Modeling Posterior Probabilities using the Linear Exponential Family

Peder Olsen[1]*, Vaibhava Goel*[1]*, Charles Micchelli*[2]*, and John Hershey*[1]

[1]T.J. Watson Research Center, IBM
[2]Department of Mathematics and Statistics, State University of New York, Albany
[2]Department of Mathematics, City University of Hong Kong
{pederao,vgoel,jrhershe}@us.ibm.com, charles_micchelli@hotmail.com

Abstract

A commonly used distribution on the probability simplex is the Dirichlet distribution. In this paper we present the linear exponential family as an alternative. The distribution is known in the statistics community, but we present in this paper a numerically stable method to compute its parameters. Although the Dirichlet distribution is known to be a good Bayesian prior for probabilities we believe this paper shows that the linear exponential model offers a good alternative in other contexts, such as when we want to use posterior probabilities as features for automatic speech recognition. We show how to incorporate posterior probabilities as additional features to an existing GMM, and show that the resulting model gives a 0.6% relative gain on a broadcast news speech recognition system.

Index Terms: Linear Exponential Family, Simplex, Divided Difference, Partition Function

1. Introduction

In [1] we discussed use of general exponential families for acoustic modeling. We shall use this as a starting point to treat the linear exponential family.

2. The Linear Exponential Family

The general exponential family is given by

$$\mathrm{P}(\mathbf{x}|\boldsymbol{\lambda}) = \frac{e^{\boldsymbol{\lambda}^T \boldsymbol{\phi}(\mathbf{x})}}{Z(\boldsymbol{\lambda})} \quad \text{where} \quad Z(\boldsymbol{\lambda}) = \int_D e^{\boldsymbol{\lambda}^T \boldsymbol{\phi}(\mathbf{x})}\, \mathrm{dx} \quad (1)$$

where $\boldsymbol{\phi}$ is the feature function, $\boldsymbol{\lambda}$ are the parameters in the exponential family and $Z(\boldsymbol{\lambda})$ is the partition function or normalizer. D is the domain for which x is defined.

The linear exponential family is the special case $\boldsymbol{\phi}(\mathbf{x}) = \mathbf{x}$. We consider this family when the domain for $\mathbf{x} \in \mathbb{R}^n$ is the probability simplex: $\sum_{i=1}^n x_i = 1$, $x_i \geq 0$. We write $\mathcal{P}_n$ for the probability simplex:

$$\mathcal{P}_n = \left\{ x \in \mathbb{R}^n | \sum_{i=1}^n x_i = 1, x_i \geq 0 \right\}. \quad (2)$$

Thus the linear exponential family can be seen as a distribution over probabilities. The partition function for the linear exponential family over simplex is

$$Z(\boldsymbol{\lambda}) = \sum_{i=1}^n \frac{e^{\lambda_i}}{\prod_{j \neq i}(\lambda_i - \lambda_j)}. \quad (3)$$

Charles Micchelli is partially supported by the NSF grant DMS 0712827 and AFOSR FA9550-09-0511

Note that $\lambda_i = \lambda_j$, $i \neq j$ is a removable singularity for $Z(\boldsymbol{\lambda})$. While $Z(\boldsymbol{\lambda})$ is defined everywhere, some subtlety is required to avoid numerical problems in its evaluation, and this we consider to be the main contribution of this paper.

Figure 1 for a plot of a sample linear exponential density in $\mathcal{P}_3$. As seen in the figure, an interesting property of this family is that the modes can only occur on the corners of the probability simplex, and it is monotonically decreasing or increasing along any line on the simplex. This we believe is an intuitively desirable property when using posterior probabilities as features; we would like to trust that the posteriors are increasingly more trustworthy the higher their values are.

A more popular prior distribution over probability simplex is the Dirichlet distribution. The Dirichlet family is also an exponential family with $\boldsymbol{\phi}(\mathbf{x}) = \log(\mathbf{x})$. The partition function for the Dirichlet family is the beta function:

$$Z_D(\boldsymbol{\lambda}) = \frac{\prod_i \Gamma(\lambda_i + 1)}{\Gamma(d + \sum_i \lambda_i)}. \quad (4)$$

However, for incorporating posteriors as features, we believe that the linear exponential family is a good alternative. Firstly, the linear exponential family is monotonic, whereas the Dirichlet family is not. Secondly, the behavior of the linear exponential family near the simplex boundary is quite nice, whereas the Dirichlet family will have a likelihood that is zero or infinite at the boundary. These properties makes it easier for the Dirichlet family to be overtrained. Unlike the linear exponential family the computation of the partition function for the Dirichlet family is straightforward and suffers no numerical issues. Subsections 2.1, 2.2 and 2.3 all deal with computing the partition function (3) for the linear exponential family.

The linear exponential family and the Dirichlet distribution both belong to a larger exponential family on the probability simplex, [2]. The partition function for the larger family is also known and can be written analytically.

2.1. The Divided Difference

The partition function (3) is a special case of the divided difference. The divided difference is defined recursively by

$$\begin{aligned} [\lambda_i]\, f &= f(\lambda_i) \\ [\lambda_i, \ldots, \lambda_{i+j}]\, f &= \frac{[\lambda_{i+1}, \ldots, \lambda_{i+j}]f - [\lambda_i, \ldots, \lambda_{i+j-1}]f}{\lambda_{i+j} - \lambda_i} \end{aligned}$$

It can be shown that the divided difference is explicitly given by the following formula:

$$[\lambda_1, \lambda_2, \ldots, \lambda_n]f = \sum_{i=1}^n \frac{f(\lambda_i)}{\prod_{j \neq i}(\lambda_i - \lambda_j)}. \quad (5)$$

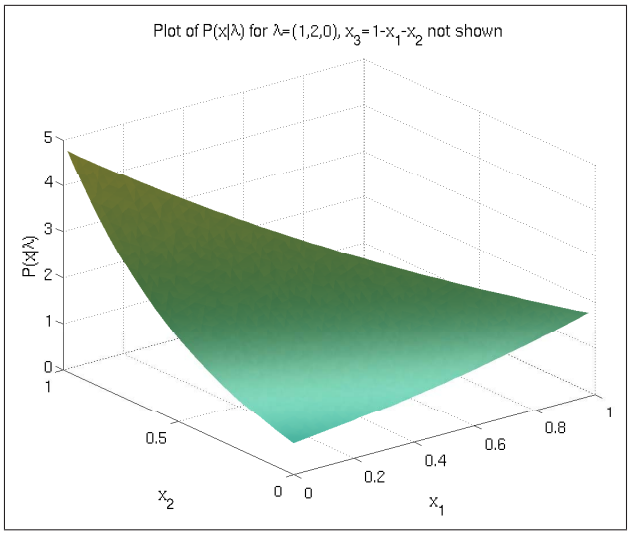

Figure 1: *Plot of* $P(\mathbf{x}|\boldsymbol{\lambda})$ *for* $\boldsymbol{\lambda} = (1, 2, 0)$ *and* $x_1 + x_2 \leq 1$. $x_3 = 1 - x_1 - x_2$ *is implicit in the plot.*

Our interest in the divided difference is mainly due to the Hermite-Genocchi formula that relates the divided difference to an integral over the probability simplex:

$$[\lambda_1, \lambda_2, \ldots, \lambda_n]f = \int_{\mathcal{P}_n} f^{(n-1)}(\boldsymbol{\lambda}^T \mathbf{x})\, \mathrm{d}\mathbf{x}, \tag{6}$$

where $f^{(n-1)}$ indicates the $n-1$th derivative.
The choice $f(x) = e^x$ in the Hermite Genocchi formula gives us the explicit formula for the partition function for the linear exponential function, namely

$$Z(\boldsymbol{\lambda}) = [\lambda_1, \lambda_2, \ldots, \lambda_n] \exp = \sum_{i=1}^{n} \frac{e^{\lambda_i}}{\prod_{j \neq i}(\lambda_i - \lambda_j)}. \tag{7}$$

The analytic expression given appears to have singularities at the values where $\lambda_i = \lambda_j$ for some $i \neq j$, but this is clearly not the case since the partition function is an integral of a bounded function over a finite volume. Therefore there are only removable singularities. We still need a way to compute the partition function in a numerically stable manner when two or more of the parameters are close or equal to each other. We shall see in the next section how we can apply Opitz's formula, [3, 4], to compute the function for any value of $\boldsymbol{\lambda}$ in a numerically stable way.
Let's review a few of the properties of the partition function that will be useful later.

1. **Invariance to permutation:**

   $$Z(\lambda_1, \lambda_2, \ldots, \lambda_n) = Z(\lambda_{\sigma(1)}, \lambda_{\sigma(2)}, \ldots, \lambda_{\sigma(n)}),$$

   for any permutation, $\sigma : \mathbb{Z}_n \to \mathbb{Z}_n$.

2. **Shift invariance:** This property ensures that the density is unchanged when a constant is added to each of the λ_i's.

   $$Z(\lambda_1 + t, \lambda_2 + t, \ldots, \lambda_n + t) = e^t Z(\lambda_1, \lambda_2, \ldots, \lambda_n).$$

3. **Derivatives and limits.** The partial derivative with respect to a variable λ_i is simply computed by adding the variable twice:

   $$\frac{\partial Z(\boldsymbol{\lambda})}{\partial \lambda_i} = Z(\lambda_i, \boldsymbol{\lambda}). \tag{8}$$

The first property is common to all divided differences, the second is easy to verify and the third can be reduced to showing $\frac{\partial Z(\boldsymbol{\lambda})}{\partial \lambda_1} = \lim_{t \to \lambda_1} Z(t, \boldsymbol{\lambda})$ by the first property. Since each side can be computed directly, we leave it to the reader to verify that they are indeed equal.

2.2. Efficient and Stable Computation of the Partition Function (3)

The direct formula for the computation of the partition function requires $\mathcal{O}(n^2)$ operations. The problem with computing the value when two or more entries of $\boldsymbol{\lambda}$ are close is a problem common to all divided differences. This problem has already been solved for us. A nice exposition of divided differences can be found in [4]. Let us describe some of it here.
The divided difference satisifies Leibniz' rule,

$$\begin{aligned}[\lambda_0, \ldots, \lambda_n](fg) \;=\; & [\lambda_0]f\ [\lambda_0, \ldots, \lambda_n]g \\ & + [\lambda_0, \lambda_1]f\ [\lambda_1, \ldots, \lambda_n]g \\ & + \cdots + [\lambda_0, \ldots, \lambda_n]f\ [\lambda_n]g.\end{aligned} \tag{9}$$

Define the divided difference matrix,

$$T_f(\boldsymbol{\lambda}) = \begin{pmatrix} [\lambda_0]f & [\lambda_0, \lambda_1]f & \ldots & [\lambda_0, \ldots, \lambda_n]f \\ 0 & [\lambda_1]f & \ldots & [\lambda_1, \ldots, \lambda_n]f \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \ldots & 0 & [\lambda_n]f \end{pmatrix}.$$

The divided difference matrix has both an additive and a mutliplicative structure, $T_{f+g}(\boldsymbol{\lambda}) = T_f(\boldsymbol{\lambda}) + T_g(\boldsymbol{\lambda})$ and $T_{f \cdot g}(\boldsymbol{\lambda}) = T_f(\boldsymbol{\lambda}) \cdot T_g(\boldsymbol{\lambda})$. The multiplicative structure is a direct consequence of Leibniz' rule for divided difference. With $f(x) = e^x$ we get the divided difference matrix for the exponential $E(\boldsymbol{\lambda}) = T_{\exp}(\boldsymbol{\lambda})$ which equals

$$E(\boldsymbol{\lambda}) = \begin{pmatrix} Z(\lambda_0) & Z(\lambda_0, \lambda_1) & \ldots & \mathbf{Z(\boldsymbol{\lambda})} \\ 0 & Z(\lambda_1) & \ldots & Z(\lambda_1, \ldots, \lambda_n) \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \ldots & 0 & Z(\lambda_n) \end{pmatrix}.$$

The element in the upper right hand corner is the value of the partition function we are after. Now, assume that $f(x) = \sum_{i=0}^{\infty} a_i x^i$ has a Taylor expansion that is valid everywhere. Denote the i'th power function by $p_i(x) = x^i$. Then we have

$$T_f(\boldsymbol{\lambda}) = \sum_{i=0}^{\infty} a_i T_{p_i}(\boldsymbol{\lambda}) = \sum_{i=0}^{\infty} a_i (T_{p_1}(\boldsymbol{\lambda}))^i = \sum_{i=0}^{\infty} a_i J^i,$$

where $J = T_{p_1}(\boldsymbol{\lambda})$. This is known as Opitz's formula. By definition we have $[\lambda_1]p_1 = \lambda_1$, and by application of the Hermite-Genocchi formula we see that $[\lambda_1, \lambda_2]p_1 = 1$ and $[\lambda_1, \lambda_2, \ldots, \lambda_i]p_1 = 0$ for $i > 2$. We conclude that

$$J = T_{p_1}(\boldsymbol{\lambda}) = \begin{pmatrix} \lambda_0 & 1 & 0 & 0 & \cdots & 0 \\ 0 & \lambda_1 & 1 & 0 & \cdots & 0 \\ 0 & 0 & \lambda_2 & 1 & & 0 \\ \vdots & \vdots & & \ddots & \ddots & \\ 0 & 0 & 0 & 0 & & \lambda_n \end{pmatrix}. \tag{10}$$

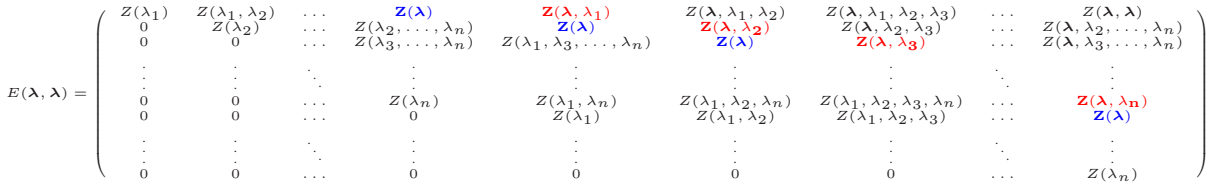

Figure 2: *The partition function, $Z(\boldsymbol{\lambda})$, and its gradient, $Z(\lambda_i, \boldsymbol{\lambda})$, are located at two diagonals in the divided difference matrix $E(\boldsymbol{\lambda}, \boldsymbol{\lambda})$ as shown above*

Since J is explicitly known, it is clear that Opitz's formula do not suffer the numerical problems that the analytic formula has. If we specialize Opitz's formula to the exponential we get

$$E(\boldsymbol{\lambda}) = e^J. \tag{11}$$

The computation of the exponential of a matrix is a mature subject, [5]. There are many techniques to compute the matrix exponential and we make no claim to have implemented the most efficient method. The matrix exponential satisfies the following property

$$X = J/2^m \quad e^J = (e^X)^{2^m} \quad \text{and} \quad e^X = \sum_{i=0}^{\infty} X^i/i! \tag{12}$$

given by its Taylor series. The matrices J and X are sparse, but e^X is not. $(e^X)^{2^m}$ can be computed by squaring e^X m consecutive times, at a total cost of $\mathcal{O}(mn^3)$ operations. The computation of e^X can be done in $\mathcal{O}(n^3)$ operations when the Frobenius norm $\|J/2^m\| \leq 1$. We have $\|J\|^2 = n + \|\boldsymbol{\lambda}\|_2^2$, so the complexity to compute the matrix exponential is $\mathcal{O}(n^3 \log_2(n + \|\boldsymbol{\lambda}\|_2^2))$. This is in contrast to the direct computation using (3), which has the lower complexity $\mathcal{O}(n^2)$.

2.2.1. *Computing the gradient*

In order to compute the maximum likelihood solution we need to maximize the log likelihood $\boldsymbol{\lambda}^T \mathbf{s} - \log Z(\boldsymbol{\lambda})$, where $\mathbf{s} = \frac{1}{T}\sum_{t=1}^{T} \mathbf{x}_t$ is the statistics. The problem can be solved by any optimization package, but then the gradient needs to be computed as well. This could be done by use of the derivative formula

$$\frac{\partial Z(\boldsymbol{\lambda})}{\partial \lambda_i} = Z(\boldsymbol{\lambda}, \lambda_i) = E(\lambda_i, \boldsymbol{\lambda})_{1,n+1}. \tag{13}$$

We can do better though and compute the partition function and its gradient simultaneously. The divided difference matrix $E(\boldsymbol{\lambda}, \boldsymbol{\lambda})$ contains all the information. See Figure 2 for the explicit formula. Since the entries of the gradient are on the $n'th$ diagonal the entries will be populated after n terms of the Taylor series have been calculated. The computation is less than 8 times that of computing $Z(\boldsymbol{\lambda})$, and in practice we saw that our code to compute $Z(\boldsymbol{\lambda})$ and $\nabla Z(\boldsymbol{\lambda})$ was roughly only 4 times slower. The overall cost of computing $Z(\boldsymbol{\lambda})$ and $\nabla Z(\boldsymbol{\lambda})$ is order $\mathcal{O}(n^3 \log_2(n + \|\boldsymbol{\lambda}\|_2^2))$.

2.3. Comparing Opitz's formula to direct computation

In this section we illustrate the improvement in numerical stability when using Opitz's formula (11) over the direct formula (3) to compute the divided difference. For illustrational purposes we shall use values of the partition function for which we can compute the partition function analytically. The vector $\boldsymbol{\lambda}_\epsilon = \epsilon(0, 1, \ldots, n)$ is such a value. We find a simple expression for $Z(\boldsymbol{\lambda}_\epsilon)$ by direct computation:

$$\begin{aligned} Z(\boldsymbol{\lambda}_\epsilon) &= [0, \epsilon, 2\epsilon, \ldots, n\epsilon] \exp \\ &= \sum_{i=0}^{n} \frac{e^{i\epsilon}}{\epsilon^n \prod_{j=0}^{i-1}(i-j) \prod_{j=i+1}^{n}(i-i)} \\ &= \sum_{i=0}^{n} \frac{(-1)^{n-i} e^{i\epsilon}}{\epsilon^n i!(n-i)!} \\ &= \frac{(-1)^n}{\epsilon^n n!} \sum_{i=0}^{n} (-e^{\epsilon})^i \binom{n}{k} \\ &= \frac{1}{n!} \left(\frac{e^{\epsilon} - 1}{\epsilon} \right)^n. \end{aligned} \tag{14}$$

For comparison purposes we considered the computation of $\log Z(\boldsymbol{\lambda}_\epsilon)$ by the formula to be the ground truth. Then we compared the (3), and Opitz's formula (11) to the ground truth. We can see in Table 1 that the direct computation breaks down both when ϵ becomes small and when n becomes large. Opitz's formula, on the other hand, is quite stable even for large values of n. The reason for the large error in the direct formula for the divided difference is due to positive and negative terms cancelling each other out to such a degree that it significantly affects the precision of the result. This happens because the divided difference is zero for the first $n-1$ powers of x. This means that for $n = 10$, $\epsilon = 0.1$ the function e^x will have the first 10 terms of its Taylor series cancelled and the first contributing term will be $\epsilon^{10}/10!$, which is much smaller than $e^{0.1}$. Figure 3 shows the individual terms of the divided difference for $n = 10$ $\epsilon = 0.1$. The largest term is 12 orders of magnitude larger than the divided difference.

3. Incorporating Posteriors into Gaussian Distribution

In the experiments conducted in this paper, the vector of posterior probabilities $\mathbf{p}$ was incorporated as additional features to the baseline Gaussian distribution features $[\mathbf{x}\ \mathbf{x}^2]$. We tested the following two exponential families

$$\text{Normal + Linear} \quad : \quad \phi(\mathbf{x}, \mathbf{p}) = \begin{bmatrix} \mathbf{x} & \mathbf{x}^2 & \mathbf{p} \end{bmatrix} \tag{15}$$

$$\text{Normal +Dirichlet} \quad : \quad \phi(\mathbf{x}, \mathbf{p}) = \begin{bmatrix} \mathbf{x} & \mathbf{x}^2 & \log(\mathbf{p}) \end{bmatrix} \tag{16}$$

In both these cases, the parameters corresponding to the Gaussian features are kept fixed. The parameters for posterior features are initialized to 0 and estimated under the bMMI objective. Using standard extended Baum Welch techniques it can be shown that the BMMI objective can be optimized via an auxilliary function that coincides with the ML objective function, [6].

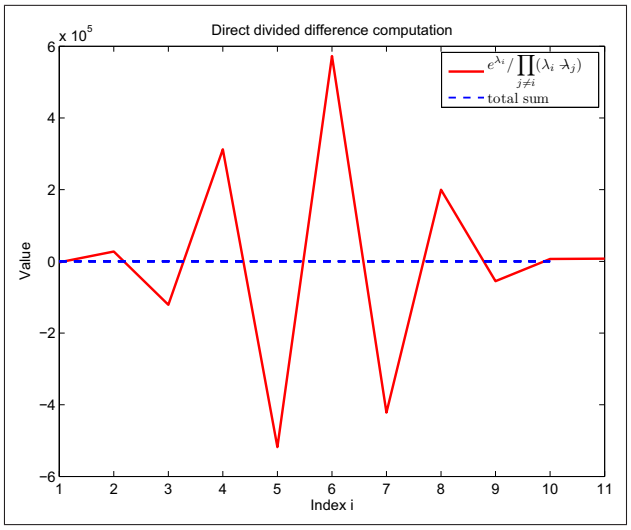

Figure 3: *Plot showing* $e^{\lambda_i}/\prod_{j\neq i}(\lambda_i - \lambda_j)$ *for* $\boldsymbol{\lambda} = (0, 0.1, 0.2, \ldots, 1)$. *The largest term is of the order* $6 * 10^5$, *while the divided difference* $Z(\boldsymbol{\lambda}) = 4.56 * 10^{-7}$ *is 12 orders of magnitude smaller.*

4. Experiments

We evaluated the linear exponential family on a Broadcast News LVCSR task. The acoustic model training set comprises 50 hours of data from the 1996 and 1997 English Broadcast News Speech corpora (LDC97S44 and LDC98S71), and was created by selecting entire shows at random. The EARS Dev-04f set (dev04f), a collection of 3 hours of audio from 6 shows from November 2003, is used for testing the models.

The acoustic features are obtained by first computing 13-dimensional perceptual linear prediction (PLP) features with speaker-based mean, variance, and vocal tract length normalization. Nine such features were concatenated and projected to a 40 dimensional space using LDA. The 40 dimensional features were further normalized using one feature space linear regression transform (fMLLR) per speaker. An fMMI transform [7] was estimated to arrive at the final feature space in which acoustic models were trained. The acoustic models consist of 44 phonemes with each phoneme modeled as three-state, left-to-right HMMs with no skip states. Mixtures of exponential distributions are used to model each state, with the overall model having 50K components.

The baseline (fMMI only) acoustic models were built using first Maximum Likelihood (ML) training and then the boosted MMI [7] estimation process. These models had a word error rate of 19.4% on the dev04f test set.

The posterior probabilities were obtained using Sparse Representation Phone Identification Features (SPIF) [8]. These posteriors were at phoneme level, and 44 phoneme posteriors where considered as additional features for each of the individual Gaussians. Each Gaussian can then decide how much weight to give each of the classifiers. The resulting combined exponential models (15) and (16) then had exponential features of dimension 124=80(Gaussian)+44(Posterior).

Table 2 shows recognition word error rates for the exponential models combining fMMI and SPIF posterior features. It also shows WER using Dirichlet distribution. As seen from the table, we get a 0.6% absolute gain from using the linear exponential family. The Dirichlet family does not give an error rate reduction. We are investigating whether this is due to overtraining and whether the results generalizes to other test sets.

n	ϵ	$\log(Z(\boldsymbol{\lambda}_\epsilon))$	error 1	error 2
1	1.0000	0.54132	0	2.2204e-16
1	0.1000	0.05042	2.9976e-15	6.245e-17
1	0.0100	0.00500	1.9163e-14	8.5869e-17
1	0.0010	0.00050	6.907e-13	1.8681e-16
10	1.0000	-9.69116	3.742e-09	3.5527e-15
10	0.1000	-14.6002	0.0095754	3.5527e-15
10	0.0100	-15.0543	nan	3.5527e-15
100	1.0000	-309.606	inf	1.1369e-13
100	0.0001	-363.734	nan	1.1369e-13
1000	1.0000	-5370.80	inf	3.638e-12
1000	0.0001	-5912.08	9874.4	8.1855e-12

Table 1: *The error in computing* $\log(Z(\boldsymbol{\lambda}_\epsilon))$ *when using (3) for the divided difference is referred to as "error 1", and the error obtained when using Opitz's formula (11) is "error 2". Note that (3) breaks down rapidly when* $n > 10$. *The values "nan" (not a number) and "inf" (infinity) indicates numerical overflow, or that the algorithm evaluated* log *of zero or a negative number.*

	WER
fMMI only	19.4%
fMMI + linear exp family	18.8%
fMMI + Dirichlet	20.2%

Table 2: *Word error rate results*

5. Acknowledgements

The authors would like to thank Tara Sainath and Bhuvana Ramabhadran for providing us with SPIF features.

6. References

[1] V. Goel and P. Olsen, "Acoustic modeling using exponential families," in *Proceedings of Interspeech 2009*, Brighton, UK, September 2009, pp. 1423–1426.

[2] O. E. Barndorff-Nielsen and B. Jörgensen, "Some parametric models on the simplex," *Journal of Multivariate Analysis*, vol. 39, no. 1, pp. 106–116, October 1991.

[3] G. Opitz, "Steigungsmatrizen," *Z. Angew. Math. Mech*, vol. 44, pp. T52–T54, 1964.

[4] C. de Boor, "Divided differences," *Surv. Approx. Theory*, vol. 1, pp. 46–49, 2005.

[5] C. Moler and C. V. Loan, "Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later," *SIAM Review*, vol. 46, no. 1, pp. 3–49, 2003.

[6] S. Axelrod, V. Goel, R. A. Gopinath, P. A. Olsen, and K. Visweswariah, "Discriminative estimation of subspace constrained gaussian mixture models for speech recognition," *Transactions in Speech and Audio Processing*, vol. 15, no. 1, pp. 172–189, January 2007.

[7] D. Povey, D. Kanevsky, B. Kingsbury, B. Ramabhadran, G. Saon, and K. Visweswariah, "Boosted MMI for model and feature-space discriminative training ," in *Proc. ICASSP*, Las Vegas, Nevada, March 2008, pp. 4057–4060.

[8] T. N. Sainath, D. Nahamoo, R. Ramabhadran, and D. Kanevsky, "Sparse representation phone identification features for speech recognition," Speech and Language Algorithms Group, IBM, Tech. Rep., 2010.