INTERSPEECH 2009 BRIGHTON

Compacting Discriminative Feature Space Transforms for Embedded Devices

Etienne Marcheret[1], *Jia-Yu Chen*[2], *Petr Fousek*[3], *Peder Olsen*[1], *Vaibhava Goel*[1]

[1]IBM T. J. Watson Research, Yorktown Heights, NY, USA
[2]Dept. of Computer Science, University of Illinois at Urbana-Champaign, IL, USA
[3]IBM Research, Prague, Czech Republic

{etiennem,pederao,vgoel}@us.ibm.com chen192@uiuc.edu petr_fousek@cz.ibm.com

10.21437/Interspeech.2009-82

Abstract

Discriminative training of the feature space using the minimum phone error objective function has been shown to yield remarkable accuracy improvements. These gains, however, come at a high cost of memory. In this paper we present techniques that maintain fMPE performance while reducing the required memory by approximately 94%. This is achieved by designing a quantization methodology which minimizes the error between the true fMPE computation and that produced with the quantized parameters. Also illustrated is a Viterbi search over the allocation of quantization levels, providing a framework for optimal non-uniform allocation of quantization levels over the dimensions of the fMPE feature vector. This provides an additional 8% relative reduction in required memory with no loss in recognition accuracy.

Index Terms: Discriminative training, Quantization, Viterbi

1. Introduction

Discriminative training of the feature space using the minimum phone error objective function was introduced by Povey et. al. in [1], and enhanced in [2]. On our tasks this technique has given remarkable improvements. For instance, in an embedded setup the sentence error rate for a maximum likelihood trained system was 5.83%, a system built with model space discriminative training was 4.89%, and with feature space discriminative training (fMPE) was 3.76%.

The price of these gains is a parameter space consisting of millions of parameters, and recognition accuracy rapidly degrades when the number of parameters are reduced. This introduces a tradeoff in embedded ASR systems, where optimal fMPE performance translates into unacceptable consumption of memory.

In this paper we investigate techniques to maintain optimal fMPE performance while reducing the required memory. fMPE is reviewed in Section 2. Sections 3 through 6 provide details of the proposed compression scheme, and results are presented in Section 7.

2. fMPE Parameters and Processing Pipeline

The fMPE process can be described by two fundamental stages. The first stage, level 1, relies on a set of Gaussians $\mathcal{G}$ to project the input d-dimensional feature x_t to the offset features.

$$\mathbf{o}(t,g,i) = \begin{cases} \gamma_g \frac{\left(x_t^{(i)} - \mu_g^{(i)}\right)}{\sigma_g^{(i)}} & \text{if } i \le d \\ 5\gamma_g & \text{if } i = d+1 \end{cases} \tag{1}$$

where t denotes time, and i denotes offset dimension. γ_g is the posterior probability of $g \in \mathcal{G}$ given x_t, where $g(x_t) = \mathcal{N}(x_t; \mu_g, \sigma_g)$. $\mathcal{G}, g = 1, \ldots, G$, is arrived at by clustering the Gaussians of original acoustic model.

In general $\mathbf{o}(t,g,i)$ contains $(d+1) * G$ elements. For computational efficiency all γ_g below a threshold th are set to 0.0; resulting in a sparse $\mathbf{o}(t,g,i)$.

The output of level 1 is

$$\begin{aligned} b(t,j,k) &= \sum_{g,i} M^1(g,i,j,k)\mathbf{o}(t,g,i) & (2)\\ &= \sum_{g:\gamma_g > \mathrm{th}} \sum_i M^1(g,i,j,k)\mathbf{o}(t,g,i). & (3) \end{aligned}$$

where tensor M^1 is parameterized by Gaussian $g \in \{1 \cdots G\}$, offset dimension $i \in \{1 \,..\, (d+1)\}$, and by the outer-context $j \in \{1 \,..\, (2\mathrm{octx}+1)\}$ and final output dimension $k \in \{1 \,..\, d\}$. The next stage of fMPE, level 2, takes as input $b(t+l-\mathrm{ictx}-1, j, k)$ for $l \in \{1 \,..\, (2\mathrm{ictx}+1)\}$ and computes its output as

$$\delta(t,k) = \sum_j \sum_l M^2(j,k,l) b(t+l-\mathrm{ictx}-1, j, k) \tag{4}$$

$\delta(t,k)$ is added to $x_t(k)$ to compute the fMPE features.

In our typical setup, G is 512, d is 40, octx is 4, and ictx is 8. This results in M^1 with $512*41*40*9 = 7557120$ parameters. The posterior threshold th is typically 0.1, resulting in a small number of active Gaussians per x_t. For each active Gaussian, level 1 requires $41*40*9 = 14760$ floating point operations. At level 2, M^2 contains $9*40*(2*8+1) = 6120$ parameters, and computation of $\delta(t,k)$ at level 2 requires 6120 floating point operations.

As seen above, the level 1 fMPE process dominates in the amount of CPU and memory used. For the example given here, 7.5 million M^1 parameters use 30.5M of memory, 50 times the memory used in our standard embedded acoustic model, which requires approximately 600K of memory.

In the following section we investigate reducing fMPE memory and CPU requirements by changing parameters such octx, ictx, and G.

3. Parameter Impact on fMPE Performance

Figure 1 illustrates the impact of reducing octx, G, and ictx on ASR sentence error rate (SER). The test set shown in Figure 1, is comprised of 39K sentences, 206K words recorded in the car at 0, 30 and 60 mph.

From Figure 1, we see that as our memory usage drops from 30M to 1M we have a 25% increase in SER for the parameterized curve.

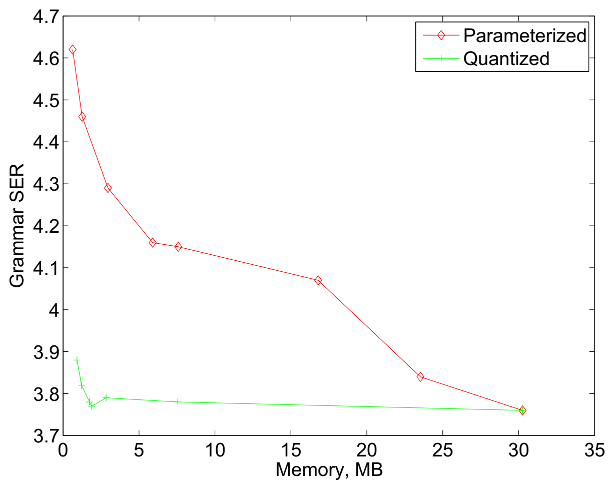

Figure 1: Impact of reducing fMPE memory usage on SER. The curve labelled “Parameterized” is obtained by changing fMPE parameters as discussed in Section 3. The curve “Quantized” is obtained by quantizing and learning quantization levels as discussed in Sections 4 and 5.

4. Quantization of Level 1 Transform

We tried the following schemes for quantizing the level 1 transform M^1

Global, linear (GlobalL): All entries of M^1 were quantized using a single quantization table. The min and max values were determined, and the range between min and max was linearly divided into desired number of quantization levels.

Per Gaussian, k-means (GaussK): Parameters corresponding to each Gaussian index g in $M^1(g,i,j,k)$ were quantized separately using their own quantization table. The K-means algorithm [3] was used to determine the quantization levels.

Per Dimension, k-means (DimK): Parameters corresponding to each dimension index k were quantized separately using their own quantization table. K-means algorithm described above was used to determine the quantization levels.

Quantization levels were further optimized as described in the following.

5. Optimization of Level 1 Transform Quantization

Let $\delta^Q(t,k)$ denote the feature perturbation obtained using the quantized level 1 transform M^{1Q}. To learn M^{1Q}, we minimize

$$E = \sum_{t,k} \left(\delta(t,k) - \delta^Q(t,k)\right)^2 \tag{5}$$

Using indicators $I_p(g,i,j,k)$, and quantization table $\mathbf{q} = \{q_p\}$ $M^{1Q}(g,i,j,k)$ can be written as

$$M^{1Q}(g,i,j,k) = \sum_p q_p I_p(g,i,j,k). \tag{6}$$

To ensure that $M^{1Q}(g,i,j,k)$ is equal to one of the quantization values in $\mathbf{q}$, we impose the additional constraint that for each (g,i,j,k) only one of $I_p(g,i,j,k)$ can be 1.
The quantized level 1 features, corresponding to (3) are

$$b^Q(t,j,k) = \sum_p q_p \sum_{g,i} I_p(g,i,j,k)\mathbf{o}(t,g,i), \tag{7}$$

and the quantized perturbation (4)

$$\begin{aligned}\delta^Q(t,k) &= \sum_p q_p \sum_{j,l} M^2(j,k,l) \\ &\quad \times \sum_{g,i} I_p(g,i,j,k)\mathbf{o}(t+l-ictx-1,g,i).\end{aligned} \tag{8}$$

We define the level 1 statistic as

$$S^1(t,j,k,p) = \sum_{g,i} I_p(g,i,j,k)o(t,g,i), \tag{9}$$

and define the level 2 statistic

$$S^2(t,k,p) = \sum_{j,l} M^2(j,k,l)S^1(t+l-ictx-1,j,k,p). \tag{10}$$

The quantized perturbation (8) becomes

$$\delta^Q(t,k) = \sum_p q_p\, S^2(t,k,p), \tag{11}$$

and the error (5) is a quadratic in $\mathbf{q}$

$$\begin{aligned} E &= \sum_{t,k}\left(\delta(t,k) - \sum_p q_p S^2(t,k,p)\right)^2, \\ &= \sum_k A(k) + \mathbf{q}^T\mathbf{B}(k)\mathbf{q} - 2\mathbf{q}^T\mathbf{c}(k) \end{aligned} \tag{12}$$

where

$$\begin{aligned} A(k) &= \sum_t \delta(t,k)^2 \\ B(k,p_1,p_2) &= \sum_t S^2(t,k,p_1)S^2(t,k,p_2) \\ c(k,p) &= \sum_t \delta(t,k)S^2(t,k,p). \end{aligned}$$

The minimum is achieved at

$$\hat{\mathbf{q}} = \left(\sum_k \mathbf{B}(k)\right)^{-1} \sum_k \mathbf{c}(k) \tag{13}$$

If we have a separate quantization table $\mathbf{q}(k)$ per dimension, then $E = \sum_k E(k)$ with

$$E(k) = A(k) + \mathbf{q}^T(k)\mathbf{B}(k)\mathbf{q}(k) - 2\mathbf{q}^T(k)\mathbf{c}(k) \tag{14}$$

and minimum attained at

$$\hat{\mathbf{q}}(k) = \mathbf{B}^{-1}(k)\mathbf{c}(k) \tag{15}$$

with

$$\hat{E}(k) = A(k) + \hat{\mathbf{q}}^T(k)\mathbf{B}(k)\hat{\mathbf{q}}(k) - 2\hat{\mathbf{q}}^T(k)\mathbf{c}(k) \tag{16}$$

We note that the sufficient statistics, and consequently the optimum $\hat{\mathbf{q}}(k)$, are a function of $I_q(g,i,j,k)$. Further reduction in error may be obtained by reassigning M^1 entries to quantization levels (i.e. updating $I_q(g,i,j,k)$) and iterating. However, we did not do that in this paper.

5.1. Level 1 and 2 Scaling

From equations (3) and (4), we note that $\delta(t,k)$ can be expressed in terms of the product $M^1(g,i,j,k)M^2(j,k,l)$. It is therefore invariant to the following form of scaling

$$\frac{M^1(g,i,j,k)}{a(j,k)}\left(M^2(j,k,l)a(j,k)\right). \quad (17)$$

The quantization levels do not satisfy the same scale invariance, and so $\hat{\mathbf{q}}(k)$ and the accuracy of the quantization will change with the scaling $a(j,k)$.
With the $a(j,k)$ scaling the level 2 statistic (10) becomes

$$\begin{aligned} S^2(t,j,k,p) &= \sum_l a(j,k)M^2(j,k,l) \\ &\quad \times S^1(t+l-\text{ictx}-1,j,k,p), \quad (18) \end{aligned}$$

where we have removed the summation across outer dimension j. The error to be minimized becomes

$$\begin{aligned} E &= \sum_t \delta(t,k)^2 - 2\sum_t \delta(t,k)\sum_p q_p(k)\sum_j S^2(t,j,k,p) \\ &\quad + \sum_{p_1,p_2} q_{p_1}(k)q_{p_2}(k) \\ &\quad \times \sum_t \sum_{j_1,j_2} S^2(t,j_1,k,p_1)S^2(t,j_2,k,p_1). \quad (19) \end{aligned}$$

Given that we know the analytic minimum with respect to $\mathbf{q}(k)$ the per dimension error is

$$\begin{aligned} E(k) &= A(k) + \sum_{j_1} a(j_1,k)\mathbf{c}_{j_1}^T(k) \\ &\quad \times \left(\sum_{j_3,j_4} a(j_3,k)a(j_4,k)\mathbf{B}_{j_3,j_4}(k)\right)^{-1} \\ &\quad \times \left(\sum_{j_2} a(j_2,k)\mathbf{c}_{j_2}(k)\right), \quad (20) \end{aligned}$$

where

$$\begin{aligned} \mathbf{B}(k) &= \sum_{j_1,j_2} a(j_1,k)a(j_2,k)\mathbf{B}_{j_1,j_2}(k) \\ B_{j_1,j_2}(k,p_1,p_2) &= \sum_t S^2(t,j_1,k,p_1)S^2(t,j_2,k,p_2) \\ \mathbf{c}(k) &= \sum_j a(j,k)\mathbf{c}_j(k) \\ c_j(k,p) &= \sum_t \delta(t,k)S^2(t,j,k,p). \quad (21) \end{aligned}$$

It is not clear how to optimize (20) analytically with respect to $\{a(j,k)\}_j$, therefore we resort to numerical optimization. The gradient of $E(k)$ is given by

$$\begin{aligned} \frac{\partial E(k)}{\partial a(j,k)} &= 2\mathbf{c}(k)^T\mathbf{B}(k)^{-1}\mathbf{c}_j(k) - 2\mathbf{c}^T(k)\mathbf{B}(k)^{-1} \times \\ &\quad \left(\sum_{j2} a(j_2,k)\mathbf{B}_{j_2,j}(k)\right)\mathbf{B}(k)^{-1}\mathbf{c}. \quad (22) \end{aligned}$$

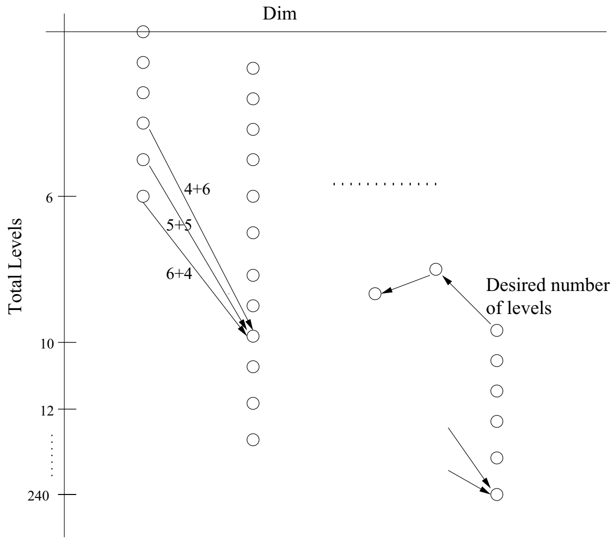

Figure 2: Viterbi search for optimal target number of quantization levels.

6. Optimal Quantization Level Allocation with a Viterbi Search

Let $n(k)$ denote the number of levels in $\mathbf{q}(k)$. The size of M^{1Q} is determined by the total number of levels $n = \sum_k n(k)$.
The independence of errors $E(k)$ across dimensions allows us to formulate a Viterbi procedure that, given a desired n, finds optimal allocation $n(k)$. Prior to carrying out this procedure, we find, for each dimension k, $E(k,n(k))$ for $n(k) = 1,\ldots,L$.

The Viterbi procedure is

1. Initialize $V(1,m) = E(1,m)$ for $m = 1,\ldots,L$
2. For $k = 2 \ldots d$, apply the recursive relation
$$V(k,m) = \min_{a+b=m}\left(E(k,a) + V(k-1,b)\right)$$
3. Once $k = d$ is reached, backtrack to find level assignment for each dimension.

This technique is illustrated in figure 2, where we have a 40 dimensional feature vector with a maximum of 6 available quantization levels per dimension. Therefore the maximum total number of quantization levels is 240, and the minimum is 40. The back pointers indicate back tracking from a desired n to obtain optimal allocation $n(k)$.

7. Experimental Evaluations

7.1. Experimental Setup

The ASR system is evaluated on various grammar tasks relevant to the in-car embedded domain, this includes digit strings, command and control, and navigation. There are 39K sentences and 206K words in the testset.
To obtain another error rate measurement, we built a word unigram LM and decoded test set using that.
The basic audio features extracted by the front-end are 13 dimensional Mel-frequency cepstral coefficients at a 15 msec frame rate. After cepstral mean normalization, nine consecutive frames are concatenated and projected onto a 40 dimensional space through an LDA/MLLT cascade. The recognition

system is built on three-state left-to-right phonetic HMMs with 865 context dependent states. The context tree is based on a word internal model, spanning 5 phonemes (2 phonemes on either side). Each context dependent state is modeled with a mixture of diagonal Gaussians for a total of 10K Gaussian models. The models are trained on 790 hours of data.

System	grammar SER	unigram WER	Mem (MB)
fMPE baseline	3.76	17.91	30.2
GlobalL 256 lvl	3.78	18.00	7.56
GlobalL 16 lvl	4.17	19.57	3.78
GaussK 10 lvl	3.83	18.08	3.15
DimK 6 lvl	3.83	18.26	2.83
+ learned	3.79	17.95	2.83
+ scaled	3.80	17.96	2.83
DimK 4 lvl	3.86	18.61	1.89
+ learned	3.77	17.97	1.89
+ scaled	3.79	17.98	1.89
DimK 2 lvl	4.31	20.37	0.94
+ learned	3.83	18.91	0.94
+ scaled	3.85	18.90	0.94

Table 1: ASR performance with baseline and various configurations. GlobalL, GaussK, and DimK are as described in Section 4. The rows labeled "+ learned" indicate levels obtained according to (15). Rows labeled "+scaled" indicate levels obtained with scaling, as discussed in Section 5.1.

System	learned	learned + scaled
DimK 1 lvl	0.39	1.06
DimK 2 lvl	27.08	28.06
DimK 3 lvl	36.12	37.13
DimK 4 lvl	35.59	36.41
DimK 5 lvl	34.06	35.01
DimK 6 lvl	30.64	31.62

Table 2: Percent reduction in error due to learning quantization levels. The reductions are measured relative to the initial assignment using DimK method, as described in Section 4. "learned" refers to obtaining quantization levels according to (15), and "learned + scaled" refers to obtaining the levels as described in Section 5.1.

7.2. Experimental Results

Our key result is summarized in Figure 1. The bottom curve shows impact of fMPE transform quantization on memory and grammar sentence error rate (SER). We achieve a reduction of 94% in memory (30.2M is reduced to 1.89M), with almost no increase in SER.
These numbers are presented in some more detail in Table 1. As seen from this Table, our distortion minimization approach allows us to quantize using 4 levels (2 bits) resulting in memory usage of 1.89MB, while affecting SER from baseline of 3.76 to 3.77 and WER using the unigram LM from 17.91 to 17.97. For the 2 level (1 bit) case, the unigram performance has a 14% relative degradation (17.91 to 20.37) which reduces to 5% relative degradation after optimizing the quantization levels.

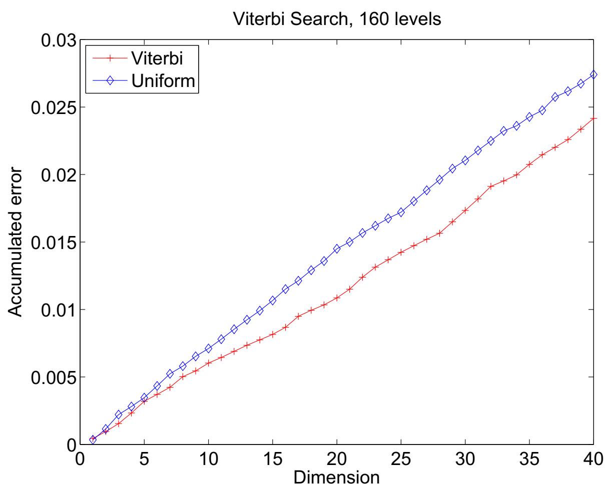

Figure 3: Accumulated error as a function of dimension under Viterbi and uniform allocation.

Table 2 presents the error reduction obtained by learning the quantization levels. This table shows that a large reduction is achieved with learning. Most of this reduction comes from (15). A smaller (approximately an additional 1.0% relative reduction) is achieved with the scaling discussed in section 5.1. We note that the slight reduction in error by use of scaling is not reflected in the recognition error rates shown in table 1.
Figure 3 shows accumulated error as a function of dimension under Viterbi and uniform allocations. The target number of levels was 160, corresponding to a uniform assignment of 4 levels per dimension. The Viterbi algorithm gives approximately 12% relative reduction in the error over uniform assignment.
Looking at Table 3, we see that this reduction in quantization error does not translate to a reduction in grammar SER. However, the Viterbi algorithm provides flexibility to pick the optimal assignment for any desired total number of levels. In particular, we can choose less than 80 total levels, resulting in an average of less than 1 bit per dimension.

System	Grammar SER	Unigram WER	Mem (MB)	Quant Error
DimK 4 lvl	3.79	17.98	1.89	0.027
DimK 2 lvl	3.85	18.90	0.94	0.099
160 lvl Viterbi	3.79	18.01	1.85	0.024
147 lvl Viterbi	3.78	18.08	1.74	0.028
100 lvl Viterbi	3.82	18.49	1.22	0.061
80 lvl Viterbi	3.88	19.06	0.92	0.098
70 lvl Viterbi	4.07	19.76	0.71	0.125

Table 3: ASR performance, quantized transform size (MB), and quantization error for uniform vs. Viterbi allocation.

8. References

[1] Povey, D., Kingsbury , B., Mangu, L., Saon, G., Soltau, H., Zweig., G. "FMPE : Discriminatively Trained Features for Speech Recognition", in ICASSP, 2005.

[2] Povey, D. "Improvements to fMPE for Discriminative Training of Features", in Interspeech, 2005.

[3] Duda, R., Hart, P., Stork, D. "Pattern Classification, Second Edition", John Wiley & Sons, Inc., 2001.