Continuous Speech Recognition with Penalized Logistic Regression Machines

Øystein Birkenes\textsuperscript{ab}, Tomoko Matsui\textsuperscript{a}, Kunio Tanabe\textsuperscript{c}, and Tor André Myrvoll\textsuperscript{ab}

\textsuperscript{a}The Institute of Statistical Mathematics, Tokyo, Japan
\textsuperscript{b}Department of Electronics and Telecommunications, NTNU, Trondheim, Norway
\textsuperscript{c}Waseda University, Tokyo, Japan
{birkenes,myrvoll}@iet.ntnu.no, tmatsui@ism.ac.jp, tanabe.kunio@waseda.jp

ABSTRACT

Penalized logistic regression machines (PLRMs) have recently been shown to give good performance on isolated word speech recognition. In this paper, we extend this framework to continuous speech recognition. We present two approaches that both make use of the output from an HMM Viterbi recognizer. The first approach performs probabilistic prediction with PLRM on the segments obtained from the HMM Viterbi recognizer. The resulting subwords and subword probabilities are combined to form a sentence and a sentence probability, respectively. In the second approach, an N-best list generated by the HMM Viterbi recognizer is rescored using PLRM. Experiments on the Aurora2 connected digits database show that both approaches outperform the baseline HMM Viterbi recognizer.

1. INTRODUCTION

The penalized logistic regression machine (PLRM) \cite{1,2,3} is a statistical learning machine for multiclass classification. It is a discriminative classification approach in the sense that it models the conditional probability for each class given the evidence of the test example.

Recently, a PLRM for isolated word speech recognition was presented \cite{4}. In this approach, a set of hidden Markov models (HMMs) used as regressors are trained discriminatively together with the weight matrix of the PLRM using a penalized likelihood criterion. During recognition of a speech signal, estimates of the conditional probabilities for each word in the vocabulary given the speech signal are obtained. The recognized word is then simply the one with the highest conditional probability. Experiments on the TH 46 E-set database demonstrated the power of this approach. The error rate was reduced by 72.6% relative to the baseline HMM Viterbi recognizer.

In this paper, we extend the work in \cite{4} and present two approaches for continuous speech recognition with PLRM. Both approaches make use of an HMM Viterbi recognizer. The first approach uses the HMM Viterbi recognizer only for segmentation. Each segment is then treated in isolation and probabilistic prediction is done as in \cite{4}. The geometric mean of the estimated subword probabilities is taken as an estimate of the conditional probability of the sentence given the utterance. In the second approach, an N-best list generated by the HMM Viterbi recognizer is rescored using a PLRM to compute an estimate of the conditional probability of each sentence hypothesis in the N-best list. Experiments conducted on the Aurora2 connected digits database show improvements of both approaches relative to the baseline HMM Viterbi recognizer.

The organization of the paper is as follows. In the next section we review isolated word speech recognition with PLRM. Extensions to continuous speech recognition are presented in Sec. 3. In Sec. 4 we present experimental results on the Aurora2 connected digits database, before we state the conclusion and discuss future work in Sec. 5.

2. THE PLRM AND ITS APPLICATION TO ISOLATED-WORD SPEECH RECOGNITION

We start off by giving an overview of the Penalized Logistic Regression Machine (PLRM), before we review its application to isolated word speech recognition.

2.1. The Penalized Logistic Regression Machine

Let \((x, y) \in \mathcal{X} \times \mathcal{Y}\) be a random pair drawn according to an unknown probability distribution \(p(x, y)\). In classification, the label set \(\mathcal{Y}\) is finite, and the goal is to find a mapping \(h : \mathcal{X} \rightarrow \mathcal{Y}\) that gives good prediction on any feature \(x \in \mathcal{X}\). Let \(K\) denote the number of classes and let each class be represented as an integer in the set \(\mathcal{Y} = \{1, \ldots, K\}\).

The penalized logistic regression machine (PLRM) \cite{1,2,3} makes an estimate \(\hat{p}(y|x)\) of the conditional probability distribution based on a set of training examples \(\mathcal{D} = \{(x^{(n)}, y^{(n)})\}_{n=1}^{N}\). Prediction on a given feature \(x\) is

\[\hat{y} = \arg\max_{y \in \mathcal{Y}} \hat{p}(y|x).\]  (1)

The conditional distribution for each class \(k\) is modeled using a nonlinear logistic regression model with parameter vector \(w_k\), i.e.,

\[\hat{p}_k = \hat{p}_k(x; w_k) = \frac{\exp w_k^T \phi(x; \Lambda)}{\sum_{l=1}^{K} \exp w_l^T \phi(x; \Lambda)},\]  (2)
where \( \phi(x; \Lambda) = [1, \phi(x; \lambda_1), \ldots, \phi(x; \lambda_M)]^T \) is a vector of \( M \) regressors augmented by the scalar 1. Each regressor \( \phi(x; \lambda_m) \) has a hyperparameter vector \( \lambda_m \). Let the matrix \( \Lambda \) with columns \( \lambda_m \) denote the set of all such hyperparameters. Similarly, let the matrix \( W \) with rows \( w_k \) denote the set of all the parameters of the model. The inclusion of 1 as the first element of \( \phi(x; \Lambda) \) ensures that the discriminant functions \( f_k = w_k^T \phi(x; \Lambda) \) are affine transformations of the \( M \) regressors, i.e., that one of the terms in each scalar product is a bias term. Fig. 1 illustrates the nonlinear logistic regression model in PLRM.

![Fig. 1. The nonlinear logistic regression model in PLRM.](image)

Given a set of training data \( D = \{(x^{(n)}, y^{(n)})\}_{n=1}^N \), the parameter matrix \( W \) is estimated by minimizing

\[
\mathcal{P}^\delta(W; D) = -\sum_{n=1}^N \log \hat{p}_{\phi}^{(n)} + \frac{\delta}{2} \text{trace} \Gamma W \Sigma W^T, \tag{3}
\]

where the first term is the negative logarithm of the logistic regression likelihood, and the second term is a penalty term weighted by a hyperparameter \( \delta > 0 \). The matrix \( \Gamma \) is a \( K \times K \) diagonal matrix whose \( k \)th diagonal element is the fraction of training samples with the \( k \)th class label. The purpose of \( \Gamma \) is to compensate for possible differences in the number of training samples from each class. The matrix \( \Sigma = \Sigma(\Lambda) \) is an \((M+1) \times (M+1)\) positive definite matrix.

In this paper we let \( \Sigma \) be the sample moment matrix of the mapped feature data, that is, \( \Sigma = 1/N \Phi \Phi^T \), where \( \Phi \) is an \((M+1) \times N\) matrix with columns \( \phi(x^{(n)}, \Lambda) \).

The criterion function (3) is convex with respect to \( W \) [1]. In [3], the author presented an efficient algorithm for minimization of (3) that combines Newton’s method with the conjugate gradient method. After choosing an initial weight matrix \( W^0 \), the iteration step in Newton’s method is

\[
W^{i+1} = W^i - \alpha \Delta W^i, \quad (\alpha > 0). \tag{4}
\]

The update matrix \( \Delta W^i \) is the solution to

\[
\sum_{n=1}^N \left( \text{diag} \hat{p}^{(n)} - \hat{p}^{(n)} \hat{p}^{(n)T} \right) \Delta W^i \phi^{(n)} \phi^{(n)T} + \delta \Gamma \Delta W^i \Sigma = (P - Y) \Phi^T + \delta \Gamma W^i \Sigma, \tag{5}
\]

where \( \hat{p}^{(n)} \) is a \( K \)-dimensional vector whose elements are \( \hat{p}_k(x^{(n)}; w_k) \), \( P \) is a \( K \times N \) matrix whose columns are \( \hat{p}^{(n)} \), \( Y \) is a \( K \times N \) matrix where the \( n \)th column has all zeros except for a 1 in the position corresponding to the class label \( y^{(n)} \), and \( \phi^{(n)} = \phi(x^{(n)}; \Lambda) \). This equation can be solved using the conjugate gradient method. For more details see [3].

### 2.2. Isolated-Word Speech Recognition with PLRM

In isolated word speech recognition the goal is to map each feature representation \( x \) of a speech signal into one of the words \( y \) in a vocabulary \( \mathcal{V} \). In order to use PLRM for isolated word speech recognition, we need to define the mapping \( \phi \). It was proposed in [4] to use a mapping involving likelihoods of hidden Markov models (HMMs) as regressors. Here we choose

\[
x \mapsto \phi(x; \Lambda) = [1, \phi(x; \lambda_1), \ldots, \phi(x; \lambda_K)]^T, \tag{6}
\]

where \( \phi(x; \lambda_k) \) is the frame-normalized log-likelihood of the HMM with parameter \( \lambda_k \) corresponding to the \( k \)th word, with \( \lambda_k \) being the \( k \)th column of the matrix \( \Lambda \). Thus, if \( \lambda \) denotes the number of parameters in each HMM, \( \Lambda \) is a \( \lambda \times K \) matrix of all the HMM parameters. To be more specific with our choice of nonlinear mapping, let \( x = (x_1, \ldots, x_T) \) be a sequence of \( T \) feature vectors. Furthermore, let \( \lambda = (\pi, A, b) \) denote the parameters of an HMM; \( \pi = [\pi_n] \) is the vector of initial state probabilities, \( A = [a_{n,j}] \) is the transition probability matrix, and \( b = \{b_n(x)\} \) is the collection of state-conditional pdfs. Then

\[
\phi(x; \lambda) = \frac{1}{T} \log \max_q \prod_{t=1}^T a_{q_{t-1}, q_t} b_{q_t}(x_t). \tag{7}
\]

where \( q = (q_0, \ldots, q_T) \) is a state sequence.

To gain additional discriminative power, it was proposed in [4] to treat \( \Lambda \) as a parameter of the model (2) instead of just a hyperparameter of the nonlinear mapping \( \phi \). Thus, the parameters of the model are \( \theta = (W, \Lambda) \), and we are interested in finding the pair of matrices \( \theta^* = (W^*, \Lambda^*) \) that minimizes the criterion in (3), i.e.,

\[
\theta^* = \arg \min_{\theta=(W, \Lambda)} \mathcal{P}^\delta_\theta (\theta; D). \tag{8}
\]

Although the function in (3) is convex with respect to \( W \), it is not guaranteed to be convex with respect to \( \Lambda \). A local minimum can be obtained by using a coordinate descent method with coordinates \( W \) and \( \Lambda \). The algorithm is initialized with \( \Lambda_0 \), for which a reasonable choice is the ML estimate of the HMM parameters. Then the initial weight matrix can be found as

\[
W_0 = \arg \min_W \mathcal{P}^\delta_\theta (W, \Lambda_0; D). \tag{9}
\]

The iteration step is as follows:

\[
\Lambda_{i+1} = \arg \min_\Lambda \mathcal{P}^\delta_\theta (W_i, \Lambda; D), \tag{10a}
\]

\[
W_{i+1} = \arg \min_W \mathcal{P}^\delta_\theta (W, \Lambda_{i+1}; D). \tag{10b}
\]

\(^1\)This is actually an approximation to the frame-normalized log-likelihood of an HMM. Nevertheless, since this is a common approximation in the speech literature, we refer to this simply as the frame-normalized log-likelihood.
For the convex minimization with respect to $W$, a good algorithm is given by (4) and (5). As for the minimization with respect to $\Lambda$, it was proposed in [4] to use a steepest descent method.

3. CONTINUOUS SPEECH RECOGNITION WITH PLRM

In continuous speech recognition the class label $y$ is a sequences of subwords (e.g., phonemes or digits), or a sentence, instead of just a single word. Mathematically, if $\mathcal{V}$ denotes the set of subwords, the label set for a particular sentence $y$ with $L$ mutually independent subwords is $\mathcal{V}^L$. The size of this label set is then $|\mathcal{V}|^L$, a quantity which grows exponentially with the number of subwords $L$. Thus, even a small $L$ can induce a very large number of classes. Consider for example the recognition of 8-digit phone numbers, where $\mathcal{V}$ is the set of digits from 0–9 and $L = 8$. Then the number of classes is $|\mathcal{V}|^L = 10^8$. Naively applying PLRM with such a large number of classes would be extremely difficult, if not impossible. Therefore, alternative approaches are called for. In the following, two approaches for continuous speech recognition with PLRM are presented. They both use the same model and training procedure, and differ only in the way prediction is done.

3.1. PLRM Training

Assume that we somehow knew the segment boundaries of the speech signals. Then we could train the PLRM by applying the procedure presented in the previous section on segments instead of the whole utterances. Usually however, we do not know the segment boundaries for the training data, only the orthographic transcription. Then, the most straightforward thing to do would be to estimate the segment boundaries. For this, we will make use of a set of subword HMMs to perform Viterbi forced alignment segmentation. These HMMs are trained using the standard Baum–Welch re-estimation method [5]. Thus, for an orthographically labeled pair $(x, y)$ we obtain a set of subword labeled segments $\{(s_1, v_1), \ldots, (s_L, v_L)\}$, where each $s_l = (x_{T_{l-1}}, \ldots, x_{T_l})$ is a segment of $x$, i.e., a connected subsequence of the feature vector sequence $x$, and $v_l \in \mathcal{V}$ is the corresponding subword label. The training data then comprises the subword labeled segments of all the orthographically labeled sentences $\{(x^{(n)}, y^{(n)})\}$, i.e.,

$$\mathcal{D} = \{(s_l^{(n)}, v_l^{(n)})\}_{n=1, \ldots, N, l=1, \ldots, L^{(n)}},$$

where $L^{(n)}$ is the number of subwords in $y^{(n)}$.

Note that we are dealing with two different sets of subword HMMs. One set is used to perform forced alignment segmentation. This set contains a model for each subword, and preferably a model for the silence at the beginning and the end of the utterances, as well as a model for the short pauses that sometimes occur between subwords. These HMMs are kept fixed during PLRM training. The mapping $\phi$ in PLRM uses a different set of subword HMMs, which consists of one HMM for each subword. These models are used to compute the vectors $\phi(s; \Lambda)$, and their parameters $\Lambda$ are updated in the training procedure along with the weight matrix $W$.

3.2. PLRM Prediction

We propose two approaches for prediction that both make use of the HMM Viterbi recognizer used in the segmentation of the training data. The first approach uses only the segment borders from the HMM Viterbi recognizer and will be referred to as segment PLRM. The second approach is based on N-best rescoring.

3.2.1. Segment PLRM

In segment PLRM, each test utterance $x$ is segmented using the HMM Viterbi recognizer. Then, for each segment $s_l$ of the test utterance, PLRM prediction yields an estimated subword $\hat{v}_l$ along with its estimated conditional probability $p_{\hat{v}_l}$. The concatenation of the subwords gives rise to a hypothesized sentence $\hat{y} = (\hat{v}_1, \ldots, \hat{v}_L)$. Moreover, the geometric mean of the subword probabilities may serve as a confidence measure $\hat{p}_y$ for the sentence, namely

$$\hat{p}_y = \left(\prod_{l=1}^L p_{\hat{v}_l}\right)^{1/L}. \quad (12)$$

3.2.2. N-best rescoring with PLRM

For a test utterance $x$, an N-Best list of sentence hypothesis is generated using the HMM Viterbi recognizer. Each sentence $\hat{y} = (\hat{v}_1, \ldots, \hat{v}_L)$ in the N-Best list is accompanied with its most likely segmentation as well as the segment likelihood scores. Through PLRM prediction on each segment $s_l$, we obtain a conditional probability $p_{\hat{v}_l}$ of the subword $\hat{v}_l$. Combining these probabilities according to (12) gives rise to a score for each sentence hypothesis in the N-Best list. The N-Best list can then be reordered based on these sentence scores.

In this approach however, we risk getting many deletion and insertion errors. To see this, recall that a sentence hypothesis with a deletion error has one less subword than the correct sentence, which usually means that one of the corresponding segments spans two correct segments. For an insertion error, the hypothesis has one additional subword compared to the correct sentence. This usually means that a correct segment has been split into two. Such incorrect segments $s$, and their corresponding nonlinear mapping $\phi(s; \Lambda)$, have not been seen by the machine during training. Their domain typically lies outside the domain for correct segments, and the machine therefore tends to give very high probability to one subword (often incorrect) and very low probability to the others. For this reason, we choose to rescore only those hypotheses in the N-Best list that have the same number of subwords as the first hypothesis in the list. Thus, with this approach, the machine cannot find the correct sentence if the first hypothesis in the N-Best list does not contain the correct number of subwords. The same approach was taken in [6], where the authors rescored N-Best lists using the support vector machine (SVM).
4. EXPERIMENTS

Experiments were conducted on the Aurora2 connected digits database. This is a database of utterances from many different speakers of digit strings with lengths 1–7 digits. Each of the digits 1–9 was associated with one class, while 0 was associated with two classes reflecting the pronunciations “zero” and “oh”. The total number of classes was thus $K = 11$. For each of the 11 digit classes, we used an HMM with 16 states and 3 mixtures per state. These HMM topologies were the same for the HMM Viterbi recognizer as well as for the set of HMMs used to compute $\phi(s; \Lambda)$ in PLRM. In addition, for the HMM Viterbi recognizer, we used a silence sil model with 3 states and 6 mixtures per state, and a short pause sp model with 1 state and 6 mixtures. These HMM topologies are the same as the ones defined in the training script distributed with the database. Training of the HMMs was done using this same script on the clean condition training set consisting of 8440 utterances.

In the training of the PLRM we updated only the means of the HMMs while keeping the other HMM parameters fixed. For each coordinate descent iteration (10a, 10b) we iterated 5 times in the steepest descent method and 3 times in the Newton method. At each step in the steepest descent method, the stepsize was either doubled or halved, depending on whether the previous step resulted in a decreased or an increased value of the criterion. The stepsize in the Newton method was kept fixed at $1.0$.

For the test set we chose the clean data of test set a. This set consists of 4004 digit strings with lengths 1–7 digits uttered in clean conditions.

Table 1 shows the sentence accuracy on the test set for the baseline HMM Viterbi recognizer, compared with segment PLRM with $\delta = 1000$ and 5-best rescoring with $\delta = 100$. Both approaches reduce the error rate of the baseline system with 11.1%.

<table>
<thead>
<tr>
<th>Table 1. Sentence Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
</tr>
<tr>
<td>96.85</td>
</tr>
</tbody>
</table>

Figure 2 shows the sentence accuracy on the test set using segment PLRM and 5-best rescoring, respectively, for a wide range of $\delta$ values. Both approaches outperform the baseline system for all of these $\delta$ values. This suggests that the choice of $\delta$ is not critical for good performance.

5. CONCLUSION AND FUTURE WORK

We have proposed two approaches for continuous speech recognition with penalized logistic regression machines. Both approaches outperformed an HMM Viterbi speech recognizer on experiments conducted on the Aurora2 connected digits database.

The two approaches suffer somewhat from incorrect segmentation, sometimes giving very high probability to incorrect segments. This is due to the lack of incorrect segments in the training data. Therefore, a topic for future work is to include a garbage class and train this using incorrect segments.

6. REFERENCES