Product of Gaussians as a Distributed Representation for Speech Recognition

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

Distributed representations allow the effective number of Gaussian components in a mixture model, or state of an HMM, to be increased without dramatically increasing the number of model parameters. Various forms of distributed representation have previously been investigated. In this work it shown that the product of experts (PoE) framework may be viewed as a distributed representation when the individual experts are mixtures of Gaussians. However, in contrast to the standard PoE model, the individual experts are not required to be valid distributions, thus allowing additional flexibility in the component priors and variances. The performance of PoE models when used as a distributed representation on a large vocabulary speech recognition task, SwitchBoard, is evaluated.

1. Introduction

A mixture of Gaussians (MoG) is commonly used for the state representation in continuous density hidden Markov model (HMM) based speech recognition systems. These mixture models can approximate any continuous distribution given sufficient components and are easily trained using expectation maximisation (EM). However, their ability to model highly complex distributions is limited since the number of parameters that can be effectively trained is restricted by the quantity of training data. This has led to the use of distributed representations [1], which allow the effective number of Gaussian components to be increased without a dramatic increase in the number of model parameters. Distributed representations may also be applied to increase the effective number of states in an HMM system, as in factorial HMMs [2].

An alternative framework to the mixture of components, or experts, is the product of experts (PoE) [3] framework. The likelihood for a PoE model is formed by multiplying the likelihood of multiple experts. The product is then normalised to yield a valid probability density function (PDF). Whereas a standard mixture represents the *union* of experts, the product represents the *intersection*. For a mixture of experts (MoE) system, $\mathcal{M}$, composed of S experts the output likelihood is 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 $\mathcal{M}^{(s)}$ and $\sum_{s=1}^{S} c^{(s)} = 1$ to ensure a valid PDF. 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. The normalisation term Z is required to yield a valid PDF. While training MoEs using EM is normally relatively simple, training PoEs is significantly more complicated, largely because of the normalisation term. This complexity has motivated various approximate training schemes for PoEs [3]. PoEs that employ such approximations have previously been investigated for time varying data, classifying character strings, using discrete HMMs [4]. However, PoE training can be dramatically simplified by using Gaussian or mixture of Gaussian experts [5], the form investigated in this paper.

This work investigates the use of product of Gaussians (PoG) to model the states of an HMM for speech recognition. Previous work [5] focused on the connection between the PoG HMM systems and multiple stream systems. It was shown that multiple streams are equivalent to a special case of the PoG and may be used to initialise the PoG training. This paper looks in more detail at PoG as a distributed representation. Since the product of Gaussians, when appropriately normalised, is itself a Gaussian distribution, a PoE that uses mixture of Gaussian experts may also be viewed as a distributed representation. However, in contrast to the standard PoE framework which requires that each of the underlying experts be a valid PDF, in a distributed representation only the final, effective, distribution must be a valid PDF. In particular, this removes some of the constraints on the individual expert priors and variances.

The next section describes the PoE framework applied to MoGs and the two possible normalisation schemes. Section 3 then describes the additional flexibility that is possible when PoEs are viewed as a distributed representation. Finally, the results on the SwitchBoard database are given.

2. Product of Gaussians System

This section details the product of Gaussians 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.

The product of S multivariate Gaussian distributions is itself of the same form as a Gaussian, though a normalisation term is required to ensure that the integral of the PDF is one.

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

The product of S multivariate Gaussians may be expressed as

$$\prod_{s=1}^{S} \mathcal{N}\left(\mathbf{o}_t; \boldsymbol{\mu}_{m_s}^{(s)}, \boldsymbol{\Sigma}_{m_s}^{(s)}\right) = K_{\mathbf{m}} \mathcal{N}(\mathbf{o}_t; \boldsymbol{\mu}_{\mathbf{m}}, \boldsymbol{\Sigma}_{\mathbf{m}}) \tag{4}$$

where $\boldsymbol{\mu}_{m_s}^{(s)}$, and $\boldsymbol{\Sigma}_{m_s}^{(s)}$ denote the mean and covariance matrix of component m of stream s, $\mathbf{m} = \begin{bmatrix} m_1 & \dots & m_S \end{bmatrix}'$ specifies the *meta-component* and m_s specifies the component from stream s. The mean, covariance matrix and prior of each meta-component $\mathbf{m}$ may be expressed as

$$\boldsymbol{\mu}_{\mathbf{m}} = \boldsymbol{\Sigma}_{\mathbf{m}} \left(\sum_{s=1}^{S} \boldsymbol{\Sigma}_{m_s}^{(s)-1} \boldsymbol{\mu}_{m_s}^{(s)} \right) \tag{5}$$

$$\boldsymbol{\Sigma}_{\mathbf{m}} = \left(\sum_{s=1}^{S} \boldsymbol{\Sigma}_{m_s}^{(s)-1} \right)^{-1} \tag{6}$$

and the normalisation term, $K_{\mathbf{m}}$ is given by

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

In this work, the PoG is used to model the state distributions for an HMM system, so the likelihoods are conditioned on the state of the model (the dependence on the model parameters will be implicit). There are two forms of normalisation that may be considered [5].

• **Product of MoGs** Suppose a MoG is used for each stream expert. In this case, equation 2, representing a state distribution, may be expressed as

$$p(\mathbf{o}_t|q_t) = \frac{1}{Z} \prod_{s=1}^{S} \left(\sum_{m=1}^{M^{(s)}} c_m^{(s)} \mathcal{N}\left(\mathbf{o}_t; \boldsymbol{\mu}_m^{(s)}, \boldsymbol{\Sigma}_m^{(s)}\right) \right) \tag{8}$$

where $M^{(s)}$ and $c_m^{(s)}$ denote the number of components in stream s and the prior of component m of stream s. By expanding the product of sums into a sum of products, this can be rewritten in terms of the meta-components. The likelihood is

$$p(\mathbf{o}_t|q_t) = \frac{1}{Z} \sum_{\mathbf{m}} c_{\mathbf{m}} K_{\mathbf{m}} \mathcal{N}(\mathbf{o}_t; \boldsymbol{\mu}_{\mathbf{m}}, \boldsymbol{\Sigma}_{\mathbf{m}}). \tag{9}$$

where the summation is over all meta-components,

$$c_{\mathbf{m}} = \prod_{s=1}^{S} c_{m_s}^{(s)} \tag{10}$$

$$Z = \sum_{\mathbf{m}} c_{\mathbf{m}} K_{\mathbf{m}} \tag{11}$$

Unlike many PoE systems, it is possible to obtain an algebraic expression for the normalisation term, Z.

• **Normalised PoG** An alternative form of model normalises at the meta-component level instead of normalising the product of MoG at the state level. The likelihood for state q_t may be written as

$$p(\mathbf{o}_t|q_t) = \sum_{\mathbf{m}} \frac{c_{\mathbf{m}}}{K_{\mathbf{m}}} \prod_{s=1}^{S} \mathcal{N}\left(\mathbf{o}_t; \boldsymbol{\mu}_{m_s}^{(s)}, \boldsymbol{\Sigma}_{m_s}^{(s)}\right) = \sum_{\mathbf{m}} c_{\mathbf{m}} \mathcal{N}\left(\mathbf{o}_t; \boldsymbol{\mu}_{\mathbf{m}}, \boldsymbol{\Sigma}_{\mathbf{m}}\right) \tag{12}$$

where $K_{\mathbf{m}}$ is the normalisation term for the meta-component given in equation 7 and $c_{\mathbf{m}}$ by equation 10. This is the form referred to in this paper as PoG and is the one implemented.

For both forms the effective number of components, M, in the product of MoG model is the number of possible combinations of components from each MoG, $M = \prod_{s=1}^{S} M^{(s)}$.

The maximum likelihood (ML) PoG system training is more complicated than that for a standard HMM or multiple stream system. This is described in more detail in [5]. 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

$$c_m^{(s)} = \frac{\sum_{t=1}^{T} \sum_{\{\mathbf{m}:\ m_s = m\}} \gamma_t^{(\mathbf{m})}}{\sum_{t=1}^{T} \sum_{\mathbf{m}} \gamma_t^{(\mathbf{m})}} \tag{13}$$

where, $\gamma_t^{(\mathbf{m})}$ is the meta-component posterior and the denominator summation is over all meta-components of the state.

3. Distributed Representation

Since the product of multivariate Gaussians, when appropriately normalised, is itself a multivariate Gaussian, the products of MoGs may be considered in the distributed representation framework instead of the PoE framework [1]. When viewed in this framework, there are additional degrees of flexibility for the individual experts. Only the final distribution need satisfy the standard constraints of a valid probability distribution:

$$p(\mathbf{o}|\mathcal{M}) \geq 0; \quad \int_{\mathcal{R}^d} p(\mathbf{o}|\mathcal{M}) d\mathbf{o} = 1 \tag{14}$$

In contrast, for the PoE framework each of the individual experts must be a valid distribution. This loosens the restrictions governing two aspects of the individual experts: the component priors and the component covariance matrices.

3.1. Component Priors

For each expert to be a valid PDF, the component priors must satisfy

$$\sum_{m=1}^{M^{(s)}} c_m^{(s)} = 1; \quad c_m^{(s)} \geq 0 \tag{15}$$

The producted meta-component priors are then calculated according to equation 10. However, it is sufficient that only the producted meta-components satisfy the constraints:

$$\sum_{\mathbf{m}} c_{\mathbf{m}} = 1; \quad c_{\mathbf{m}} \geq 0. \tag{16}$$

In this case, the constraints on the individual expert component priors are relaxed. Rather than assuming that the strict product of component priors yields a good estimate of the meta-component prior, $c_{\mathbf{m}}$ can be calculated directly from the data. The ML estimate of $c_{\mathbf{m}}$ uses the standard closed form prior estimate, but now based on the meta-components:

$$c_{\mathbf{m}} = \frac{\sum_{t=1}^{T} \gamma_t^{(\mathbf{m})}}{\sum_{t=1}^{T} \sum_{\mathbf{m}} \gamma_t^{(\mathbf{m})}}. \tag{17}$$

Using the ML-estimate for the meta-component prior has an interesting effect of the product of MoG system. The product of MoG likelihood given in equation 9 can be re-expressed

as (writing $w_{\mathbf{m}}$ instead of $c_{\mathbf{m}}$ to denote the relaxation on the constraints)

$$p(\mathbf{o}_t|q_t) = \sum_{\mathbf{m}} \left(\frac{w_{\mathbf{m}} K_{\mathbf{m}}}{\sum_{\mathbf{m}} w_{\mathbf{m}} K_{\mathbf{m}}} \right) \mathcal{N}(\mathbf{o}_t; \boldsymbol{\mu}_{\mathbf{m}}, \boldsymbol{\Sigma}_{\mathbf{m}}) \quad (18)$$

where $w_{\mathbf{m}}$ is the weight of meta-component $\mathbf{m}$. A valid PDF requires that $w_{\mathbf{m}} K_{\mathbf{m}} \geq 0$ with at least one meta-component value greater than 0. Since $w_{\mathbf{m}}$ can take on any value, the effect of the Gaussian product normalisation term $K_{\mathbf{m}}$ can be "undone" by an appropriate value of $w_{\mathbf{m}}$. Thus the prior

$$c_{\mathbf{m}} = \frac{w_{\mathbf{m}} K_{\mathbf{m}}}{\sum_{\mathbf{m}} w_{\mathbf{m}} K_{\mathbf{m}}} \quad (19)$$

can be estimated as in 17. The resulting likelihood is the same as the PoG likelihood in equation 12. Hence when using ML meta-component priors, the two normalisation schemes yield the same form of model. Another consequence of using this form of component prior is that it allows meta-components that are in unlikely regions of acoustic space to be ignored. This becomes increasingly important as the number of components in the underlying streams increases.

When using these ML estimates for $c_{\mathbf{m}}$ there is an increase in the number of prior parameters, n. For the product form

$$n = \sum_{s=1}^{S} (M^{(s)} - 1) \quad (20)$$

and for the ML meta-component priors

$$n = \left(\prod_{s=1}^{S} M^{(s)} \right) - 1. \quad (21)$$

However, as the number of model parameters is typically dominated by the means and variances, this is not significant until there are large numbers of components in the streams.

3.2. Covariance Matrices

The second effect of using a distributed representation arises in the covariance matrices. For a valid probability Gaussian distribution the covariance matrix must be positive definite. However, in a distributed system only each meta-component covariance matrix $\Sigma_{\mathbf{m}}$ need satisfy this constraint, instead of each individual expert covariance.

The meta-component covariance matrix from equation 6 is given by (writing $\boldsymbol{\Lambda}_{m_s}^{(s)}$ instead of $\boldsymbol{\Sigma}_{m_s}^{(s)-1}$ to denote the relaxation on the constraints)

$$\boldsymbol{\Sigma}_{\mathbf{m}} = \left(\sum_{s=1}^{S} \boldsymbol{\Lambda}_{m_s}^{(s)} \right)^{-1}. \quad (22)$$

In this work only diagonal covariance matrices are used (in common with the vast majority of speech recognition systems). The variances for each dimension i, $\sigma_{m_s i}^2$, along the leading diagonal of the expert covariance matrices can be negative, provided that all of elements of the diagonal are positive in the final producted covariance matrix.

An additional flexibility, not investigated in this work, is that the rank inverse covariance matrices of the individual experts need not have full rank. Thus in contrast to the product of Gaussian system described in [6], the EMLLT framework, which may be viewed as the product of rank-1 experts, is possible.

3.3. Likelihood Calculation

For the standard PoE system the likelihood may be efficiently calculated; it is only necessary to compute the individual stream log-likelihoods and then combine them using equation 8. This is significantly less computationally expensive than computing the likelihood from the components in the producted space. When the flexibility of distributed representations is exploited, the computation is not so straightforward.

The calculation of the log-likelihood contribution of each stream may be split into two distinct parts. Since the stream distribution is not required to be a valid PDF, its output will be denoted as $f(\mathbf{o}_t, m)$ and the normalisation terms will be initially ignored.

$$f(\mathbf{o}_t, m_s) = -\frac{1}{2}(\mathbf{o}_t - \boldsymbol{\mu}_{m_s}^{(s)})' \boldsymbol{\Lambda}_{m_s}^{(s)} (\mathbf{o}_t - \boldsymbol{\mu}_{m_s}^{(s)}). \quad (23)$$

The only issue with computing $f(\mathbf{o}_t, m_s)$ is the existence of $\boldsymbol{\Lambda}_{m_s}^{(s)}$ not $\boldsymbol{\Sigma}_{m_s}^{(s)-1}$. It is then possible to write the meta-component likelihood as

$$p(\mathbf{o}_t|\mathbf{m}) = \frac{1}{(2\pi)^{\frac{d}{2}} |\boldsymbol{\Sigma}_{\mathbf{m}}|^{\frac{1}{2}}} \exp\left(\sum_{s=1}^{S} f(\mathbf{o}_t, m_s) \right). \quad (24)$$

The likelihood of a state generating the observation may be obtained by summing over all of the meta-components. For efficiency, the meta-component prior can be combined with the meta-component normalisation term. The efficiency of this style of approach for non-full-rank matrices is described in [7].

4. Results

The performance of the PoG 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 [8]. 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 Gaussian mixture tied-state HMM system was built. For multiple stream-systems, the standard HTK multiple stream system and the PoG systems, three streams were used.

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, using lattices, to avoid cross system effects. A trigram language model was used, built using the language model training data described in [8].

Figure 1 shows, on the left-hand-side, the meta-component priors and, on the right-hand-side, the ML estimated priors for

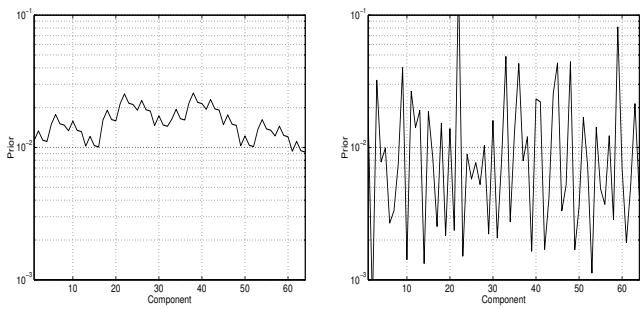

Figure 1: Effective component priors from the multiple stream system (left) and ML meta-component priors (right) for the meta-components of the 4-component 3-stream system

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, as previously observed in [5].

System	Prior	Var	Log-Lik	Error Rate
std	—	—	-69.36	46.1
stm	prd	—	-69.34	46.0
pog	ml	+ve	-68.90	44.4
	prd	−ve	-68.19	43.2
	ml	−ve	-68.17	43.1

Table 1: dev01sub SwitchBoard performance using standard (std) and multiple-stream (stm) systems and PoG (pog) systems, with 2-components per stream.

Table 1 shows the training data average log-likelihood per frame and recognition performance for various 2-component per stream systems. The priors for the systems are estimated either using the product of stream experts given in equation 10, prd, or using the ML-priors estimated using equation 17, ml. The variances are either constrained to be positive +ve, or allowed to be negative −ve. Comparing the multiple stream system with the standard MoG system, there is little difference between the two log-likelihoods. This may be partly attributed to the limitation of product form of the components and to the small number of components in the underlying streams. The word error rates of the two systems are approximately the same. The three PoG systems all have higher log-likelihoods and lower word error rates than the std and stm systems. The advantage over stm may be because of the power of the pog system to model data compared to the stm system [5]. The log-likelihoods are decreased and the error rates increased if the variances are constrained to be positive. Thus the additional flexibility of having "negative" variances is useful for pog systems. In contrast, the comparison of the ml and prd component priors shows little difference in performance.

Table 2 shows the performance of various 4-component per stream systems. As the number of components per stream increases, the advantage of using ml priors increases (though note the number of model parameters also slightly increases). This is shown in both an increase in log-likelihood and decrease in error rate. Again the use of the pog yields lower error rate. As

System	Prior	Var	Log-Lik	Error Rate
std	—	—	-68.60	43.7
stm	prd	—	-68.60	43.8
	ml	—	-68.14	43.0
pog	ml	−ve	-66.78	40.9

Table 2: dev01sub SwitchBoard performance using standard (std) and multiple stream systems (stm) and PoG systems (pog), with 4-components per stream.

previously noted [5], comparing error rates of the pog system, with standard MoG systems with the same number of components per stream is not fair since the pog system has significantly more parameters. The performance of a 12-component standard system, which has an equivalent number of parameters as the 4-component per stream pog is 39.1% on this task. This is significantly better than the best pog system, though the log-likelihood is slightly worse.

5. Conclusions

This paper has described a form of distributed representation, the PoG model, based on the PoE framework. Techniques for normalisation and training are detailed. The general characteristics of distributed representations are explored. In particular, PoG is shown to be equivalent to product of MoG. The PoG 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. The effects of the flexibility afforded by the distributed representation framework in calculating the producted system priors and individual expert covariances are also evaluated. Allowing the "variances" of individual experts to be negative is a better model in terms of likelihood and reducing the word error rate. Using ML meta-component priors is also better, but at the cost of a slight increase in the number of model parameters.

6. References

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

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

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

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

[5] S S Airey and M J F Gales, "Products of Gaussians and multiple stream systems," in *Proceedings ICASSP*, 2003.

[6] C K I Williams, F V Agakov, and S N Felderhof, "Products of Gaussians," in *Proceeding of NIPS*, 2001.

[7] P A Olsen and R A Gopinath, "Modeling inverse covariance matrices by basis expansion," in *Proceedings ICASSP*, 2002.

[8] 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.