STREAM WEIGHT TUNING IN DYNAMIC BAYESIAN NETWORKS

Arthur Kantor

University of Illinois at Urbana Champaign
Department of Computer Science
201 N Goodwin Ave
Urbana, IL 61801

Mark Hasegawa-Johnson

University of Illinois at Urbana-Champaign
ECE Department
405 N Mathews
Urbana, IL 61801

ABSTRACT

In this paper we present a family of algorithms for estimating stream weights for dynamic Bayesian networks with multiple observation streams. For the 2 stream case, we present a weight tuning algorithm optimal in the minimum classification error sense. We compare the algorithms to brute-force search where feasible, as well as to previously published algorithms and show that the algorithms perform as well as brute-force search and outperform previously published algorithms. We test the stream weight tuning algorithm in the context of speech recognition with distinctive feature tandem models. We analyze how the criterion used for weight tuning differs from the standard word error rate criterion used in speech recognition.

Index Terms—
Speech recognition

1. INTRODUCTION

This paper concerns itself with the observation model $p(o|q)$, where o is the observation vector in a given frame, and q is the discrete hidden state in that same frame. In some models, the observation vector o is partitioned into K disjoint sets $s_1, s_2, ..., s_K$ which are conditionally independent given the hidden state q, so $log(p(o|q)) = \sum_{j=1}^{K} log(p(s_j|q))$. The sets s_j are called *streams*. This kind of independence assumption is appropriate when the observation consists of both audio and video, or when the streams are expected to contain complimentary information. Alternative independence assumptions are explored in [1]. It may happen that some streams are better predictors of the hidden variable q than others, so a weighted linear combination of the stream contributions is appropriate:

$$log(p(o|q)) = \sum_{j=1}^{K} \lambda_j log(p(s_j|q)) \quad (1)$$

where λ is the weight vector. The use of stream weights in Eq. 1 is equivalent to exponentiating $p(s_j|q)$ prior to computing $p(o|q)$. The resulting $p(o|q)$ is a correctly normalized PDF only if $\lambda = 1$ (the vector of ones), but values of $\lambda \neq 1$ may have better discriminative accuracy. In particular, Iwano et al. [5] proposed treating Eq. 1 as the equation of a two-class linear classifier, and using LDA to find λ. Other approaches have also been suggested [4].

This paper proposes treating Eq. 1 as the equation of a one-class linear classifier. We find that the one-class classifier may be effectively trained using a wider variety of loss functions than the two-class classifier. In particular, the "mean classifier" (linear loss criterion) works quite well for a 2-stream recognizer, while the one-class SVM works well even for 9 streams. Sec. 2 reviews the 2-class LDA training algorithm for λ proposed by [5] while Sec. 3 reformulates the problem as a 1-class λ classification problem. Sec. 3.1 presents an algorithm to find globally optimal stream weights for the case of 2 streams and Sec. 3.2 describes a family of approximate training algorithms with an arbitrary loss function for any number of streams. Sec. 3.3 proposes the linear loss function, computes the analytically exact solution, and demonstrates resulting error bounds on the computed weight vector for any particular training corpus size. Experimental methods are described in Sec. 4.1, and results are given in Sec. 4.2.

2. CHOOSING λ USING A 2-CLASS CLASSIFIER

Iwano et al. proposed finding the value of λ that minimizes the expected number of incorrectly labeled phone segments in the training corpus:

$$\lambda^* = \arg\min_{\lambda} \sum_{i=1}^{N} [h_\lambda(O_i) \neq w_i^*] \quad (2)$$

where N is the number of phone segments (or examples) in the training corpus, $w_i^* \in W$ is the correct label of phone segment i, and $O_i = [o_{i1}, \ldots, o_{iT_i}] \in \mathbb{R}^*$ is its observation sequence with one column per frame and T_i frames. $h_\lambda : \mathbb{R}^* \rightarrow W$ is a classifier which depends on the weight vector λ and maps from a sequence of observation frames to a phone label $w \in W$. $[p]$ is the indicator function, defined to equal 1 if the proposition p is true, and 0 otherwise.

The classifier $h_\lambda(O_i)$ is a maximum likelihood classifier

$$h_\lambda(O_i) \quad = \quad \arg\max_{w \in W} \sum_{j=1}^{K} \lambda_j \log p(S_{ij}|w) \tag{3}$$

where $S_{ij} = [s_{ij1}, \ldots, s_{ijT_i}]$ is a set of rows from matrix O_i. If we ignore HMM transition probabilities, then $\log p(S_{ij}|w) = \sum_{t=1}^{T_i} \log p(s_{ijt}|q_t)$, where q_t is any (hopefully reasonable) frame alignment of the states composing phone w. Combining equations gives

$$\lambda^* = \arg\min_{\lambda} \sum_{i=1}^{N} \left[\max_{w \in W \setminus w_i^*} \left(\sum_{j=1}^{K} \lambda_j \log p(S_{ij}|w) \right) \geq \sum_{j=1}^{K} \lambda_j \log p(S_{ij}|w_i^*) \right] \tag{4}$$

The minimization in Eq. 4 defines a linear two-class classifier, where the features are $\log p(S_{ij}|w)$, and the two classes are $w_i = w_i^*$ and $w_i \neq w_i^*$. Exact minimization is difficult, but many useful approximate methods exist: Iwano et al. [5] suggested using LDA to solve an equation related to Eq. 4.

3. CHOOSING λ USING A 1-CLASS CLASSIFIER

This paper proposes converting Eq. 4 into a one-class linear classification problem, so that it can be more exactly minimized. Eq. 4 can be rewritten into:

$$\lambda^* = \arg\min_{\lambda} \sum_{i=1}^{N} \left[\max_{w \in W \setminus w_i^*} \lambda^T d_{iw} \geq 0 \right] \tag{5}$$

where $d_{iwj} = (\log p(S_{ij}|w) - \log p(S_{ij}|w_i^*))$ and the vector $d_{iw} = [d_{iw1}, \ldots, d_{iwK}]^T$. Each i such that $\max_w \lambda^T d_{iw} \geq 0$ is an example i wrongly classified as token w. The goal is to have all the d_{iw}'s on one side of the hyperplane which passes through the origin and normal to λ. However, if we chose to give up on one particular d_{ip} as a lost cause, we should also ignore all other d_{iw} such that $w \in W \setminus \{p, w_i^*\}$ in the optimization of λ.

So far, we have defined W to be the set of phones, but W can consist of words, frames or tokens of some intermediate duration.

3.1. Optimal solution for the case of 2 streams

If we have only 2 streams ($K = 2$) we can derive the globally optimal λ^* that minimizes the token classification error. In this case, λ^* is only a function of the hyperplane angle, and the objective (the number of misclassified examples) can change only if a d_{iw} moves from one side of the hyperplane to the other as the hyperplane rotates. This suggests the following algorithm: 1) Sort all d_{iw}'s by their angle 2) rotate the hyperplane while tracking the number of examples on the wrong side of the hyperplane 3) pick the angle of the hyperplane with minimum number of examples on the wrong side. The complexity of the algorithm is dominated by the sort.

3.2. Approximate solution for the case of more than 2 streams

If $K > 2$, solving the combinatorial optimization problem in Eq. 5 is hard. We can replace the 0-1 loss with the sigmoid loss and the $\max(X)$ with the p-norm $(\sum_{x \in X} x^p)^{1/p}$, $p > 1$ and then locally optimize with generalized probabilistic descent, as in standard MCE parameter training [8]. We choose instead to focus on simplifications of Eq. (5) that are quadratic, and therefore have computable global optima. We simplify Eq. 5 as

$$\lambda^* = \arg\min_{\lambda} \sum_{i=1}^{N} \sum_{w \in W \setminus w_i^*} \mathbb{L}\left(\lambda^T d_{iw}\right) \tag{6}$$

Where $\mathbb{L}$ is a loss function that depends on the classification algorithm. Eq. 6 differs from Eq. 5 in two ways. In (6) we are over-counting the error on any example i by considering *every* d_{iw} instead of only the worst-offending d_{iw}. Also, in (5) we are using the 0-1 loss function, while in (6) we choose different loss functions which yield computationally tractable classifiers.

We have experimented with three kinds of classifiers: 1) The 'mean' classifier with linear loss function, 2) the linear kernel SVM [6] classifier with the hinge loss function and 3) the LDA classifier. The mean classifier sets λ^* to the negative mean of the d_{iw}'s. The mean classifier is simple and allows us to put confidence limits on λ^* as a function of amount of training data (see Sec. 3.3). The SVM classifier has good generalization properties and the LDA classifier has been used in previous work on stream-weight tuning [5].

While the classifier with the 0-1 loss function is insensitive to the magnitude of d_{iw}, the above classifiers are sensitive to its magnitude, and the λ's they suggest will therefore be dominated by large-magnitude d_{iw}s. To counteract this, we normalize the d_{iw}'s so Eq. 6 becomes

$$\lambda^* = \arg\min_{\lambda} \sum_{i=1}^{N} \sum_{w \in W \setminus w_i^*} \mathbb{L}\left(\frac{\lambda^T d_{iw}}{\|d_{iw}\|}\right) \tag{7}$$

where $\| \bullet \|$ is the l_2 norm. In our experiments, normalization was critical for the algorithm's good performance.

3.3. Linear loss and the mean classifier

If the loss function $\mathbb{L}(x)$ is chosen to have a simple enough form, it is possible to analytically derive the PDF of the statistic λ^* specified by Eq. 6. For example, if we let $\mathbb{L}(x) = x$

and constrain $|\lambda|^2 = K$ for some constant K, the constrained optimum is obtained as

$$\begin{aligned}\lambda^* &= \arg\min_{\lambda} \sum_{i=1}^{N} \sum_{w \in W \setminus w_i^*} \lambda^T d_{iw} + \frac{\alpha}{2}\left(\lambda^T\lambda - K\right) \\ &= -\frac{1}{\alpha N(|W|-1)} \sum_{i=1}^{N} \sum_{w \in W \setminus w_i^*} d_{iw},\end{aligned} \tag{8}$$

where α is a Lagrange multiplier. The value of K is unimportant, as long as it is nonzero; for convenience, therefore, we can choose $\alpha = 1$, so that λ^* is the negative mean of the d_{iw}s. We call this solution the "mean classifier."

We can derive a confidence angle θ for the mean classifier and guarantee that with some probability the true weight vector differs from the predicted weight vector by no more than θ. This confidence angle can be used to estimate the amount of training data necessary to accurately determine the weight vector. Since calculating the d_{iw}'s is the most time consuming part of estimating λ, it is useful to know how much training data is actually needed.

Because of the central limit theorem, the statistic λ^* defined in Eq. 8 has an approximately Gaussian distribution, with a mean equal to $\bar{\lambda} = -\bar{d} = -E[d_{iw}]$. The vectors d_{iw} may be assumed to be identically distributed, but they may not be assumed to be independent. Under this condition, the covariance of random variable λ^* is equal to

$$\Sigma_{\lambda^*} = \frac{1}{|N(|W|-1)|^2} \sum_{i,j,w,p} E\left[(d_{iw} - \bar{d})(d_{jp} - \bar{d})^T\right] \tag{9}$$

In practice, we find that the log probabilities of different temporal segments tend to be uncorrelated ($E[(d_{iw} - \bar{d})(d_{jp} - \bar{d})^T] \approx 0$, $i \neq j$), but that different candidate labelings of the same temporal segment tend to have a very high correlation. If we assume that $E[(d_{iw} - \bar{d})(d_{ip} - \bar{d})^T] \approx \Sigma_d$, where Σ_d is the covariance matrix of vector d_{iw}, then Eq. 9 may be simplified to $\Sigma_{\lambda^*} \approx \Sigma_d/N$.

With probability $p \approx 0.95$ (2 standard deviations), the unknown optimum classifier vector $\bar{\lambda} = -\bar{d}$ lies within the elliptical confidence region $(\lambda^* - \bar{\lambda})^T(\Sigma_d/N)^{-1}(\lambda^* - \bar{\lambda}) < 4$. Therefore, with probability $p' > 0.95$, $\bar{\lambda}$ lies inside a sphere of radius $r = 2\sigma/\sqrt{N}$ centered at λ^*, where σ^2 is the largest eigenvalue of Σ_d. Therefore, with probability $p'' \geq p' > 0.95$, the angle $\theta = \arccos(\frac{\bar{\lambda}^T\lambda^*}{\|\bar{\lambda}\|\|\lambda^*\|})$ between the measured classifier vector λ^* and the true optimum classifier vector $\bar{\lambda}$ is $\theta \leq \arcsin(\frac{r}{\|\lambda\|})$. Given a desired confidence angle of θ_{max}, the number of training segments necessary to estimate λ^* is therefore $N \geq 4\sigma^2/\|\lambda\|^2 \sin^2\theta_{max}$.

We computed the 95% confidence angles for bootstrapped samples of $N \approx 1$ million, $N \approx 28$ million and $N \approx 40$ million. The angles between λ's derived from bootstrapped samples were all smaller than the 95% confidence angles.

4. EXPERIMENTS

4.1. Distinctive Feature based Tandem Models

We have tested our weight tuning algorithm in the context of distinctive-feature-based tandem models [3]. The 'tandem' approach involves first training a multi-layer perceptron (MLP) to perform phone classification at the frame level, and then using the suitably transformed frame-level phone posterior estimates of the MLP together with standard features such as perceptual linear prediction coefficients as the observations in HMMs. In our experiments we trained eight MLPs, with each MLP trained to compute the value, in one frame, of a particular phonological distinctive feature. The eight distinctive features were place, manner, nasality, glottal state, rounding, vowel identity, vowel height and vowel frontness. The number of states ranged from 3 to 23. The MLPs were trained on the Fisher corpus. The MLP outputs are further transformed to make them suitable for being modeled using a mixture Gaussian PDF as described in [3].

We experimented, with 2 streams and 9 streams of observations. In the 9-stream experiment, streams included the eight MLP outputs, and the PLP coefficients. In the 2-stream experiment, all 8 MLP output vectors were combined into a single stream exactly as in [3].

The speech recognizer uses 55 monophone HMMs, with up to three states per monophone and a total of 132 sub-phone states (SPS) with up to 128 Gaussians per state. All training and testing was done using the GMTK toolkit [2].

Except for the training of MLPs, the training, development and test data used in these experiments are as specified by subtask 1 of the 500-word vocabulary Svitchboard task [9]. Words in this database are highly reduced, therefore WERs tend to be above 60% [7].

For each number of streams, we experimented with two token sets: words, and sub-phone states (SPS). The size of the word token set is 501 (500 words plus silence) and the size of the SPS token set is 132. For each token set, we obtained forced alignments of the training data using the 2-stream model with equal stream weights. For each force-aligned example i, we calculated $\log p(S_{ij}|w)$, the log-likelihood of stream j given the token w, by setting the stream weight $\lambda_j = 1$ and all other stream weights to 0.

The training vectors $\frac{d_{iw}}{\|d_{iw}\|}$ were used in three experimental systems (the mean classifier, LDA, and SVM), but not in either of the two baseline systems. Both the 9-stream and 2-stream models used a two-class LDA with un-normalized log probability observation vectors (exactly as described by [5]) as a baseline. Stream weights for the 2-stream recognizer were also estimated by using a grid search to explicitly minimize development set WER.

For each set of weights λ, we locally optimized the WER as a function language model penalty and scale on a subset of the development dataset using the amoeba algorithm [10].

model	token set	$\lambda = 1$	grid search	mean	LDA	SVM	LDA [5]
2-stream		60.5 (1.0)	60.2 (1.0525)				
	word			60.6 (1.1509)	60.0 (1.0611)	60.5 (1.1940)	68.4 (0.0)
	sub-phone state			60.2 (1.0489)	60.8 (1.0355)	60.2 (1.9987)	60.0 (1.8493)
9-stream		64.1					
	word			63.8	61.9	62.2	68.5
	sub-phone state			64.0	62.1	61.1	70.0

Table 1. WER of recognizers using the stream weights λ estimated by various strategies. For 2-stream model, the number in parentheses is the ratio of MLP to PLP stream weights.

The λ along with the optimal language model penalty and scale were then used for Viterbi decoding of the test set.

4.2. Results

In table 1, the $\lambda = 1$ column shows the baseline WER if the stream weights are set to 1 (no tuning). The results are similar to previous work [7] reported on the same data set. The grid search was only practical for the 2-stream model. In the grid search, the PLP weight was fixed at 1, and the MLP weight varied from 0 to 1.5. Minimum dev-set WER within this range was achieved with the MLP weight set to 1.0525.

In addition to the grid search baseline, we implemented the LDA algorithm as described in [5], which uses LDA to find a separator between the correct and incorrect $\log p(S_{ij}|w)$ vectors ("two-class LDA"). If any of the resulting weights are negative, they are truncated to 0. As an additional experiment, we used LDA to separate the normalized d_{iw}'s from the origin ("one-class LDA"), and allowed negative weights. The results for two-class and one-class LDA are in the 'LDA [5]' and 'LDA' columns of table 1 respectively.

Table 1 shows that stream weight tuning using an accurate classifier is particularly important as the number of streams increases.

5. CONCLUSION

We have presented a globally optimal algorithm for stream weight tuning for 2 streams, and a family of approximate stream weight algorithms for an arbitrary number of streams. Using the algorithm with the SVM classifier applied to short duration tokens yields lower WER than no weight tuning, with improvements becoming more significant as the number of streams increases.

6. REFERENCES

[1] J. Bilmes and K. Kirchhoff. Directed graphical models of classifier combination: Application to phone recognition. In *ICSLP*, 2000.

[2] J. Bilmes and G. Zweig. The graphical models toolkit: An open source software system for speech and time-series processing, 2002.

[3] O. Cetin, A. Kantor, S. King, C. Bartels, M. Magimai-Doss, J. Frankel, S. King, and K. Livescu. An articulatory feature-based tandem approach and factored tandem observation modeling. In *ICASSP*, 2007.

[4] G. Gravier, S. Axelrod, G. Potamianos, and C. Neti. Maximum entropy and MCE based HMM stream weight estimation for audio-visual ASR. In *ICASSP*, 2002.

[5] K. Iwano, K. Kojima, and S. Furui. A weight estimation method using LDA for multi-band speech recognition. In *ICSLP*, volume 7, pages 2534–2537, 2006.

[6] T. Joachims. Training linear svms in linear time. In *KDD*, pages 217–226, 2006.

[7] O. Cetin K. Livescu, M. Hasegawa-Johnson, S. King, C. Bartels, N. Borges, A. Kantor, P. Lal, L. Yung, A. Bezman, S. Dawson-Haggerty, B. Woods, J. Frankel, M. Magimai-Doss, and K. Saenko. Articulatory feature-based methods for acoustic and audio-visual speech recognition: Summary from the 2006 JHU summer workshop. In *ICASSP*, 2007.

[8] S. Katagiri, B. Juang, and C.H. Lee. Pattern recognition using a family of design algorithm based upon the generalized probabilistic descent method. *Proc. of IEEE*, 1998.

[9] S. King, C. Bartels, and J. Bilmes. Svitchboard 1: Small vocabulary tasks from switchboard 1. In *Interspeech*, 2005.

[10] J.A. Nelder and R. Mead. A simplex method for function minimization. *Computer J.*, 7, 1965.