PRODUCT OF GAUSSIANS AND MULTIPLE STREAM SYSTEMS

S.S. Airey and M.J.F. Gales

Cambridge University Engineering Department
Trumpington Street, Cambridge, CB2 1PZ, UK
{ssa26,mjfg}@eng.cam.ac.uk

ABSTRACT

Recently there has been interest in the use of classifiers based on the product of experts (PoE). PoEs offer an alternative to the standard mixture of experts (MoE) framework. This paper presents a particular form of PoE, the product of Gaussians (PoG), within an hidden Markov model framework. Training and initialisation procedures are described for this PoG system. In addition, the relationship of PoG to standard multiple stream systems is explored. The PoG system performance is examined on the SwitchBoard task and is compared to standard Gaussian mixture systems and multiple stream systems.

1. INTRODUCTION

Mixture of Gaussians (MoG) are commonly used as the state representation in hidden Markov model (HMM) based speech recognition. These Gaussian mixture models are easy to train using EM techniques and are able to approximate any distribution given a sufficient number of components. However, the number of parameters that can be effectively trained is restricted by the quantity of training data. This limits the ability of these systems to model highly complex distributions. Several alternatives have been devised to overcome this limitation. In particular, schemes that are based on distributed representations, such as factorial HMMs [1, 2], are popular. One such approach is multiple stream modeling [3]. Here, the feature vector is assumed to consist of independently modeled *streams*. Observations from these streams are concatenated together to form the feature vector. Performance for this form of model is limited by the independent stream assumption and to date multiple stream systems have had very limited success when applied to large vocabulary speech recognition tasks.

An alternative distributed representation is the products of experts (PoE) [4] framework. Here a set of experts are used to model the feature vector. The output from all the experts are producted together to form the system output. This output from PoE systems can be thought of as an *intersection* of all the individual experts. In contrast, the output from the standard mixture of experts (MoE) system, of which the MoG is one example, is the *union* of all the individual experts. For a MoE system, $\mathcal{M}$, composed of S experts the output likelihood may be expressed as

$$p(\mathbf{o}_t|\mathcal{M}) = \sum_{s=1}^{S} c^{(s)} p(\mathbf{o}_t|\mathcal{M}^{(s)}) \tag{1}$$

where $c^{(s)}$ is the prior for expert $\mathcal{M}^{(s)}$. For this to be a valid PDF $\sum_{s=1}^{S} c^{(s)} = 1$. The equivalent output likelihood for a PoE system may be expressed as

$$p(\mathbf{o}_t|\mathcal{M}) = \frac{1}{Z}\prod_{s=1}^{S} p(\mathbf{o}_t|\mathcal{M}^{(s)}) \tag{2}$$

$$Z = \int_{\mathcal{R}^d} \prod_{s=1}^{S} p(\mathbf{o}|\mathcal{M}^{(s)}) d\mathbf{o}. \tag{3}$$

where the integral is over the d-dimensional feature-space. Z is the normalisation term required to yield a valid PDF. PoEs have previously been investigated for time varying data, classifying character strings, using discrete HMMs [5].

The training of MoE systems is normally relatively simple and extensive use is made of the EM algorithm. However for the PoE system the training is more complex, mainly as a result of the normalisation term, leading to the use of various approximate training schemes [4]. This paper investigates systems where the individual experts are Gaussian or MoG. Products of these experts are used to model the states of a HMM. Using this form of expert, the training is dramatically simplified from the general PoE case. In addition, this form of model is, under certain restrictions, related to multiple stream systems.

2. PRODUCT OF GAUSSIANS SYSTEM

This section details the product of Gaussians (PoG) model. Two forms of representation are discussed. The first uses MoG as the experts. The second form, and the one evaluated in this paper, considers normalised versions of the product of individual components from the MoG experts.

Consider a MoG for each stream expert. In this case, equation 2 may be expressed as

$$p(\mathbf{o}_t|\mathcal{M}) = \frac{1}{Z}\prod_{s=1}^{S}\left(\sum_{m=1}^{M^{(s)}} c_m^{(s)} \mathcal{N}\left(\mathbf{o}_t; \boldsymbol{\mu}_m^{(s)}, \boldsymbol{\Sigma}_m^{(s)}\right)\right) \tag{4}$$

where $M^{(s)}$, $c_m^{(s)}$, $\boldsymbol{\mu}_m^{(s)}$, and $\boldsymbol{\Sigma}_m^{(s)}$ denote the number of components in stream s, the prior, mean and covariance matrix of component m of stream s. By expanding the product of sums into a sum of products, the *producted space*, this can be rewritten as

$$p(\mathbf{o}_t|\mathcal{M}) = \frac{1}{Z}\sum_{m_1=1}^{M^{(1)}} \cdots \sum_{m_S=1}^{M^{(S)}} \prod_{s=1}^{S} c_{m_s}^{(s)} \mathcal{N}\left(\mathbf{o}_t; \boldsymbol{\mu}_{m_s}^{(s)}, \boldsymbol{\Sigma}_{m_s}^{(s)}\right) \tag{5}$$

Sarah Airey thanks the Marshall Aid Commemoration Commission for funding. The authors would like to thank IBM for supplying equipment under the SUR program.

0-7803-7663-3/03/$17.00 ©2003 IEEE

As a product of Gaussian distributions itself has the form of a Gaussian distribution, this may be rewritten in terms of Gaussians, *meta-components*, in the producted space.

$$p(\mathbf{o}_t|\mathcal{M}) = \frac{1}{Z}\sum_{m_1=1}^{M^{(1)}}\dots\sum_{m_S=1}^{M^{(S)}} c_{\mathbf{m}}K_{\mathbf{m}}\mathcal{N}(\mathbf{o}_t;\boldsymbol{\mu}_{\mathbf{m}},\boldsymbol{\Sigma}_{\mathbf{m}}). \qquad (6)$$

where $\mathbf{m} = \left[\begin{array}{ccc} m_1 & \dots & m_S \end{array}\right]'$, m_s specifies the component from stream s. $K_{\mathbf{m}}$ is an observation-independent normalisation and can be expressed as

$$\begin{aligned} K_{\mathbf{m}} \quad = \quad & \frac{(2\pi)^{\frac{d}{2}}|\boldsymbol{\Sigma}_{\mathbf{m}}|^{\frac{1}{2}}}{\prod_{s=1}^{S}(2\pi)^{\frac{d}{2}}|\boldsymbol{\Sigma}^{(s)}_{m_s}|^{\frac{1}{2}}} \qquad (7)\\ & \exp\left[\frac{1}{2}\left(\boldsymbol{\mu}'_{\mathbf{m}}\boldsymbol{\Sigma}_{\mathbf{m}}\boldsymbol{\mu}_{\mathbf{m}} - \sum_{s=1}^{S}(\boldsymbol{\mu}^{(s)\prime}_{m_s}\boldsymbol{\Sigma}^{(s)-1}_{m_s}\boldsymbol{\mu}^{(s)}_{m_s})\right)\right]. \end{aligned}$$

The mean, covariance matrix and prior of each meta-component $\mathbf{m}$ may be expressed as

$$\begin{aligned} \boldsymbol{\mu}_{\mathbf{m}} &= \boldsymbol{\Sigma}_{\mathbf{m}}\left(\sum_{s=1}^{S}\boldsymbol{\Sigma}^{(s)-1}_{m_s}\boldsymbol{\mu}^{(s)}_{m_s}\right) && (8)\\ \boldsymbol{\Sigma}_{\mathbf{m}} &= \left(\sum_{s=1}^{S}\boldsymbol{\Sigma}^{(s)-1}_{m_s}\right)^{-1} && (9)\\ c_{\mathbf{m}} &= \prod_{s=1}^{S}c^{(s)}_{m_s}. && (10) \end{aligned}$$

In this form, the effective number of components, M, in the PoE model is the number of possible combinations of components from each MoG ($M = \prod_{s=1}^{S}M^{(s)}$). The normalisation term can be written as $Z = \sum_{m_1=1}^{M^{(1)}}\dots\sum_{m_S=1}^{M^{(S)}}c_{\mathbf{m}}K_{\mathbf{m}}$.

An alternative form of model uses a normalisation term for each meta-component $\mathbf{m}$ rather than at the product of MoG level. This is the product of Gaussians (PoG) system examined in this paper. As PoG will be used to model the state distributions for an HMM system, the likelihoods will now be conditioned on the state of the model (the dependence on the model parameters will be implicit). The likelihood for state q_t may be written as

$$\begin{aligned} p(\mathbf{o}_t|q_t) &= \sum_{m_1=1}^{M^{(1)}}\dots\sum_{m_S=1}^{M^{(S)}}\frac{c_{\mathbf{m}}}{K_{\mathbf{m}}}\prod_{s=1}^{S}\mathcal{N}\left(\mathbf{o}_t;\boldsymbol{\mu}^{(s)}_{m_s},\boldsymbol{\Sigma}^{(s)}_{m_s}\right)\\ &= \sum_{m_1=1}^{M^{(1)}}\dots\sum_{m_S=1}^{M^{(S)}}c_{\mathbf{m}}\mathcal{N}\left(\mathbf{o}_t;\boldsymbol{\mu}_{\mathbf{m}},\boldsymbol{\Sigma}_{\mathbf{m}}\right) \qquad (11) \end{aligned}$$

where $K_{\mathbf{m}}$ is the normalisation term for the meta-component given in equation 6. By using this form of normalisation it possible to closely relate this form of model to multiple stream systems.

The effects of using a PoG distributed representation may be split into two distinct parts. The first is that the position in acoustic space of the meta-component $\mathbf{m}$ is assumed to be well modeled using the means and variances derived in equations 8 and 9. Second, the prior for component $\mathbf{m}$ is close to the product of priors given in equation 10. These two aspects of the distributed representation may be considered, and evaluated, separately.

The maximum likelihood (ML) PoG system training is more complicated than that for a standard HMM or multiple stream system. To estimate the producted means and variances, a generalised EM formulation is used. The complete data set for the auxiliary function is based on the observations and the posterior probability at time t of a meta-component $\mathbf{m}$, given the current model parameters and the all the observations, $\gamma^{(\mathbf{m})}_t$. A generalised EM framework is used as there are no closed form solutions to estimate the means and variances, so gradient descent schemes are used. For further details of this training see [6]. Using the expressions in [6] requires statistics to be accumulated for each meta-component. It is then possible to guarantee that the auxiliary function increases at each step. However, this makes training systems with large numbers of streams, or components per stream, impractical. Alternatively, it is possible to use more general gradient descent learning schemes where it is possible to store updates with the individual stream components. This will be investigated in future work. An important issue in training PoE systems is initialisation. This will be discussed in section 3.

In contrast to the means and variances, the ML-estimate of the priors have simple closed form solutions. When the meta-component prior is determined using equation 10, the prior for component m of stream s may be estimated as

$$c^{(s)}_m = \frac{\sum_{t=1}^{T}\sum_{\{\mathbf{m}:\, m_s=m\}}\gamma^{(\mathbf{m})}_t}{\sum_{t=1}^{T}\sum_{\mathbf{m}}\gamma^{(\mathbf{m})}_t} \qquad (12)$$

where the summation in the denominator is over all meta-components of the state. Rather than assuming that the form of equation 10 gives a good estimate of the prior for the meta-component, the meta-component prior can be estimated directly. This will be referred to as the ML estimate of the meta-component prior (in contrast to the distributed estimate). This prevents the distributed representation incorrectly assigning high priors to regions of the feature space unobserved in the training data. The ML estimate of $c_{\mathbf{m}}$ uses the standard prior estimate, but now based on the meta-components. This ML-estimate for the meta-component priors can be applied to any form of distributed representation. The statistics required to be stored for this estimate are the accumulated posterior counts, one float per meta-component. This is feasible even for relatively complex systems.

3. MULTIPLE STREAM SYSTEMS

This section describes multiple stream systems and relates them to the PoG system described in the previous section. The form of multiple stream system considered here is the synchronous independent stream model implemented in HTK[3]. This form of multiple stream model makes the assumption that, given the state, the observations from each of the streams are independent of one another. This may be expressed as

$$p(\mathbf{o}_t|q_t) = \prod_{s=1}^{S}\left(\sum_{m=1}^{M^{(s)}}c^{(s)}_m\mathcal{N}\left(\mathbf{o}^{(s)}_t;\boldsymbol{\mu}^{(s)}_m,\boldsymbol{\Sigma}^{(s)}_m\right)\right) \qquad (13)$$

where $\mathbf{o}_t = \left[\begin{array}{ccc}\mathbf{o}^{(1)}_t & \dots & \mathbf{o}^{(S)}_t\end{array}\right]'$. In a similar fashion to the PoG system, this may be expressed in the producted space given in equation 11. The meta-component means and variances are now

$$\boldsymbol{\mu}_{\mathbf{m}} = \left[\begin{array}{c}\boldsymbol{\mu}^{(1)}_{m_1}\\ \vdots\\ \boldsymbol{\mu}^{(S)}_{m_S}\end{array}\right] \quad \boldsymbol{\Sigma}_{\mathbf{m}} = \left[\begin{array}{ccc}\boldsymbol{\Sigma}^{(1)}_{m_1} & \dots & \mathbf{0}\\ \vdots & \ddots & \vdots\\ \mathbf{0} & \dots & \boldsymbol{\Sigma}^{(S)}_{m_S}\end{array}\right] \qquad (14)$$

The total number of effective full-dimensional Gaussians is then the number of possible combinations of stream elements and is the same as the PoG system. The prior for an effective component is given by the product of the individual stream component priors, shown in equation 10.

The relationship between the PoG system and multiple stream systems is best illustrated by an example. Consider a two stream PoG system where the covariance matrices of component 1 of the two streams are given by

$$\boldsymbol{\Sigma}_1^{(1)} = \begin{bmatrix} \boldsymbol{\Sigma}^{(1)} & \mathbf{0} \\ \mathbf{0} & \sigma^2\mathbf{I} \end{bmatrix}, \quad \boldsymbol{\Sigma}_1^{(2)} = \begin{bmatrix} \sigma^2\mathbf{I} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma}^{(2)} \end{bmatrix} \qquad (15)$$

For the situation where $\sigma^2 = \infty$, using equation 9 to compute the meta-component variance yields

$$\boldsymbol{\Sigma}_{[11]'} = \begin{bmatrix} \boldsymbol{\Sigma}^{(1)} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma}^{(2)} \end{bmatrix} \qquad (16)$$

Thus when the "cross-stream" variances for a PoG system are very large, a PoG system becomes the same as a multiple stream system. Similarly, the mean of PoG will have same form as the multiple stream mean.

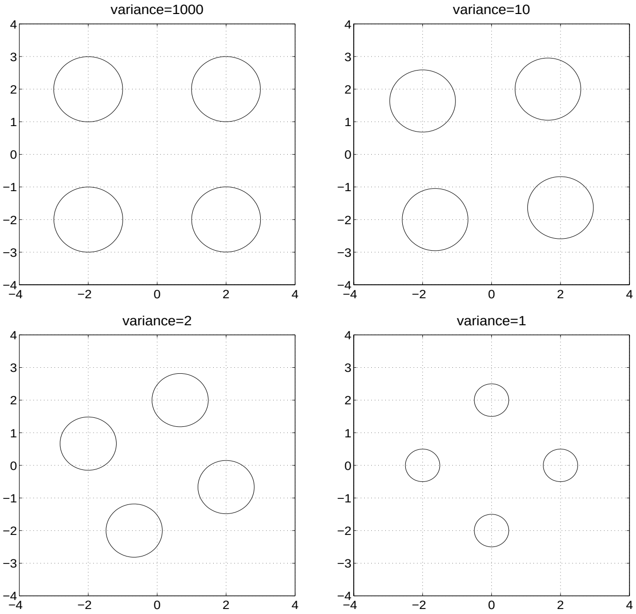

Fig. 1. The circles show one standard deviation contours of each Product of Experts. The products shift as off-stream variance is reduced.

Figure 1 shows the effect of varying the value of σ^2, the cross-stream variance, on the meta-component positions. The figure shows two MoG experts, for stream 1 the means are at [2 2]$'$ and [-2 -2]$'$ and for stream 2 at [2 -2]$'$ and [-2 2]$'$. The "within-stream" variances are 1. The top left plot has $\sigma^2 = 1000$ and the resultant meta-components are the same as a multiple stream system. As σ^2 decreases the meta-components are no longer aligned with the axis. For $\sigma^2 = 1$ the components are rotated by almost 45 degrees compared to the $\sigma^2 = 1000$. The PoG system can be seen to be more powerful than the multiple stream system, as there is no assumption about the meta-components aligning with the "axis" of the two streams. However, there is an increase in the number of model parameters, since each expert in a PoG system models the complete feature vector.

One issue in training a PoG system is how to appropriately initialise the system. Various approaches are possible [4]. For this work the relationship between the PoG system and the multiple stream system is used. A multiple stream system is built by partitioning the feature vector. This multiple stream system is then converted into a PoG system by "padding" the covariance matrix with high cross-stream values[1]. This is similar to the subspace initialisation in [4].

4. RESULTS

The performance of the PoG and multiple stream systems were evaluated on a standard large-vocabulary speaker-independent speech recognition task. Hub5, or SwitchBoard. This is a telephone bandwidth spontaneous speech recognition task. The acoustic training data is obtained from two corpora: SwitchBoard-1 (Swb1) and Call Home English (CHE). The full training corpus consists of an 265 hour training set, 4482 sides from Swb1 and 235 sides from CHE. For the experiments performed in this section a subset of this was used. A total of 68 hours was chosen to include all the speakers from Swb1 in h5train00 as well as a subset of the available CHE sides. 862 Swb1 sides and 92 CHE sides were used in this subset. This is the `h5train00sub` training set described in [7]. The speech waveforms were coded using perceptual linear prediction cepstral coefficients derived from a Mel-scale filterbank (MF-PLP) covering the frequency range from 125Hz to 3.8kHz. A total of 13 coefficients, including c_0, and their first and second order derivatives were used. Cepstral mean subtraction and variance normalisation were performed for each conversation side. Vocal tract length normalisation (VTLN) was applied in both training and test. A gender-independent cross-word-triphone diagonal-covariance mixture-Gaussian tied-state HMM system was built

All results are quoted on a three hour subset of the 2001 development data, referred to as `dev01sub`. This has been found to be a good predictor of system performance. For all recognition experiments single pass decodes were performed, rather using lattices, to avoid cross system effects. A trigram language model was used built using the language model training data described in [7].

System	Number of Components					
	2	4	6	8	10	12
`std`	46.1	43.7	42.0	40.7	39.3	39.1
`stm`	46.0	43.8	43.1	42.4	42.1	42.0

Table 1. `dev01sub` SwitchBoard performance using MoG (`std`) and three-stream multiple-stream (`stm`) systems.

Table 1 shows a comparison of a standard MoG HMM system with a 3-stream multiple stream system, the streams were the static, first and second derivative parameters. For small numbers of components there was little performance difference between the two systems. However, as the number of components was in-

[1] In practice a constant times the inverse of the variance floor was used.

creased the performance difference became large. The standard system significantly outperformed the multiple stream system.

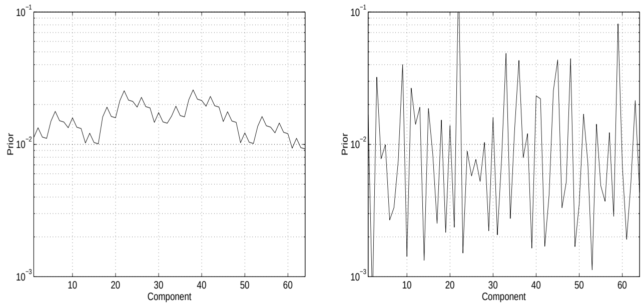

Fig. 2. Effective component priors from the multiple stream system (left) and ML meta-component priors (right) for the meta-components of the 4-component 3-stream system

Figure 2 shows, on the left-hand-side, the meta-component priors and, on the right-hand-side, the ML estimated priors for a state of the 4-component multiple stream system. Using the standard stream priors there is a clear structure. However, the ML priors have no such structure. This indicates that the priors are not well modelled using the distributed representation. The performance of the system using the ML meta-component priors was 43.5% error rate compared to 43.8% for the multiple stream system. It should be noted that there is an increase in the number of model parameters. The standard 4-component 3-stream system has 9 free parameters per state for the weights compared to 63 parameters for the ML weights. If all the model parameters were trained using ML priors the error rate dropped to 43.0%.

Number Comps.	M	Word Error Rate (%)
2	8	43.1
3	27	41.3
4	64	40.9

Table 2. `dev01sub` SwitchBoard performance using a 3-stream PoG systems, M indicates the effective number of components

In preliminary experiments similar gains were observed using ML meta-component priors for the PoG system as the multiple stream system, ML meta-component priors were therefore used for all PoG experiments. Table 2 shows the performance of 3-stream PoG systems. The 2-components per stream system gave a 43.1% error rate compared to 46.1% for the standard MoG system. Though a significant reduction in error rate was obtained the number of model parameters in the PoG system is approximately three times that of the MoG system. For the 4-components per stream system the error rate of the PoG system was 40.9%. Again this is significantly better than the MoG system performance, 43.7%. It is also better than the multiple stream system using ML meta-component priors, 43.0%. This illustrates that significant use is being made of the cross-stream variances. This is not surprising since the performance of the multiple stream systems indicate that the independent stream assumption is poor for speech recognition with MF-PLP parameters.

If the total number of model parameters is considered, rather than the components per stream, the 4-component PoG system is equivalent to the 12-component MoG system, which had an error rate of 39.1%. The PoG system performance was significantly worse than the MoG performance for approximately the same number of model parameters. However, comparing the average training data log-likelihoods of the two systems, the PoG system is slightly higher, -66.8, compared to the MoG system, -67.1. The 4-component PoG system better models the training data, though it is not a better model for classification.

5. CONCLUSIONS

This paper has described a new form of distributed representation, the PoG model, based on the PoE framework. Techniques for training and initialising this new model are presented. In addition, the relationship between this model and a multiple stream model is described. This new model is compared to standard MoG and multiple stream state-representations for HMM-based speech recognition. A standard speech recognition task, SwitchBoard, was used for the evaluation. As expected, the performance of the multiple stream system became worse than that of the MoG system as the number of components increased. Part of this degradation in performance was shown to be due to the poor representation of the priors in the system. Additional experiments on larger systems are required to further evaluate this effect. The performance of the PoG system was better than that of the equivalent number of components per stream MoG or multiple stream system. However, the PoG system has more parameters. The largest PoG system trained, 4-components per stream, had a performance significantly worse than that of the best, 12-component, MoG system. Future work will investigate building larger PoG systems to see whether as the number components increase the performance exceeds the standard MoG system. There are a number of issues that still need to be resolved for the PoG system. As well as efficient training and initialisation schemes, there is the need for efficient decoding schemes and techniques for selecting the number of streams. Future work will also examine the product of MoG and product of MoG HMMs systems for speech recognition.

6. REFERENCES

[1] Z Ghahramani and M I Jordan, "Factorial hidden Markov models," *Machine Learning*, vol. 29, pp. 245–275, 1997.

[2] M J F Gales, "Transformation streams and the HMM error model," *Computer Speech and Language*, vol. 16, pp. 225–243, 2002.

[3] S J Young, J Jansen, J Odell, D Ollason, and P Woodland, *The HTK Book (for HTK Version 2.0)*, Cambridge University, 1996.

[4] G Hinton, "Products of experts," in *Proceeding of ICANN*, 1999.

[5] A D Brown and Hinton G E, "Products of hidden markov models," Tech. Rep. GCNU TR 2000-08, Gatsby Computational Neuroscience Unit, 2000.

[6] S.S. Airey, "Products of Gaussians," M.S. thesis, University of Cambridge, 2002.

[7] T Hain, P C Woodland, G Evermann, and D Povey, "The CU-HTK March 2000 HUB5E transcription system," in *Proceedings of the Speech Transcription Workshop*, 2000.