International Symposium on Chinese Spoken Language Processing (ISCSLP 2002)
Taipei, Taiwan
August 23-24, 2002

A COMPARATIVE STUDY OF QUICKPROP AND GPD OPTIMIZATION ALGORITHMS FOR MCELR ADAPTATION OF CDHMM PARAMETERS

Jian WU and Qiang HUO

Department of Computer Science and Information Systems
The University of Hong Kong, Pokfulam Road, Hong Kong
(Email: jwu@csis.hku.hk, qhuo@csis.hku.hk)

ABSTRACT

In our previous work, we have presented an approach of minimum classification error linear regression (MCELR) for adaptation of Gaussian mixture continuous density HMM (CDHMM) parameters. It is shown that a stochastic approximation approach known as the GPD (generalized probabilistic descent) can be used to optimize the MCE objective function. However, it is relatively difficult to set an appropriate value for the learning control parameter to achieve a fast yet stable GPD optimization process. In this paper, we study another batch-mode approximate second-order optimization approach, namely Quickprop, aiming at speeding up the convergence of the objective function of MCELR while making the learning more robust. It is demonstrated by a series of experiments for supervised speaker adaptation that Quickprop is a better alternative to GPD for MCELR.

1. INTRODUCTION

In [1], we have presented a formulation of minimum classification error linear regression (MCELR) approach for adaptation of Gaussian mixture continuous density HMM (CDHMM) parameters. It is demonstrated that the MCELR can be used to adapt the MCE-trained HMM parameters under a consistent criterion of minimum classification error. In a supervised speaker adaptation application, we observe that such adapted models perform better than the MLLR-adapted ones from ML-trained seed models. We also observe that the MCELR performs consistently better than MLLR for either set of seed models. In all of these experiments, we have used a stochastic approximation approach known as the GPD (generalized probabilistic descent) for optimizing the MCE objective function. However, it is well known that the effectiveness of the GPD-based optimization relies heavily on the appropriate setting of the learning rate. That is, for different training data and smoothed objective function, the optimal learning rate may be different. Generally speaking, the learning rate should be fine-tuned by a series of experiments for each specific application task and scenario. This turns out to be not so practical for MCELR-based adaptation because of the changing size and nature of the adaptation data for different speaker. Traditional second-order optimization methods such as Newton's method or secant methods (historically also called quasi-Newton's methods) are thus attractive alternatives for MCELR because they are less sensitive to the learning control parameters.

This research was supported by grants from the RGC of the Hong Kong SAR (Project No. HKU7022/00E) and the ATR-SLT Labs of Japan.

Nevertheless, the exact implementation of such second-order methods demands a Hessian matrix that can be either calculated analytically or approximated via some computationally intensive procedures [2]. For large scale optimization problems, some heuristic strategies have to be adopted to make the algorithms practical. Among many approaches, the Quickprop [3] is demonstrated to be useful in quite a few applications (e.g., [4], [5]). Motivated by these previous studies, we have conducted a comparative study by using Quickprop and GPD for MCELR-based speaker adaptation in a continuous Putonghua (Mandarin Chinese) recognition system. In the following, we compare these two approaches in the context of MCELR, report the experimental results, and discuss the strength and weakness of the individual algorithms.

2. MCELR FORMULATION

Like most of linear regression based adaptation approaches, such as MLLR [6], all of the Gaussian components from all CDHMMs will be clustered into several regression classes as determined by a regression class tree in the context of MCELR. For a particular class m, R similar Gaussian components $\{m_r\}_{r=1}^{R}$ share two linear transformations for the estimation of the new mean vectors and covariance matrices. In the scenario of speaker adaptation, these linear transformations are estimated using a set of adaptation samples, $\mathcal{X} = \{X_1, X_2, \ldots, X_I\}$, from the target speaker.

Similar to that of the MCE framework formulated originally in [7], the objective function of MCELR is formed as the empirical classification error defined on the adaptation set,

$$\ell(\Theta) = \frac{1}{I}\sum_{i=1}^{I} l(X_i; \Lambda, \Theta) \ , \qquad (1)$$

where Θ and Λ denote the parameters of linear transformations and CDHMMs, respectively. The loss function $l(X; \Lambda, \Theta)$ in Eq.(1) is approximated by a sigmoid function as follows:

$$l(X; \Lambda, \Theta) = \frac{1}{1 + exp(-\alpha d(X; \Lambda, \Theta) + \beta)} \ , \qquad (2)$$

where α and β are two control parameters. In the above equation, $d(X; \Lambda, \Theta)$ is a misclassification measure defined as follows:

$$d(X; \Lambda, \Theta) = -g(X; \Lambda, \Theta) + \bar{g}(X; \Lambda, \Theta) \ , \qquad (3)$$

where $g(X; \Lambda, \Theta)$ is a discriminant function for recognition decision making, which is usually defined as the log-likelihood of X under the current HMM parameters transformed from Λ by

Θ. $\bar{g}(X; \Lambda, \Theta)$ is an antidiscriminant function based on either the competing N-best list or word lattice (refer to [1] for details).

For simplicity, only the linear transformation W^m's for mean vectors are considered as transformation parameters in the following discussion and experiments. Generally speaking, for an unconstrained optimization problem, the accuracy of the solution can be greatly affected by the conditioning of the Hessian matrix at that solution. In MCELR, although it is difficult to derive the Hessian matrix and thus impossible to properly scale each transformation parameter to improve the conditioning, it is still helpful, according to our preliminary experiments, to normalize the mean vectors during the transformation as follows:

$$\tilde{\mu}_{m_r} = C_{m_r} \mu_{m_r} \ , \tag{4}$$

where C_{m_r} is the matrix satisfying

$$\Sigma_{m_r}^{-1} = C_{m_r} C_{m_r}^T \tag{5}$$

and μ_{m_r} and Σ_{m_r} are the n-dimensional mean vector and the $n \times n$ covariance matrix of the m_r-th Gaussian component before transformation, respectively. Then the linear transformation W^m is applied to the normalized mean vectors and the new mean vector $\hat{\mu}_{m_r}$ will be

$$\hat{\mu}_{m_r} = C_{m_r}^{-1} W^m \tilde{\xi}_{m_r} \ , \tag{6}$$

where $\tilde{\xi}_{m_r} = (1, \tilde{\mu}_1, \tilde{\mu}_2, \ldots, \tilde{\mu}_n)^T$ is the extended vector of $\tilde{\mu}_{m_r}$ and $W^m = \{\omega_{uv}^m\}$ is the $n \times (n+1)$ matrix to be estimated.

Based on the above notations, the (u, v)-th element of the gradient direction $\dot{l}^m$ used for the optimization of W^m, given an adaptation sample with a sequence of T feature vectors, $X = \{x_t\}_{t=1}^T$, is derived as

$$\dot{l}_{uv}^m \triangleq \frac{\partial l}{\partial \omega_{uv}^m} = \alpha l(1-l) \sum_{t=1}^{T} \sum_{r=1}^{R} a_{m_r}(t) \phi_{uv}(x_t, m_r) \ , \tag{7}$$

where $a_{m_r}(t)$ is a coefficient related to the occupancy probability of m_r-th Gaussian component after transformation at time t and $\phi_{uv}(x_t, m_r)$ is the (u, v)-th element of the following matrix:

$$\Phi(x_t, m_r) = C_{m_r}(x_t - \hat{\mu}_{m_r}) \cdot \tilde{\xi}_{m_r}^T \ . \tag{8}$$

To facilitate the clear presentation of the algorithms in next section, W^m and $\dot{l}^m$ are also re-written as the (n^2+n)-dimensional vectors $\omega_m = \{\omega_j^m\}_{j=1}^J$ and $\dot{l}_m = \{\dot{l}_j^m\}_{j=1}^J$, respectively, while for each index j, there is an unique index pair (u, v) associated with it.

3. OPTIMIZATION ALGORITHMS FOR MCELR

3.1. Generalized Probabilistic Descent Algorithm

In order to minimize the objective function of Eq.(1), the following GPD algorithm can be applied to the updating of ω_m,

$$\omega_j^m(p, k+1) = \omega_j^m(p, k) - \epsilon_{p,k} \dot{l}_j^m(p, k) \ , \tag{9}$$

where p refers to the current epoch of iterations and k refers to the cumulative number of training samples presented in this epoch so far. The learning rate $\epsilon_{p,k}$ tends to zero with the increase of p and k, following the schedule described in [1], from an initial value ϵ_0 which is the only parameter controlling the optimization process of GPD. $\dot{l}_j^m(p, k)$ denotes $\dot{l}_j^m$ calculated using Eq.(7) with the current transformed HMM parameters and the k-th sample of p-th epoch.

Although the efficacy of this sequential procedure has been demonstrated by many experiments, it is still uneasy to use in the optimization of MCELR. Firstly, the sequential procedure has to be used if the training samples are not available before learning starts. However, if all samples are available in the case of batch-mode supervised adaptation, collecting the total gradient information before deciding the next step can be useful to avoid a mutual interference of the parameter changes, especially for the large-scale problems. Undoubtedly, one of the reasons in favor of the sequential approach is that it can introduce some randomness that might be helpful in escaping from a “bad” local optimum. However, there is also the risk of missing a good local optimum for the same reason. Actually, the choice of using a sequential or a batch mode optimization depends on the nature of the task under investigation. Secondly, GPD needs a careful tuning, via “trial-and-error”, of the learning rate, to achieve a good learning behavior. Unfortunately, according to our experiments on MCELR, the GPD optimization process is sensitive to the initial value, ϵ_0, of the learning schedule, yet the most suitable value varies greatly for different speakers. Therefore, it is difficult to exploit the full potential of MCELR if only a fixed learning rate schedule can be used for different speakers with different amount of adaptation data, that is unfortunately the case in speaker adaptation application.

3.2. Quickprop Algorithm

An alternative approach for mitigating the above drawbacks of simple gradient scheme is to use second-order information to decide the parameter updates. Usually, the second-order methods are implemented in a batch mode. For example, the batch mode Newton's method will adopt the following formula to update the parameters:

$$\omega_m(p+1) = \omega_m(p) - [H_m(p)]^{-1} \dot{l}_m(p) \ , \tag{10}$$

if the $J \times J$ Hessian matrix $H_m(p) \big(\triangleq \nabla \dot{l}_m(p) \big)$ is positive definite, where $\dot{l}_m(p) = \sum_{i=1}^{I} \dot{l}_m(p, i)/I$ is the total gradient information of the current epoch. However, for an objective function like Eq.(1), the computation of the Hessian matrix is very complicated. In the work presented here, we use the following Quickprop algorithm to approximate the j-th diagonal elements h_{jj} of the Hessian matrix and treat the other off-diagonal elements as zero:

$$h_{jj}(p) = \frac{\partial \dot{l}_j^m(p)}{\partial \omega_j^m} \approx \frac{\dot{l}_j^m(p) - \dot{l}_j^m(p-1)}{\omega_j^m(p) - \omega_j^m(p-1)} \ . \tag{11}$$

Using Eq.(10) and (11) gives

$$\omega_j^m(p+1) = \omega_j^m(p) - \frac{\dot{l}_j^m(p)}{\dot{l}_j^m(p) - \dot{l}_j^m(p-1)} [\omega_j^m(p) - \omega_j^m(p-1)] \ . \tag{12}$$

The rationale behind the diagonal approximation of Hessian matrix by Quickprop algorithm can be explained in another way based on two “risky” assumptions made by Fahlman [3]:

- First, the curve of objective function with respect to each individual linear transformation parameter can be approximated by a parabola whose arms open upward;
- Second, the change in the slope of the curve, as seen by each individual transformation parameter, is not affected by all the other transformation parameters that are indeed changing at the same time.

Therefore, for each parameter, we can use the gradient vectors measured in the previous and current iterations, say, $\dot{l}_j^m(p-1)$ and $\dot{l}_j^m(p)$, and the related parameter update, $\Delta\omega_j^m(p-1)\big(\triangleq \omega_j^m(p) - \omega_j^m(p-1)\big)$, to predict independently the next step according to the upward parabola determined by these measurements. It thus needs only simple computation to jump directly to the minimum point of the suppositional parabola:

$$\begin{aligned} \Delta\omega_j^m(p) &= \kappa(p)\Delta\omega_j^m(p-1) \\ &= \frac{\dot{l}_j^m(p)}{\dot{l}_j^m(p-1) - \dot{l}_j^m(p)}\Delta\omega_j^m(p-1)\ , \end{aligned} \tag{13}$$

which is in concordance with that of Eq.(12).

It should be noted that the Eq.(10) can lead to a local optimum only when the Hessian matrix H_m is positive definite and the initial point is sufficiently close to the optimum. The Quickprop also suffers from such limitations. Hence, for the case where the current gradient is same in sign as the previous gradient but is the same size or even larger in magnitude, which leads to a zero or negative h_{jj}, the assumptions of Quickprop will not be true any more. If Eq.(13) is followed blindly, an unreasonable infinite or actual backward step will occur, which would end up the iteration at a point out of the trustworthy region or even toward a local maximum. In this case a limit on the next step size has to be set as the $\boldsymbol{\mu}$ times the previous update step. This parameter $\boldsymbol{\mu}$ is called "maximum growth factor" and usually set as 1.75. On the other hand, when the current gradient direction $\dot{l}_j^m(p)$ is same in sign as the previous gradient $\dot{l}_j^m(p-1)$ but is less in magnitude, $\kappa(p)$ will be positive which means the direction of the minimization is still not changed and previous step should be maintained in sign for the current update. When the current gradient is opposite in sign with the previous gradient, $\kappa(p)$ will be negative, which means the local optimum was crossed just now and a back step should be taken.

Another requirement in implementing Quickprop is to bootstrap the learning process when there is no previous update and gradient direction at the beginning of the first epoch. Also the similar case will occur when the process should be restarted for a particular parameter that was taken a step of zero previously but is now located in a non-zero-gradient region because something is changed with the update of other parameters. In our experiments, a GPD term, where the learning rate ϵ_p tends to zero from ϵ_0 with the increase of the epochs, once again, is used alone for the bootstrapping.

4. EXPERIMENTS AND RESULTS

4.1. Experimental Setup and Baseline Systems

To compare and identify the strength and weakness of GPD and Quickprop algorithms for optimization in MCELR adaptation, a series of experiments for continuous speech recognition of 410 Putonghua (Mandarin Chinese) base syllables disregarding tones are performed. As for the syllable language model, a uniform grammar with a syllable perplexity of 411 (i.e., each syllable can be followed by any of the 410 syllables and silence) is used. All of the recognition experiments are performed with the search engine of HTK3.0 toolkit [8]. Other experimental setup is the same as that in [9] and is outlined in the following.

Two speech corpora are used for our experiments. The first speech corpus is the HKU96 Putonghua Corpus, which consists of a total of 20 native Putonghua speakers, 10 females and 10 males,

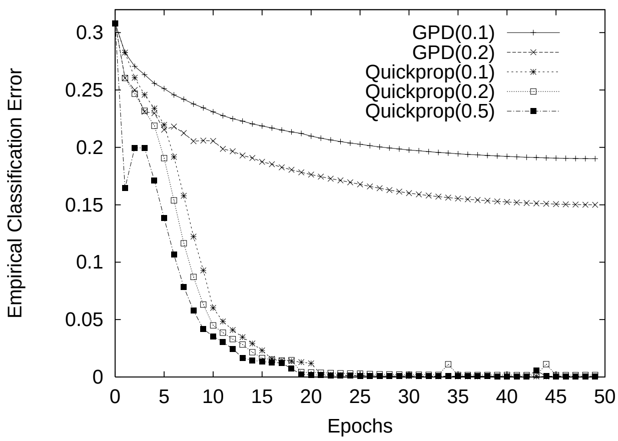

Fig. 1. Learning curves, in terms of empirical classification error, averaged over 12 speakers with respect to the number of epochs (The number within parentheses denotes the value of ϵ_0)

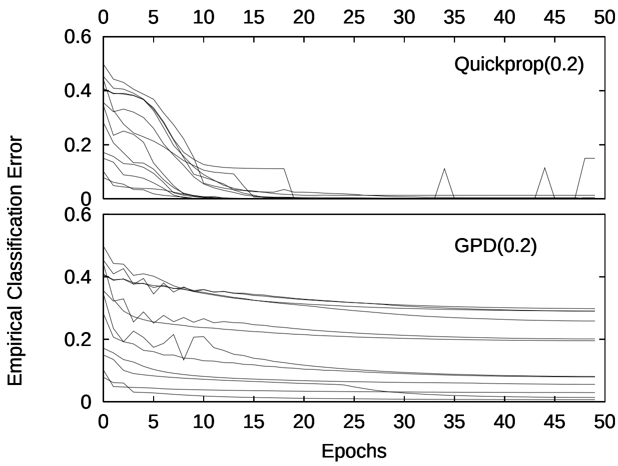

Fig. 2. Learning curves of 12 individual speakers

each speaking hundreds of sentences from newspaper text. 18224 sentences (about 15.5 hours of raw speech) from 18 speakers (9 males and 9 females) are used to train our baseline SI (speaker independent) ASR systems using MCE criterion. The second database is the 863 Putonghua Corpus acquired from mainland China. 12 speakers (6 males and 6 females) are randomly chosen from this corpus to serve as the SI testing speakers. Among all of the 519 sentences by each speaker, 100 sentences are reserved for testing and the remaining for adaptation.

To construct our baseline systems, for each speech frame, a 39-dimensional feature vector is generated which consists of 12 MFCCs and log-scaled energy normalized by the peak of the individual sentence, plus their first and second order derivatives. Sentence-based cepstral mean subtraction is applied in both training and adaptation/testing. The basic speech units are the triphones considering both the within-syllable and cross-syllable contextual dependencies. Each triphone is modeled by a left-to-right 3-state CDHMM. There are 3000 tied-states in total, each having 4 Gaussian mixture components with diagonal covariance matrices. The

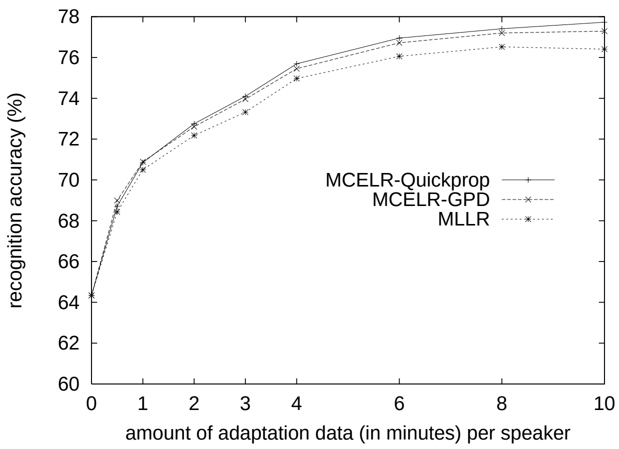

Fig. 3. Performance (syllable accuracy in %) comparison averaged over 12 speakers with respect to the amount of adaptation data per speaker (in minutes)

baseline system is a speaker independent decision-tree based mixture Gaussian tied-state HMM system trained first by using the ML criterion from the speech data in HKU96 corpus and then updated iteratively by minimizing the embedded syllable string error rate.

Hereinafter, all of the MCELR experiments will be started from this MCE-trained seed model with the initial transformation parameters being the MLLR matrices. A common regression tree with 161 leaf nodes and the word lattices used for MCELR are both derived from the ML-trained CDHMM using the HTK3.0 toolkit.

4.2. Comparison of Learning Behavior of GPD and Quickprop Algorithms

Starting from the MCE-trained speaker independent CDHMMs, we performed supervised MCELR adaptation experiments on 12 testing speakers by using both sequential mode GPD and batch mode Quickprop algorithms for optimization. One set of experimental results when the amount of adaptation data for each speaker is 6 minutes are plotted in Fig.1. In these experiments, different values of ϵ_0 for both GPD and the bootstrapping process to start/restart Quickprop are chosen while all the other control parameters, such as α and β, are kept the same. It is observed that in terms of objective function (empirical classification error), the convergence of sequential gradient descent method is much slower than that of Quickprop algorithm, albeit the optimization of Quickprop is not fast either or even oscillates during the first several iterations.

By comparing the learning behavior of GPD and Quickprop algorithms with different learning rate, it is observed that GPD is more sensitive to the ϵ_0 value than Quickprop. Furthermore, we plot individually, in Fig.2, the learning curves of 12 speakers corresponding to the curves GPD(0.2) and Quickprop(0.2) in Fig.1. It is observed that, for Quickprop algorithm, the objective functions for all speakers are reduced to a similar low level after several epochs. However, the dynamic range of the learning curves by GPD is almost unchanged, which implies that the objective functions for some speakers may not reach their optimum, given the common parameter value ϵ_0.

4.3. Comparison of Performance

The total number of training epochs is set to be 10 in this set of MCELR experiments. Fig.3 shows the performance comparison of three sets of speaker adaptation experiments on 12 speakers with respect to the amount (in minutes) of adaptation data per speaker. In the figure, MCELR-Quickprop and MCELR-GPD refer to the MCELR adaptation starting from the MCE-trained seed models by Quickprop and GPD algorithms, respectively.

It is observed that although MCELR-Quickprop decreases the objective function much more than MCELR-GPD, it can not assure a similar impressive performance improvement in recognition accuracy. The performance of MCELR-Quickprop only outperforms slightly over that of MCELR-GPD. For the case of small amount of adaptation data, Quickprop is even worse than GPD algorithm. It might be attributed to the "over-training" problem when the adaptation data are not sufficient and representative enough.

5. SUMMARY

We have presented a heuristic approach of an approximate second-order optimization algorithm, namely Quickprop, for MCELR adaptation. By a series of comparative experiments, we demonstrate that in comparison with GPD, Quickprop algorithm can achieve faster convergence yet be less sensitive to the setting of the initial learning parameter in the optimization of the objective function for MCELR adaptation. It is our hope that this study will enable more ASR systems to be constructed by using MCE for both training and adaptation.

6. REFERENCES

[1] J. Wu and Q. Huo, "Supervised adaptation of MCE-trained CDHMMs using minimum classification error linear regression," in *Proc. of ICASSP 2002*, May 2002.

[2] R. Battiti, "First- and second- order methods for learning: between steepest descent and Newton's method," *Neural Computation*, vol. 4, pp. 141–166, 1992.

[3] S. E. Fahlman, "An empirical study of learning speed in back-propagation networks," Tech. Rep. CMU-CS-88-162, Carnegie Mellon University, Pittsburgh, PA, 1988.

[4] V. Valtchev S. Kapadia and S. J. Young, "MMI training for continuous phoneme recognition on the TIMIT database," in *Proc. of ICASSP 1993*, 1993.

[5] A. E. Biem, "Minimum classification error traning of Hidden Markov Models for handwriting recognition," in *Proc. of ICASSP 2001*, 2001.

[6] C. J. Leggetter and P. C. Woodland, "Maximum likelihood linear regression for speaker adaptation of continuous density hidden Markov models," *Computer Speech and Language*, vol. 9, pp. 171–185, 1995.

[7] B.-H. Juang and S. Katagiri, "Discriminative learning for minimum error classification," *IEEE Trans. on Signal Processing*, vol. 40, no. 12, pp. 3043–3054, 1992.

[8] S. Young et al., *The HTK Book (for HTK V3.0)*, July 2000.

[9] Q. Huo and B. Ma, "On-line adaptive learning of continuous density hidden Markov models based on multiple-stream prior evolution and posterior pooling," *IEEE Trans. on SAP*, vol. 9, pp. 388–398, 2001.