Dynamic estimation of a noise over estimation factor for Jacobian-based adaptation

Christophe Cerisara, Jean-Claude Junqua, Luca Rigazio

DYNAMIC ESTIMATION OF A NOISE OVER ESTIMATION FACTOR FOR JACOBIAN-BASED ADAPTATION

Christophe Cerisara

LORIA UMR 7503
BP 239 - 54506 Vandoeuvre
FRANCE

Jean-Claude Junqua, Luca Rigazio

Panasonic Speech Technology Laboratory
3888 State St. suite 202, Santa Barbara, CA, 93105
USA

ABSTRACT

In this paper we propose an enhancement of the Jacobian adaptation by estimating automatically a noise over estimation factor which yields to a closer approximation of Parallel model combination (PMC) than the traditional Jacobian adaptation. Noise over estimation factors are estimated at run-time for a set of clustered Gaussians obtained on the training set. Experiments conducted on a French natural number database show that similar performance as PMC can be obtained at the expense of a slight increase in computational complexity as compared to Jacobian adaptation.

1. INTRODUCTION

For most automatic speech recognition systems, adapting the acoustic models to the environment is an efficient solution to reduce the mismatch between the training and test corpus. A popular method to adapt the models is Parallel Model Combination (PMC) [1]. This method has proven to be very effective but it is generally too computationally expensive. An alternative method, Jacobian adaptation (JA), has been proposed by Sagayama et al. [2]. The main advantage of JA when compared to PMC is its very low cost, both in memory and in computation time. In the next section, we review briefly PMC and JA. Section 3 describes our new method while section 4 estimates and compares the computational costs of PMC, JA and our new method dynamic α-JA. Section 5 presents some experimental results and section 6 concludes the paper.

2. REVIEW OF FIXED α-JACOBIAN ADAPTATION

2.1. Notations

In the next sections, we use the following notations:

- The acoustic speech models used are Hidden Markov Models (HMM). As we adapt only the static components of the Gaussian means of the HMMs, we denote $\{S\}$ the set of these vectors.
- S and N represents, respectively, the spectral vectors for clean speech and noise. N^{ref} refers to the background noise in the training corpus, while N^{tar} refers to the unknown background noise in the test corpus. In practice, real speech models do not model S, but rather $\dot{S} = S + N^{ref}$.
- $C(X)$ is the cepstral representation of the spectral vector X. Then, $C(X) = F \cdot \log(X)$, where F is the Discrete Cosine Transform matrix. We note F^* the conjugate transpose of F.
- N_f and N_c are respectively the sizes of the spectral and static cepstral vectors. For the purpose of estimating the computational complexity, we assume that $N_f = 2N_c$. Similarly, we note N_g the total number of Gaussians S.
- We note hereafter $\cdot$ the usual matrix-vector multiplication and $\times$ the component wise vector-vector multiplication. In the computation of Jacobian matrices (e.g. Eq.3), diagonal matrices are used in place of vectors.

2.2. Adaptation equations of PMC and JA

We now recall the adaptation equations of PMC and Jacobian adaptation. As reviews of these adaptation methods have already been presented in previous papers, we refer the reader to [1] for a complete description of PMC and to [2] for Jacobian adaptation.

When both the cepstral models $C(\dot{S})$ and their spectral representations $\dot{S}$ are saved in memory, then PMC can adapt each model using the following equation:

$$C(S + N^{tar}) = F \cdot \log(\dot{S} - N^{ref} + N^{tar}) \qquad (1)$$

In this equation, the *logarithm* operation has the highest computational cost, and it must be computed for every $\dot{S}$, every time the models are adapted.

Jacobian adaptation adapts the models with a linear operation. This greatly reduces the cost of the adaptation as compared to PMC:

$$C(S + N^{tar}) \simeq C(\dot{S}) + J_S \cdot \big(C(N^{tar}) - C(N^{ref})\big) \quad (2)$$

where J_S is the Jacobian matrix associated with S. It is computed with the following equation

$$J_S = F \cdot \frac{N^{ref}}{S + N^{ref}} \cdot F^* \quad (3)$$

The Jacobian matrices are computed at training time. The memory cost required to store them can be considerably reduced by using a dimensionality reduction technique, as proposed in [3].

2.3. α-JACOBIAN ADAPTATION

When the test environment is much noisier than the training environment, JA often under-estimates the effect of noise. The basic idea of α-Jacobian adaptation is to use a parameter α to compensate for this effect, and to boost the estimation of the background noise.

In [3], we proposed a first version of this adaptation scheme using the following adaptation equation:

$$C(S + N^{tar}) \simeq C(\dot{S}) + F \cdot \frac{\alpha N^{ref}}{S + \alpha N^{ref}} \cdot F^* \cdot \big(C(N^{tar}) - C(N^{ref})\big) \quad (4)$$

The parameter α is a scalar. It is identical for all the models S, and it is estimated at training time on a development corpus. Hereafter, We will refer to this adaptation scheme as *fixed* α- Jacobian adaptation.

This method gives interesting results, but presents the following drawbacks:

- It is very unlikely that all the Gaussians and all the cepstral coefficients share the same behavior when the background noise increases;
- The value of α, computed on a development corpus, might be dependent on this development corpus.

We address these issues in the next section.

3. DYNAMIC α-JACOBIAN ADAPTATION

3.1. Dynamic α-Jacobian adaptation equation

We propose to dynamically estimate the value of α. We can not use Eq.4 any more, as Jacobian matrices would need to be recomputed every time a new α is estimated. Therefore, we propose the following adaptation equation:

$$C(S + N^{tar}) \simeq C(\dot{S}) + \alpha_S \times \Big[F \cdot \tfrac{N^{ref}}{S+N^{ref}} \cdot F^* \cdot \big(C(N^{tar}) - C(N^{ref})\big)\Big] \quad (5)$$

where α_S is a N_c-dimensional vector. The Jacobian matrices are still computed at training time, and the vectors $J_s \cdot \big(C(N^{tar}) - C(N^{ref})\big)$ are multiplied by α_S during testing.

An important difference between Eq.5 and Eq.4 is that a different α_S is used for each coefficient of each Gaussian mean S. Then, our new adaptation scheme makes use of $N_g \cdot N_c$ parameters, instead of 1 parameter for Eq.4. We will see in the next section how it is possible to reduce the number of these additional parameters.

3.2. Dynamic estimation of α_S

Jacobian adaptation may be considered as a linear approximation of the "exact" non-linear adaptation scheme given in Eq.1. In this sense, the optimal value of α_S verifies:

$$C(\dot{S}) + \tilde{\alpha}_S \times J_S \cdot \big(C(N^{tar}) - C(N^{ref})\big) = F \cdot \log(\dot{S} - N^{ref} + N^{tar}) \quad (6)$$

This optimal value is:

$$\tilde{\alpha}_S = \frac{C(S + N^{tar}) - C(S + N^{ref})}{J_S \cdot (C(N^{tar}) - C(N^{ref}))} \quad (7)$$

Obviously it is not possible to compute $\tilde{\alpha}_S$ for every S, as the cost of this operation would exceed the cost of PMC. Our main goal is now to estimate these optimal values $\tilde{\alpha}_S$ at a low cost. Many solutions exist. Hereafter, are some of them that we experimented with.

- We evaluated the use of a multiple linear regression scheme to estimate the value of α_S during testing:

  $$\alpha_S = a_{S,0} + \sum_{k \geq 1} a_{S,k} X_k \quad (8)$$

  The parameters $(a_{S,k})$ of this multiple regression were trained on several development corpus. But the main difficulty of this method is to find one or several pertinent variables (X_k) that can be used to compute the multiple regression. We tested for example $X_1 = N^{tar}$, but results were disappointing. Another potentially interesting solution would be to choose some $X_k = \tilde{\alpha}_{S_k}$.

- Another solution, which also makes use of a set of $(\tilde{\alpha}_S)$ generated on a development corpus, consists to project these vectors onto a low-dimensional subspace.

An orthonormal basis of this subspace can be obtained by the singular value decomposition of $(\tilde{\alpha}_S)$. Then, during testing, if N_d is the number of dimensions of this subspace, N_d optimal coefficients of the target vector α_S are computed. The coordinates of these coefficients in the subspace are then computed by solving a linear system, and the whole vector α_S can thus be estimated. This method gave good results, but happened to be less efficient than the following proposal.

- The proposed solution whose results are presented in this paper, consists simply to cluster the vectors S into a few classes and to share the same α_S for all the S that belong to the same class. One vector S per class is chosen to represent that class. Then, $\tilde{\alpha}_S$ is computed for this vector. The main issue in this method is to efficiently cluster the set of S, as explained in the next section.

3.3. Gaussian mean clustering

Our preliminary experiments show that the choice of the clustering algorithm for the Gaussian mean vectors S has a great influence on the precision of adaptation. We first tested two "blind" classical clustering algorithms: the *k-means* and a hierarchical bottom-up clustering algorithms. However, the best results for now were obtained using the following clustering algorithm, inside the HMM states:

- We first decide how many clusters n should be used in each HMM state. The total number of clusters is then $N_s = N_{states} \cdot n$, and verifies $N_{states} \leq N_s \leq N_g$.
- The n most likely Gaussians S of each state are chosen to represent one cluster each.
- Finally, the remaining Gaussians in each state are assigned to one of the n clusters, based on the Euclidean distance criterion.

4. COMPARISON OF THE COSTS OF PMC, JAC AND α-JAC

The following costs represent the number of basic operations (i.e. scalar additions and multiplications) that are required to perform adaptation of all the Gaussians S. The cost of the $\log$ operation is assumed to be equal to ten times the cost of a basic operation, as benchmarked on a pentium processor. Some simplifications, which essentially consist of taking away additive terms that might be neglected, are realized.

4.1. Cost of PMC

Using Eq.1, we can compute the lowest possible cost for PMC, by assuming that both the models $C(\dot{S})$ and $\dot{S}$ are stored in memory, and that the target noise is available in the spectral domain.

The cost of PMC is then

$$C_{PMC} = 6N_gN_c^2 \tag{9}$$

4.2. Cost of Jacobian adaptation

By using Eq.2, the cost of Jacobian adaptation is

$$C_{JAC} = 2N_gN_c^2 \tag{10}$$

4.3. Cost of dynamic α-Jacobian adaptation

Let us first consider the cost to compute $\tilde{\alpha}_S$ (Eq.7). This value is computed for each of the N_s clusters of S. The cost to compute $C(S + N^{tar})$ is the cost of PMC, i.e. $6N_c^2$ per cluster. Moreover, as α-Jacobian adaptation always computes $J_S \cdot \big(C(N^{tar}) - C(N^{ref})\big)$ for all the Gaussians S, the denominator does not need to be computed again. Thus, the cost to compute all the α_S is

$$C_\alpha \simeq 6N_sN_c^2 \tag{11}$$

Once all the α_S have been computed, these α_S are multiplied by the vector $J_S \cdot \big(C(N^{tar}) - C(N^{ref})\big)$. Then, α-Jacobian adaptation is similar to the classical Jacobian adaptation. Thus, the total cost of α-Jacobian adaptation is

$$C_{\alpha-JAC} = 6N_sN_c^2 + N_gN_c + 2N_gN_c^2 \tag{12}$$

4.4. Scalability of dynamic α-Jacobian adaptation

It is possible, by varying N_s, to change the precision of adaptation as well as its cost. However, this scalability is limited, as the number of clusters n per state may only range from 1 to the number of Gaussians per state. Furthermore, α-Jacobian adaptation is interesting only when its cost is much lower than the cost of PMC. Based on the results of section 4, we think that a reasonable cost for α-Jacobian adaptation should be less than $4N_gN_c^2$. Thus, we want

$$\begin{aligned} & 6N_sN_c^2 + N_gN_c + 2N_gN_c^2 < 4N_gN_c^2 \\ \equiv \quad & N_s < \frac{N_g}{3} \end{aligned} \tag{13}$$

5. EXPERIMENTS

5.1. Experimental set-up

5.1.1. Recognition system

We used the HTK toolkit [4]. The models are left-to-right HMMs with 13 emitting states and 16 Gaussian per states. Signal encoding is realized every 10 ms, with overlapping windows of 20 ms length. These windows are coded into 13 MFCC coefficients, including c_0, plus the first and second derivatives. Diagonal covariances are used.

5.1.2. Task and database

The task consists in recognizing unconstrained French sequences of natural numbers. We have chosen the VODIS database [5], which has been recorded in three different cars, at different speeds and in different noisy conditions. Table 1 shows some of the variabilities in the database.

	Training corpus	**Test corpus**
# of sentences	2757	697
average SNR	31 dB	12 dB
Fan on	22 %	19 %
Window open	24 %	5 %
Raining	4 %	26 %

Table 1. Database recording conditions

We have trained 25 digits and numbers models on the signal recorded by 140 speakers, with a high-quality close-talking microphone. Testing has been realized on 20 different speakers with a far-talking microphone fixed at the rear-view mirror inside the car.

5.2. Experimental results

Adaptation is performed only on the static coefficients of the Gaussian mean vectors. The target noise is estimated using the first 100 milliseconds of silence at the beginning of each sentence. Recognition results of PMC, JA, *fixed* α-JA and *dynamic* α-JA are reported in table 2.

For *fixed* α-JA, α has been estimated on a development corpus composed of 100 sentences recorded in the same conditions as the test corpus. The development environment has been chosen close to the testing environment, in order to obtain the best recognition accuracy from *fixed* α-JA. In such conditions, we found $\alpha = 2$. As shown in table 2, *fixed* α-JA does not give better accuracy results than JA: This might be due to the variable conditions of the database, that makes it difficult to find a single scalar α which provides a good compensation bias. Moreover, the relatively high SNR of the test corpus tends to bring JA and *fixed* α-JA closer. This is confirmed by the fact that the optimal α found in such conditions is quite close to one, i.e. to classical JA. This is a particular case, as on most of the other databases we have tested, α was greater than 5. With more parameters than *fixed* α-JA, *dynamic* α-JA can improve recognition rates, as shown in table 2.

For *dynamic* α-JA, we have chosen $N_s = 3$ clusters in each state. This number comes from Eq.13, which defines the available range of N_s as $1 \leq N_s \leq 5$.

Adaptation	**Recognition accuracy**	**Cost**
None	51.7 %	0 Mop
JA	71.8 %	1.7 Mop
fixed α-JA	71.7 %	1.8 Mop
dynamic α-JA	73.9 %	2.7 Mop
PMC	73.4 %	5.1 Mop

Table 2. Recognition results and computational costs (in millions of operations) of several adaptation algorithms

6. CONCLUSIONS

We have proposed an enhancement of the Jacobian adaptation scheme proposed by Sagayama et al. Intuitively, a noise over-estimating factor α is used to compensate for the difference between the adaptation bias given by Jacobian adaptation and the "optimal" bias given by PMC. We propose a method to dynamically estimate this bias, and show that the cost of this new adaptation algorithm is still much lower than the cost of PMC. Experiments carried out on a French natural number database recorded in a car further show that the recognition results are comparable with those of PMC. Future work will include the testing of the algorithm on several other realistic databases to further validate the effectiveness of the proposed method. Furthermore, we plan to improve the Gaussian clustering algorithm by considering techniques such as the Bhattacharya metric or entropy measures. Indeed, we have already observed that the clustering criterion has a great impact on the performances of the algorithm, and future researches should concentrate on this part of the algorithm.

7. REFERENCES

[1] M. Gales, "Predictive model-based compensation schemes for robust speech recognition," *Speech Communication*, vol. 25, pp. 49–74, 1998.

[2] S. Sagayama, Y. Yamaguchi, S. Takahashi, and J. Takahashi, "Jacobian approach to fast acoustic model adaptation," in *ICASSP'97*, Munich, 1997, pp. 835–838.

[3] Christophe Cerisara, Luca Rigazio, Robert Boman, and Jean-Claude Junqua, "Transformation of Jacobian matrices for noisy speech recognition," in *ICSLP'2000*, Beijing, China, October 2000, vol. 1, pp. 369–372.

[4] P. Woodland and S. Young, "The htk continuous speech recogniser," in *Eurospeech'93*, Berlin, Sept. 1993, pp. 2207–2219.

[5] C. Gassert and J.-F. Mari, "Spécification, réalisation et validation d'un corpus oral pour la reconnaissance de la parole dans une voiture," in *Journées d'Etude de la Parole*, 1998.