Efficient Estimation of Maximum Entropy Language Models with N-gram features: an SRILM extension

Tanel Alumäe[1], *Mikko Kurimo*[2]

[1]Institute of Cybernetics, Tallinn University of Technology, Estonia
[2]Adaptive Informatics Research Centre, Aalto University, Finland

tanel.alumae@phon.ioc.ee, mikko.kurimo@tkk.fi

Abstract

We present an extension to the SRILM toolkit for training maximum entropy language models with N-gram features. The extension uses a hierarchical parameter estimation procedure [1] for making the training time and memory consumption feasible for moderately large training data (hundreds of millions of words). Experiments on two speech recognition tasks indicate that the models trained with our implementation perform equally to or better than N-gram models built with interpolated Kneser-Ney discounting.

Index Terms: language modeling, maximum entropy, open source

1. Introduction

Although maximum entropy (ME) language models (also known as exponential language models) for speech recognition were proposed long time ago [2], they have gained significant attention in recent years (for example, [3, 4]) and are regarded as one of the most promising techniques by many language modeling researchers. Also, much research has been devoted to advances in efficient training, smoothing and adaptation methods for ME models and generalized linear models in general (e.g., [5, 6, 7]).

There are numerous open-source toolkits for training and applying ME models, e.g., MEGAM[1], Maximum Entropy Modeling Toolkit for Python and C++[2], TADM[3], Classias[4]. However, when the size of the output vocabulary in a ME model is large (which is the case with language models), naive training becomes extremely resource consuming, requiring huge amounts of memory and days or weeks to finish, even with small and medium sized training data. One technique to overcome this problem is to cluster words into classes and create two or more models instead of a single one: one for classes, and a second one for words-given-classes [8]. This allows to drastically reduce computation time since the size of the output vocabulary in each of the models is now much smaller. A second approach takes advantage of the hierarchical nature of N-gram features by sharing computations on lower-order N-gram levels [1]. Unfortunately, none of the open-source ME toolkits currently supports such techniques out-of-the-box, although it is not difficult to apply the class-based method with any toolkit. In addition, writing wrapper code for applying generic ME models for language processing tasks (such as rescoring) requires considerable efforts.

In this paper, we describe an extension to SRILM [9] for training ME models with N-gram features. SRILM is a highly popular toolkit among language modeling researchers and has a wide range of features. By implementing ME models according to the SRILM API, we automatically get access to many useful components that apply language models, such as N-best rescoring, lattice rescoring, model interpolation, etc. We chose to implement ME models as an extension to SRILM since it would have been very resource-consuming to implement such features independently. We used the hierarchical feature technique [1] and a scalable optimization algorithm [5] to be able to handle large sizes of training data. The paper gives an overview of our implementation and reports perplexity and N-best rescoring results on two speech recognition tasks. Our experiments demonstrate the ability of the implementation to scale up to moderately large training data.

2. Maximum entropy language models

Maximum entropy (ME) modeling is a framework that has been used in a wide area of natural language processing tasks. A conditional ME model has the following form:

$$P(x|h) = \frac{e^{\sum_i \lambda_i f_i(x,h)}}{Z(h)} \tag{1}$$

where x is an outcome (in case of a LM, a word), h is a context (the word history) and

$$Z(h) = \sum_{x_i \in V} e^{\sum_j \lambda_j f_j(x_i,h)} \tag{2}$$

where V a set of all possible outcomes (words). The functions f_i are (typically binary) feature functions. During ME training, the optimal weights λ_i corresponding to features $f_i(x,h)$ are learned. More precisely, finding the ME model is equal to finding weights that maximize the log-likelihood $L(X;\Lambda)$ of the training data X. The weights are learned via improved iterative scaling algorithm or some of its modern fast counterparts (e.g., conjugate gradient descent).

To avoid overfitting, ME models are usually smoothed (regularized). The most widely used smoothing method for ME LMs is Gaussian priors [10]: a zero-mean prior with a given variance is added to all feature weights, and the model optimization criteria becomes:

$$L'(X;\Lambda) = L(X;\Lambda) - \sum_{i=1}^{F} \frac{\lambda_i^2}{2\sigma_i^2} \tag{3}$$

[1]http://www.cs.utah.edu/~hal/megam
[2]http://homepages.inf.ed.ac.uk/lzhang10/maxent_toolkit.html
[3]http://tadm.sourceforge.net
[4]http://www.chokkan.org/software/classias/

where F is the number of feature functions. Typically, a fixed hyperparameter $\sigma_i = \sigma$ is used for all parameters. The optimal variance is usually estimated on a development set. Intuitively, this method encourages feature weights to be smaller, by penalizing weights with big absolute values.

Estimating optimal feature weights for language models can take prohibitively long time if done straightforwardly: in each iteration of the estimation algorithm, one has to calculate normalization factors $Z(h)$ for all observed contexts in the training data. For each context, this requires looping over all words in the vocabulary – also those that didn't occur in a given context. However, [1] proposed a technique that greatly decreases the complexity of calculating normalization factors when features are nested and not overlapping, e.g., N-gram features. For example, for a history w_{i-1}, w_{i-2}, the normalization factor can be calculated as follows:

$$\begin{aligned} Z(w_{i-1}, w_{i-2}) = & \\ \textstyle\sum_{w_i \in V} & \quad e^{f_{w_i}} + \\ \textstyle\sum_{w_i \in V_{w_{i-1}}} & \quad (e^{f_{w_{i-1}w_i}} - 1)e^{f_{w_i}} + \\ \textstyle\sum_{w_i \in V_{w_{i-2}w_{i-1}}} & \quad (e^{f_{w_{i-2}w_{i-1}w_i}} - 1)e^{f_{w_{i-1}w_i}} \end{aligned} \tag{4}$$

where V is the vocabulary, $V_{w_{i-1}}$ is the set of words observed after context w_{i-1} and $V_{w_{i-2}w_{i-1}}$ is the set of words observed after context $w_{i-2}w_{i-1}$. The first part of the sum doesn't depend on the context and can be precomputed. The second part of the sum is fixed for all contexts ending with w_{i-1} and its value can be shared between histories ending with w_{i-1}. The final part requires summing over the set of words observed after the history $w_{i-2}w_{i-1}$ which is very small for most contexts. This approach can be extended for features that are not strictly nested.

3. Design and implementation

The ME extension for SRILM was designed with the following goals in mind: (1) ability to handle large amounts of training data (i.e., hundreds of millions of words); (2) incorporation of state-of-the-art algorithms for training ME models; (3) transparent integration into SRILM, in order to make using ME models for research and application development as easy as possible.

As the scalability to large training corpus was one of the design targets, we represent features hierarchically and apply the hierarchical training trick [1] described in the previous section. During the writing of this paper, only strictly nested N-gram features up to any length are supported, but we intend to add support for more complex features in the future. The training process was inspired by the TextModeller toolkit (now part of Python SciPy) [11]. We use the C++ *valarray* data structure for high-performance numerical operations when computing the normalization factors and feature expectations. The *valarray* data structure provides a mechanism for expressing many one-loop operations as a single expression which could be highly optimized by the compiler. In addition, most expensive parts of computing normalization factors and expectations are parallelized using OpenMP directives. For parameter optimization, we use the Orthant-Wise Limited-memory Quasi-Newton (OWL-QN) method [5] through the libLBFGS[5] library. The $\ell_1 + \ell_2^2$ regularization scheme is used. Under this scheme, the objective function becomes

$$L_{\ell_1+\ell_2^2}(X;\Lambda) = L(X;\Lambda) - \frac{\alpha}{D}\sum_{i}^{F}|\lambda_i| - \sum_{i=1}^{F}\frac{\lambda_i^2}{2\sigma_i^2 D} \tag{5}$$

where D is the number of training observations. The regularization hyperparameters can be set by the user, with default parameters set to their global optimal values ($\alpha = 0.5, \sigma^2 = 6$) as discovered empirically in [4].

The software can also perform simple adaptation as described in [12]: this technique uses parameters estimated from out-of-domain data as the prior means when learning parameters from in-domain data. The objective function then becomes

$$L_{\ell_1+\ell_2^2}(X;\Lambda) = L(X;\Lambda) - \frac{\alpha}{D}\sum_{i}^{F}|\lambda'_i - \lambda_i| - \sum_{i=1}^{F}\frac{(\lambda'_i - \lambda_i)^2}{2\sigma_i^2 D} \tag{6}$$

where λ'_i are parameters estimated from out-of-domain data.

We also designed a simple file format to represent ME models with N-gram features.

4. Experiments

4.1. Performance of ME models

In this section, we report how ME models trained with our implementation perform on two speech recognition tasks.

Task 1: English Broadcast News. This recognition task consists of the English broadcast news section of the 2003 NIST Rich Transcription Evaluation Data. The data includes six news recordings from six different sources with a total length of 176 minutes.

As acoustic models, the CMU Sphinx open source triphone HUB4 models for wideband (16kHz) speech[6] were used. The models have been trained using 140 hours of speech.

For training the LMs, two sources were used: first 5M sentences from the Gigaword (2nd ed.) corpus (99.5M words), and broadcast news transcriptions from the TDT4 corpus (1.19M words). The latter was treated as in-domain data in the adaptation experiments. We used only 5M sentences of Gigaword data because our initial implementation didn't scale properly to larger training data. A vocabulary of 26K words was used. It is a subset of a bigger 60K vocabulary, and only includes words that occurred in the training data. The OOV rate against the test set was 2.4%.

The audio used for testing was segmented into parts of up to 20 seconds in length. Speaker diarization was applied using the LIUM_SpkDiarization toolkit [13]. The CMU Sphinx 3.7 was used for decoding. A three-pass recognition strategy was applied: the first pass recognition hypotheses were used for calculating MLLR-adapted models for each speaker. In the second pass, the adapted acoustic models were used for generating a 5000-best list of hypotheses for each segment. In the first two passes, a trigram LM was used that was an interpolation of source-specific models which were estimated using Kneser-Ney discounting. In the third pass, different LMs in turn were compared in rescoring the hypotheses. Finally, the 1-best hypothesis was selected, based on the new LM scores.

Task 2: Estonian Broadcast Conversations. The second recognition task consists of four recordings from different live talk programs from three Estonian radio stations. Their format consists of hosts and invited guests, spontaneously discussing

[5]`http://www.chokkan.org/software/liblbfgs/`

[6]`http://www.speech.cs.cmu.edu/sphinx/models/`

Table 1: *Performance comparison of various trigram and ME models on two tasks.*

		Perplexity		WER	
Task	Model training	Trigram	Maxent	Trigram	Maxent
English	Out-of-domain	301	297		
	In-domain	315	309		
	Interpolated	218	219	25.7	26.0
	Adapted	N/A	217		25.5
	Adapted ME + Interpolated trigram	206		25.5	
Estonian	Out-of-domain	245	239		
	In-domain	446	458		
	Interpolated	184	187	39.4	39.2
	Adapted	N/A	177		38.0
	Adapted ME + Interpolated trigram	167		37.5	

current affairs. There are 40 minutes of transcriptions, with 11 different speakers.

The acoustic models were trained on various wideband Estonian speech corpora: the BABEL speech database (9h), manually transcribed Estonian broadcast news (7.5h) and manually transcribed radio live talk programs (10h). The models are triphone HMMs, using MFCC features.

For training the LMs, two sources were used: about 10M sentences from various Estonian newspapers, and manual transcriptions of 10 hours of live talk programs from three Estonian radio stations. The latter is identical in style to the test data, although it originates from a different time period and covers a wider variety of programs, and was treated as in-domain data.

As Estonian is a highly inflective language, morphemes are used as basic units in the LM. We use a morphological analyzer [14] for splitting the words into morphemes. After such processing, the newspaper corpus includes of 185M tokens, and the transcribed data 104K tokens. A vocabulary of 30K tokens was used for this task, with an OOV rate of 1.7% against the test data. After recognition, morphemes were concatenated back to words.

As with English data, a three-pass recognition strategy involving MLLR adaptation was applied.

Results. For both tasks, we measured the perplexity of a wide range of models against the test data and performed a few N-best rescoring experiments. We built trigram models using interpolated Kneser-Ney discounting from out-of-domain data and in-domain data, and interpolated the models, using development data for optimizing the interpolation weight. Similarly, ME models were built using hierarchical trigram features. We also built an adapted ME model, using the out-of-domain model as a prior for estimating in-domain model. This model also included N-gram features occurring only in in-domain data, which thus received a zero prior value for the corresponding weight – this is similar to [12]. Finally, we mixed two best-performing models of different kind – interpolated N-gram and adapted ME model. No feature cut-off was applied for any of the models. For ME models, default regularization parameters were used. The results are given in Table 1.

We analyzed the differences in per-speaker WER scores using the Wilcoxon test. For the English task, the WER differences were not statistically significant, except for the difference between interpolated N-gram (25.7) and adapted ME (25.5) models. On the other hand, for the Estonian task, all WER differences were significant, except the difference between interpolated N-gram (39.4) and interpolated ME (39.2) models. The results confirm the findings in [12] that the adaptation technique is very suitable for situations where in-domain training data is very limited (only 104K tokens for the Estonian task).

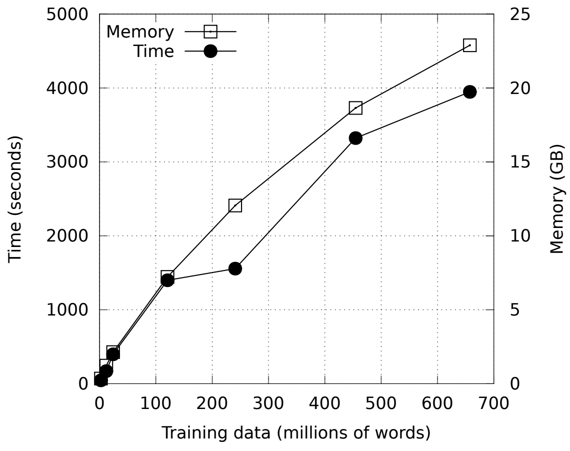

Figure 1: Graph of training time and maximum RAM usage *vs.* size of training data (40K vocabulary, trigram features).

4.2. Training time and memory usage

We measured the time and maximum RAM required for training ME models with varying sizes of training data. We took increasing amounts of sentences from the English Gigaword (2nd ed.) corpus, starting from the beginning of the corpus. We used a 40K vocabulary (most frequent words in the first 10M sentences in the corpus) and trigram features with no cutoff. The machine used for the experiments had an Intel Quad Core Xeon X5550 processor (2.66 GHz, 8 MB cache) and 24 GB DDR3 RAM. All four cores were participating in the computations. The training times (excluding time required for loading the data and storing the model) and peak RAM usages are reported in Figure 1.

The most time-consuming sections of the ME training algorithm – calculating normalization factors and expectations, and precomputing exponents of feature weights on each evaluation of the objective function – were parallelized using OpenMP instructions. We measured the duration of one training run with increasing maximum number of threads (using the OMP_NUM_THREADS environment variable), using 240M words of Gigaword training data, 40K vocabulary (most fre-

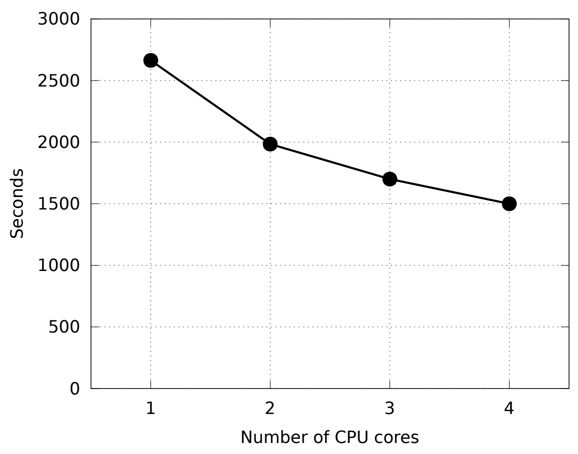

Figure 2: Graph of training time *vs.* number of CPU cores used (training with 240M words, 40K vocabulary, trigram features).

quent words in the data), and trigram features. The times are given in Figure 2.

5. Discussion and Future Plans

We implemented an extension to the SRILM toolkit for training and applying ME language models with N-gram features. The goal of our work was to provide a fast and memory-efficient open-source implementation of ME models for language modeling purposes. Experiments show that ME models trained with our extension are competitive in accuracy to N-gram models trained using interpolated Kneser-Ney smoothing. We implemented a hierarchical training technique [1] and used a scalable training algorithm [5] enabling us to train models on moderately large training corpora (hundreds of millions of words) in relatively short time (up to an hour) on a moderately equipped workstation (quad core CPU with 24 GB RAM). We also implemented a simple adaptation technique which uses a model built from background data as a prior for estimating in-domain model [12]. This technique outperforms linear interpolation when in-domain data is very limited.

The implementation has currently a few shortcomings. The models are restricted to nested N-gram features and as such provide little performance benefits compared to N-gram models with similar features, especially considering that N-gram models can be trained in shorter time with less memory. Therefore, we plan to add support for more complex feature hierarchies in the future. First, we intend to implement the hierarchical training technique for overlapping hierarchical features which would make possible to use other factors of the history besides word identities (such as word classes, syntactic features) since this is the area where ME models should provide benefits over N-gram models. Secondly, we want to add support for hierarchical "feature sets", where a model consists of general and domain-specific features, and during training, global and domain-specific parameters are optimized jointly, using parameters built from pooled data as priors for domain-specific parameters [15, 7]. This method should be ideally suited for language modeling in speech recognition: we almost always have access to large amounts of data from written sources but commonly the speech to be recognized is stylistically noticeably different. The hierarchical adaptation method enables to use even a small amount of in-domain data to modify the parameters estimated from out-ouf-domain data, if there is enough evidence.

We plan to make the extension available for incorporation into the main SRILM distribution.

6. Acknowledgments

This research was partly funded by the Academy of Finland in the project Adaptive Informatics, by the target-financed theme No. 0322709s06 of the Estonian Ministry of Education and Research and by the National Programme for Estonian Language Technology.

7. References

[1] J. Wu and S. Khudanpur, "Building a topic-dependent maximum entropy model for very large corpora," in *Proceedings of ICASSP*, Orlando, Florida, USA, 2002.

[2] R. Rosenfeld, "Adaptive statistical language modeling: A maximum entropy approach," Ph.D. dissertation, Carnegie Mellon University, 1994.

[3] R. Sarikaya, M. Afify, Y. Deng, H. Erdogan, and Y. Gao, "Joint Morphological-Lexical language modeling for processing morphologically rich languages with application to dialectal arabic," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 16, no. 7, pp. 1330–1339, 2008.

[4] S. F. Chen, "Performance prediction for exponential language models," in *Proceedings of HLT-NAACL*, Boulder, Colorado, 2009, pp. 450–458.

[5] G. Andrew and J. Gao, "Scalable training of L1-regularized log-linear models," in *Proceedings of the 24th International Conference on Machine Learning*. Corvalis, Oregon, USA: ACM, 2007, pp. 33–40.

[6] C. Foo, C. B. Do, and A. Y. Ng, "A majorization-minimization algorithm for (multiple) hyperparameter learning," in *Proceedings of the 26th Annual International Conference on Machine Learning*. Montreal, Quebec, Canada: ACM, 2009, pp. 321–328.

[7] J. R. Finkel and C. Manning, "Hierarchical Bayesian domain adaptation," in *Proceedings of HLT-NAACL*, Boulder, Colorado, 2009, pp. 602–610.

[8] J. Goodman, "Classes for fast maximum entropy training," in *Proceedings of ICASSP*, Utah, USA, 2001.

[9] A. Stolcke, "SRILM – an extensible language modeling toolkit," in *Proceedings of ICSLP*, vol. 2, Denver, USA, 2002, pp. 901–904.

[10] S. F. Chen and R. Rosenfeld, "A survey of smoothing techniques for ME models," *IEEE Transactions on Speech and Audio Processing*, vol. 8, no. 1, pp. 37–50, 2000.

[11] E. J. Schofield, "Fast parameter estimation for joint maximum entropy language models," in *Proceedings of Interspeech*, Jeju, Korea, 2004, pp. 2241 – 2244.

[12] C. Chelba and A. Acero, "Adaptation of maximum entropy capitalizer: Little data can help a lot," *Computer Speech & Language*, vol. 20, no. 4, pp. 382–399, Oct. 2006.

[13] P. Deléglise, Y. Estéve, S. Meignier, and T. Merlin, "The LIUM speech transcription system: a CMU Sphinx III-based system for French broadcast news," in *Proceedings of Interspeech*, Lisboa, Portugal, 2005.

[14] H.-J. Kaalep and T. Vaino, "Complete morphological analysis in the linguist's toolbox," in *Congressus Nonus Internationalis Fenno-Ugristarum Pars V*, Tartu, Estonia, 2001, pp. 9–16.

[15] H. Daume III, "Frustratingly easy domain adaptation," in *Proceedings of ACL*, 2007, pp. 256–263.