[6] Youssef Oualil and Dietrich Klakow, "A batch noise contrastive estimation approach for training large vocabulary language models," *arXiv preprint arXiv:1708.05997*, 2017.

[7] Yoshua Bengio and Jean-Sébastien Senécal, "Adaptive importance sampling to accelerate training of a neural probabilistic language model," *IEEE Transactions on Neural Networks*, vol. 19, no. 4, pp. 713–722, 2008.

[8] Michael Gutmann and Aapo Hyvärinen, "Noise-contrastive estimation: A new estimation principle for unnormalized statistical models," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 2010, pp. 297–304.

[9] Roni Rosenfeld, "A maximum entropy approach to adaptive statistical language modeling," 1996.

[10] John N Darroch and Douglas Ratcliff, "Generalized iterative scaling for log-linear models," *The annals of mathematical statistics*, pp. 1470–1480, 1972.

[11] Joshua Goodman, "Classes for fast maximum entropy training," in *Acoustics, Speech, and Signal Processing, 2001. Proceedings.(ICASSP'01). 2001 IEEE International Conference on*. IEEE, 2001, vol. 1, pp. 561–564.

[12] Stanley F Chen, "Performance prediction for exponential language models," in *Proc. of NAACL*. Association for Computational Linguistics, 2009, pp. 450–458.

[13] Fadi Biadsy, Keith Hall, Pedro J Moreno, and Brian Roark, "Backoff inspired features for maximum entropy language models," in *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

[14] Fadi Biadsy, Mohammadreza Ghodsi, and Diamantino Caseiro, "Effectively building tera scale maxent language models incorporating non-linguistic signals," *Proc. Interspeech 2017*, pp. 2710–2714, 2017.

[15] Tongzhou Chen, Diamantino Caseiro, and Pat Rondon, "Entropy based pruning of backoff maxent language models with contextual features," 2018.

[16] Jun Wu, "Maximum entropy language modeling with non-local and syntactic dependencies," 2002.

[17] Stanley F Chen, "Shrinking exponential language models," in *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, 2009, pp. 468–476.

[18] Tomáš Mikolov, Anoop Deoras, Daniel Povey, Lukáš Burget, and Jan Černockỳ, "Strategies for training large scale neural network language models," in *Automatic Speech Recognition and Understanding (ASRU), 2011 IEEE Workshop on*. IEEE, 2011, pp. 196–201.

[19] Welin Chen, David Grangier, and Michael Auli, "Strategies for training large vocabulary neural language models," *arXiv preprint arXiv:1512.04906*, 2015.

[20] Ciprian Chelba and Noam Shazeer, "Sparse non-negative matrix language modeling for geo-annotated query session data," in *Automatic Speech Recognition and Understanding (ASRU), 2015 IEEE Workshop on*. IEEE, 2015, pp. 8–14.

[21] Ronnie Chaiken, Bob Jenkins, Per-Åke Larson, Bill Ramsey, Darren Shakib, Simon Weaver, and Jingren Zhou, "Scope: easy and efficient parallel processing of massive data sets," *Proceedings of the VLDB Endowment*, vol. 1, no. 2, pp. 1265–1276, 2008.

[22] Keith B Hall, Scott Gilpin, and Gideon Mann, "Mapreduce/bigtable for distributed optimization," in *NIPS LCCC Workshop*, 2010.

[23] Barret Zoph, Ashish Vaswani, Jonathan May, and Kevin Knight, "Simple, fast noise-contrastive estimation for large rnn vocabularies," in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2016, pp. 1217–1222.

[24] Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson, "One billion word benchmark for measuring progress in statistical language modeling," *arXiv preprint arXiv:1312.3005*, 2013.