EM-STYLE OPTIMIZATION OF HIDDEN CONDITIONAL RANDOM FIELDS FOR GRAPHEME-TO-PHONEME CONVERSION

Georg Heigold, Stefan Hahn, Patrick Lehnen, and Hermann Ney

Human Language Technology and Pattern Recognition
Computer Science Department, RWTH Aachen University, Aachen, Germany
heigold,hahn,lehnen,ney@cs.rwth-aachen.de

ABSTRACT

We have recently proposed an EM-style algorithm to optimize log-linear models with hidden variables. In this paper, we use this algorithm to optimize a hidden conditional random field, i.e., a conditional random field with hidden variables. Similar to hidden Markov models, the alignments are the hidden variables in the examples considered. Here, EM-style algorithms are iterative optimization algorithms which are guaranteed to improve the training criterion in each iteration - without the need for tuning step sizes, sophisticated update schemes or numerical line optimization (with hardly predictable complexity). This is a rather strong property which conventional gradient-based optimization algorithms do not have. We present experimental results for a grapheme-to-phoneme conversion task and compare the convergence behavior of the EM-style algorithm with L-BFGS and Rprop.

Index Terms— EM-style optimization, hidden conditional random fields, grapheme-to-phoneme conversion

1. INTRODUCTION

An emerging and promising framework for string recognition are conditional random fields (CRFs) [1]. These are structured log-linear models for string recognition. Recently, the CRFs have been extended to hidden CRFs (HCRFs) to include hidden variables [2]. Hidden variables are an important means to represent variability (e.g. alignment).

Numerical optimization plays a key role in CRFs, HCRFs, and pattern recognition in general. The optimization problem is typically solved using gradient-based algorithms, e.g. L-BFGS [3] or Rprop [4]. This type of optimization algorithms has proven to be efficient in practice. However, they require the tuning of step sizes, sophisticated update schemes or numerical line optimization to guarantee convergence to a local optimum. These approaches are often associated with heuristics or hardly predictable running time per iteration.

For a few special cases, EM-style algorithms are known. These algorithms allow for a simple and safe convergence to a local optimum at low computational cost per iteration. The most prominent examples include expectation-maximization (EM) [5] and generalized iterative scaling (GIS) [6, 7]. Unfortunately, these algorithms cannot be applied to our problem: EM applies to generative models (e.g. Gaussian mixture models) and GIS only works for log-linear models *without* hidden variables.

In [8], we proposed an extension to GIS, G-GIS, to include hidden variables. For one of the special instances considered here, our algorithm and the algorithm for maximum mutual information from incomplete data [9] are identical, and experimental results for relatively small parsing tasks can be found in [10]. In fact, G-GIS is an example for generalized EM (GEM) [11, p.545] where the maximization step consists of a GIS iteration. In general, an arbitrary number of GIS iterations could be done in the maximization step [12]. This, however, adds considerable complexity to each iteration and will presumably not reduce the overall computation time.

The goal of the present work is to apply the EM-style algorithm in [8] to grapheme-to-phoneme conversion [13]. This task has the advantage of being of practical interest while the modest complexity of the task and setup allows for a careful experimental evaluation and comparison with state-of-the-art optimization algorithms, without the need for approximations.

The outline of the remainder of the paper is as follows. Section 2 gives an overview on the grapheme-to-phoneme conversion task. Section 3 summarizes the EM-style algorithm for HCRFs. Comparative experimental results can be found in Section 4. The paper concludes with Section 5.

2. GRAPHEME-TO-PHONEME CONVERSION

Grapheme-to-phoneme conversion (g2p) is an important task to build a state-of-the-art ASR system. It is used to enrich the pronunciation lexicon by words where the lexical representation is known, but not the phonetic presentation. To use CRFs for g2p, a 1-to-1 alignment between source and target side is needed. We adopted the so-called Begin-Inside-Out scheme [14], so we are able to transfer any given alignment into a

This work was partly realized as part of the Quaero Programme, funded by OSEO, French State agency for innovation.

1-to-1 representation. An example of a possible 1-to-1 alignment for a word/pronunciation pair using this scheme would look like this:

"throw"	=	t	h	r	o	w
[θrɐ]		θ_B	θ_I	r_B	ɐ_B	ɐ_I

3. EM-STYLE OPTIMIZATION

In this section, the key results for EM-style optimization as proposed in [8] are summarized. This algorithm is referred to as G-GIS because it is a generalization of generalized iterative scaling (GIS) [6, 7]. G-GIS is then applied to two concrete examples based on linear-chain HCRFs.

Consider the formal training criterion

$$F(\lambda) = \sum_r \log \left(\frac{\sum_C v_{rC} \Phi_\lambda(X_r, C)}{\sum_C w_{rC} \Phi_\lambda(X_r, C)} \right) \quad (1)$$

where X denotes some feature vector and C stands for the class. In our case, the score function Φ is log-linear: $\Phi_\lambda(X, C) = \exp(\sum_i \lambda_i f_i(X, C))$ for a set of feature functions $f_i(X, C)$ and model parameters $\lambda = (\lambda_1, \dots)$. For greater flexibility, we allow for weighting the score function with some non-negative constants v and w. In addition, there is a sum over the index r, e.g., the observation or the position index. Typically, the training criterion is used in combination with ℓ_2-regularization. This adds some notational complexity but is rather straightforward to incorporate, see [15, Chapter 6.4].

Obviously, conventional GIS does not apply to the training criterion in Equation (1) because of the weighted sum over the classes in the numerator. In [8], we derived generalized update rules for this training criterion

$$\lambda_i = \lambda'_i + \frac{1}{F} \log \frac{N_i(\lambda')}{Q_i(\lambda')}$$

where λ' and λ are the old and new model parameters, respectively. The update rules resemble the update rules for conventional GIS but use slightly different accumulation statistics

$$N_i(\lambda') := \sum_r \sum_C \frac{v_{rC} \Phi_{\lambda'}(X_r, C)}{\sum_{\tilde{C}} v_{r\tilde{C}} \Phi_{\lambda'}(X_r, \tilde{C})} f_i(X_r, C). \quad (2)$$

The Q-statistics are defined similarly. Like for conventional GIS, F is the feature count and, in case of binary features, counts the maximum number of features that are simultaneously active. In the following, we discuss two special instances of Equation (1) in more detail.

For sake of concreteness, consider the conversion from a letter string $X = s_1^M$ to a phoneme string t_1^N (see Section 2). Here, we use the hidden variables a_1^M to align the two strings, i.e., $C = (a_1^M, t_1^N)$. Conditional maximum likelihood (CML) for the associated HCRF is defined as

$$F_{CML}(\lambda) = \log \left(\frac{\sum_{a_1^M} \Phi_\lambda(s_1^M, a_1^M, t_1^N)}{\sum_{\tilde{t}_1^N} \sum_{a_1^M} \Phi_\lambda(s_1^M, a_1^M, \tilde{t}_1^N)} \right).$$

It can be implemented by a suitable choice for the weights v and w in Equation (1). The weight v is set to one for the alignments representing the true phoneme string and to zero otherwise, i.e., v is used as a filter. The weight w is always one. The accumulation statistics in Equation (2) for the lexical features, $f_{sa}(s_1^M, a_1^M, t_1^N) := \sum_m \delta(s_m, s)\delta(a_m, a)$, then reads

$$N_{sa}(\lambda') := \sum_{m=1}^{M} \delta(s_m, s) p_{\lambda'}(a_m = a | t_1^N, s_1^M)$$

(the sum over different training words is omitted for simplicity). The statistics can be expressed in terms of conditional probabilities, which are derived from the scoring functions Φ via Bayes rule and marginalization. These quantities include sums over a combinatorial number of strings and, assuming certain independence assumptions, can be efficiently computed with the forward-backward algorithm [15]. The use of weighted finite-state transducers to represent the summation space considerably simplifies the implementation [15, Chapter 3]. The statistics for other features (see Section 4.1 for a detailed setup) and Q are defined and computed similarly.

CML for HCRFs is rather standard. The training criterion in Equation (1), however, also supports more exotic training criteria. This is illustrated by another example, the position-wise CML criterion. In contrast to CML, which optimizes the string log-posteriors, this criterion optimizes the position-wise log-posteriors

$$F_{POS}(\lambda) = \sum_{n=1}^{N} \log \left(\frac{\sum_{\tilde{t}_1^N : \tilde{t}_n = t_n} \sum_{a_1^M} \Phi_\lambda(s_1^M, a_1^M, \tilde{t}_1^N)}{\sum_{\tilde{t}_1^N} \sum_{a_1^M} \Phi_\lambda(s_1^M, a_1^M, \tilde{t}_1^N)} \right).$$

The inspection of the minimum Bayes risk decision rule using the phoneme error gives some motivation for this criterion: the risk can be expressed in terms of the position-wise posteriors and thus, the direct optimization of these posteriors is expected to provide better estimates in general. Position-wise CML has the same formal structure as the training criterion in Equation (1). Again, the weight v is used to filter the hypotheses $C = (a_1^M, t_1^N)$ in the numerator. The weight w is always one. The accumulation statistics in Equation (2) for the lexical features then reads

$$N_{sa}(\lambda') := \sum_{n,m} \delta(s_m, s) \frac{\displaystyle\sum_{\tilde{t}_1^N : \tilde{t}_n = t_n} \sum_{a_1^M : a_m = a} \Phi_{\lambda'}(s_1^M, a_1^M, \tilde{t}_1^N)}{\displaystyle\sum_{\tilde{t}_1^N : \tilde{t}_n = t_n} \sum_{a_1^M} \Phi_{\lambda'}(s_1^M, a_1^M, \tilde{t}_1^N)}.$$

The other statistics are defined similarly. The feature count F is the same as for CML because the denominator is the same. These statistics can be interpreted as (formal) second order statistics (cf. covariance) and thus, can be efficiently computed with the forward/backward algorithm using the expectation semiring, see [15, Chapter 3] for further details.

Table 1. Phoneme error rates (PER) for NETtalk development and evaluation corpus using different initializations, optimized with L-BFGS.

	PER [%]	
initialization	Dev	Eva
from scratch	12.5	12.5
suboptimal	10.8	10.9
maximum	10.6	10.8

4. EXPERIMENTAL RESULTS

We evaluate the EM-style algorithm described in Section 3 for the grapheme-to-phoneme conversion task NETtalk. In particular, we check the feasibility and the theoretical properties of our EM-style algorithm (G-GIS), and compare it to conventional gradient-based optimization algorithms such as L-BFGS and Rprop.

4.1. Database & Setup

For the reported experiments, the English NETtalk 15k corpus has been chosen. It consists of 26 different graphemes and 50 phonemes. The data set is partitioned in roughly 14k training words ('Train'), 1k for development and tuning of the system ('Dev'), and 5k testing words ('Eva').

In the experiments CRFs were used with binary ($\in 0, 1$) feature functions, where lexical features $(a_n = a', s_{n+\epsilon} = s')$, bigram features $(t_{n-1} = t'', t_n = t')$, and any "and"-combination of them (to capture n-grams) is possible. For the optimal setup, lexical features were used with offsets $-4, \ldots, 4$ and combined features are composed of all monotone and overlapping combinations of lexical features of lengths two up to six. The setup with the full feature set achieves phoneme error rates around 6%.

Here, we choose a simpler setup by dropping the combined features to reduce the computational load. We do not believe that the conclusions will significantly change for more complex setups. Table 1 provides a few baseline results. The optimization is performed with L-BFGS, the conventional algorithm for HCRFs. The algorithm terminates if the difference in the training criterion of two subsequent iterations is smaller than the machine precision. Three different ways to initialize the HCRF are used: 'from scratch' (model parameters are set to zero, Dev PER=113.6%), 'suboptimal' (some suboptimal model, Dev PER=14.6%), and 'maximum' (manual segmentation used as the truth, Dev PER=11.1%). The results in Table 1 suggest that local optima are an issue and to achieve optimal performance, a good initial model is required.

4.2. Comparison

Next, we compare G-GIS with L-BFGS and Rprop. The progress of training is plotted in Figure 1 (training criterion on training corpus), Figure 2 (phoneme error rate on development corpus), and Figure 3 (phoneme error rate on evaluation corpus). Here, we are mainly interested in the convergence behavior of the optimization algorithms, which can be described in terms of convergence and efficiency.

By construction, L-BFGS and G-GIS converge to a local optimum. In contrast, Rprop does not necessarily converge. Nevertheless, the curves in Figures 1, 2, and 3 suggest that all three algorithms finally converge to a local optimum. As expected, the local optima are different but comparable, see Table 2.

For the given example, the number of training iterations gives a reasonable estimate of the actual computational cost. In general, this is not true as the computational load per iteration varies for L-BFGS due to the implicit line minimization. Both L-BFGS and Rprop converge within a few hundred iterations. G-GIS, however, takes about ten times more iterations to converge.

Hence, what makes G-GIS so much slower than L-BFGS? The update rules for G-GIS are constructed such as to always improve the training criterion, also in the worst case. In contrast, L-BFGS implements a "trial-and-error" approach: start with a reasonable step size and reduce it until the current iteration improves the last iteration. In general, this can be costly (up to ten trials in the given example, i.e., the actual step size is 2^{10} times smaller than the initial). In practice, however, this only occurs rarely such that the overall training time is hardly affected. Keeping this in mind, G-GIS favorably compares to L-BFGS and Rprop.

5. SUMMARY

We compared EM-style to conventional gradient-based optimization for hidden conditional random fields. Experiments were performed for the grapheme-to-phoneme conversion task NETtalk. The experimental results suggest that EM-style optimization is feasible on this task and shows the expected convergence behavior. In particular, it increases the training criterion in each iteration. The downside of the increased safety and simplicity of the EM-style algorithm is that it comes at the expense of reduced convergence speed. Hence, experimenters need to carefully consider the trade-off between safety/simplicity and efficiency for numerical optimization. Also, EM-style algorithms aim to effectively converge to a local optimum. To find good optima, however, additional heuristics (e.g. good initial model) are required in general.

6. REFERENCES

[1] J. Lafferty, A. McCallum, and F. Pereira, "Conditional random fields: Probabilistic models for segmenting and

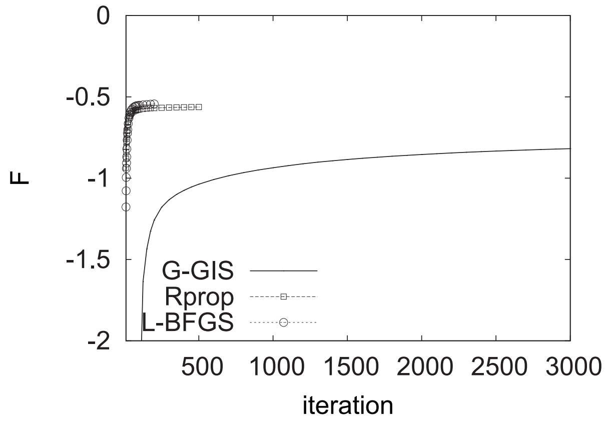

Fig. 1. Progress of training criterion F for different optimization algorithms (training data).

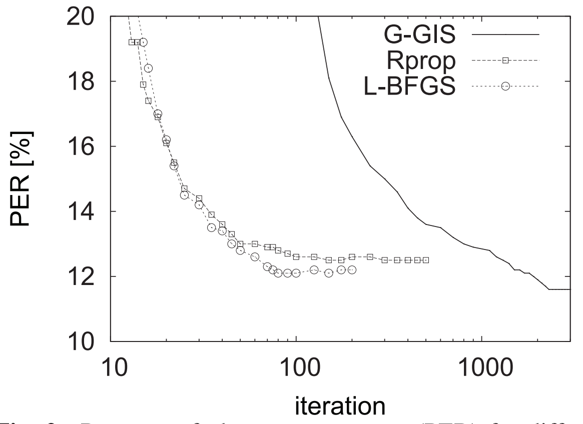

Fig. 2. Progress of phoneme error rate (PER) for different optimization algorithms (development data).

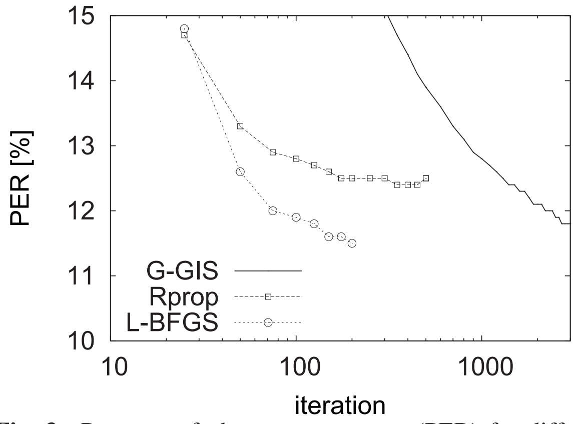

Fig. 3. Progress of phoneme error rate (PER) for different optimization algorithms (evaluation data).

Table 2. Phoneme error rates (PER) for NETtalk development and evaluation corpus using different optimization algorithms, initialized from scratch.

	PER [%]	
optimization	Dev	Eva
Rprop	12.5	12.5
L-BFGS	12.2	11.5
G-GIS	11.6	12.0

labeling sequence data," in *ICML*, San Francisco, CA, USA, June – July 2001.

[2] A. Gunawardana, M. Mahajan, A. Acero, and J.C. Platt, "Hidden conditional random fields for phone classification," in *Interspeech*, Lisbon, Portugal, Sept. 2005.

[3] J. Nocedal and S.J. Wright, *Numerical Optimization*, Springer, 1999.

[4] M. Riedmiller and H. Braun, "A direct adaptive method for faster backpropagation learning: The Rprop algorithm," in *ICNN*, San Francisco, CA, USA, 1993.

[5] A.P. Dempster, N.M. Laird, and D.B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society*, vol. 39, no. B, 1977.

[6] J. Darroch and D. Ratcliff, "Generalized iterative scaling for log-linear models," *Annals of Mathematical Statistics*, vol. 43, 1972.

[7] S.A. Della Pietra, V.J. Della Pietra, and J. Lafferty, "Inducing features of random fields," *PAMI*, vol. 19, no. 4, 1997.

[8] G. Heigold, T. Deselaers, R. Schlüter, and H. Ney, "GIS-like estimation of log-linear models with hidden variables," in *ICASSP*, Las Vegas, NV, USA, Apr. 2008.

[9] S. Riezler, *Probabilistic Constraint Logic Programming*, Ph.D. thesis, Universität Tübingen, Germany, 1998.

[10] S. Riezler, J. Kuhn, D. Prescher, and M. Johnson, "Lexicalized stochastic modeling of constraint-based grammars using log-linear measures and EM training," in *ACL*, Hong Kong, Oct. 2000.

[11] C.M. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2006.

[12] S. Wang, D. Schuurmans, and Y. Zhao, "The latent maximum entropy principle," in *ISIT*, Lausanne, Switzerland, June – July 2002.

[13] M. Bisani and H. Ney, "Joint-sequence models for grapheme-to-phoneme conversion," *Speech Communication*, vol. 50, no. 5, pp. 434–451, 2008.

[14] L. Ramshaw and M. Marcus, "Text chunking using transformation-based learning," in *Proc. of the 3rd Workshop on Very Large Corpora*, Cambridge, MA, USA, June 1995.

[15] G. Heigold, *A Log-Linear Discriminative Modeling Framework for Speech Recognition*, Ph.D. thesis, RWTH Aachen University, Aachen, Germany, June 2010.