Training Log-Linear Acoustic Models in Higher-Order Polynomial Feature Space for Speech Recognition

M. Tahir[1], H. Huang[1,2], R. Schlüter[1], H. Ney[1,3], L. ten Bosch[2], B. Cranen[2], L. Boves[2]

[1] Human Language Technology and Pattern Recognition
Computer Science Department, RWTH Aachen University, Germany
[2]Centre for Language and Speech Technology, Radboud University Nijmegen, the Netherlands
[3]Spoken Language Processing Group, LIMSI CNRS, Paris, France
{tahir,schlueter,ney}@cs.rwth-aachen.de, {h.huang,l.tenbosch,b.cranen,l.boves}@let.ru.nl

Abstract

The use of higher-order polynomial acoustic features can improve the performance of automatic speech recognition. However, the dimensionality of the polynomial representation can be prohibitively large, making the training of acoustic models using polynomial features for large vocabulary ASR systems infeasible. This paper presents an iterative polynomial training framework for acoustic modeling, which recursively expands the current acoustic features into their second-order polynomial feature space. In each recursion the dimensionality is reduced by a linear projection, such that increasingly higher order polynomial information is incorporated while keeping the dimensionality of the acoustic models constant. Experimental results obtained for a large-vocabulary continuous speech recognition task show that the proposed method outperforms conventional mixture models.

Index Terms: Discriminative training, polynomial features, feature transformation

1. Introduction

The use of second-order polynomial features improves the performance of phone classification [1], and automatic speech recognition (ASR) systems [2, 3]. Higher-order polynomial features [4, 5], as a natural extension to the second-order polynomial features, might convey discriminative information which adds to the first-/second-order polynomial features and might improve ASR performance. Pioneering research [6, 7, 8] has already reported a decrease of the word error rate by using third-order polynomials. Therefore, the performance of ASR systems might be expected to benefit from extending the feature space by adding higher-order (≥ 4) polynomials.

However, expanding the feature space by adding ≥ 4-order polynomials for acoustic modeling of ASR may be infeasible in actual practice, especially if a long contextual window is used [8] to characterize each frame. The resultant higher-order polynomial feature space will have a prohibitively large dimension, which makes it practically impossible to train the acoustic models. Motivated by these facts, this paper addresses a novel computationally feasible approach, by seeking a low-dimensional subspace in the higher-order polynomial feature space that contains most of the relevant information. The basic idea is to iteratively expand the feature space via its second-order polynomials and log-linearly train a low-dimensional representation [9, 3] in the expanded feature space to formulate a new feature space. During the iterative procedure, the higher-order polynomials (≥ 4 degree) are implicitly involved in the log-linear acoustic models. The second-order polynomial expansion and the subsequent dimensionality reduction can also guarantee that the combination of (higher-order) polynomials is compact and the computational burden is affordable.

This paper is organized as follows: Section 2 briefly reviews the log-linear acoustic models and their representation with polynomials. The iterative training framework is then presented in Section 3. Section 4 and Section 5 show the experimental setup and results in a large-vocabulary corpus, followed by Section 6 with conclusions.

1.1. Relation to Prior Work

This paper presents an algorithmic framework that makes it possible to train ≥ 4-order polynomials for ASR, whereas previous works on explicit polynomial feature training could only afford to use ≤ 3-order polynomials [6, 7, 8, 2, 3]. Most works exploring the higher-order polynomial feature space rely on the use of polynomial kernels [4, 5], which is infeasible in the case of the large amounts of training data that are necessary in ASR.

2. Log-Linear Acoustic Modeling

Assume that the acoustic representation of a speech frame is denoted by $\vec{x}^\top \in \mathbf{R}^D$, with a label belonging to one of S triphone classes. Each class $s = 1, 2, \ldots, S$ can be modeled by Gaussians with the parameter set $\theta_s = \{\mu_s, \Sigma_s\}$, which can be trained by Maximum Likelihood (ML) estimators. Gaussian models can be converted to log-linear models [10] by collecting the quadratic and first-order terms of $\vec{x}$ (as well as a constant):

$$p_\theta(s|\vec{x}) = \frac{\exp(\vec{x}^\top \Lambda_s \vec{x} + \lambda_s^\top \vec{x} + \alpha_s)}{\sum_{s'} \exp(\vec{x}^\top \Lambda_{s'} \vec{x} + \lambda_{s'}^\top \vec{x} + \alpha_{s'})} \quad (1)$$

in which $\theta = \{\Lambda_s, \lambda_s, \alpha_s, s = 1, 2, \ldots, S\}$. The most conspicuous advantage of log-linear models over Gaussian models lies in the fact that the exponential terms do not strictly need to be probability distributions.

H. Ney was partially supported by a senior chair award from DIGITEO, a French research cluster in Ile-de-France.

The research of Heyun Huang was funded by the European Community's Seventh Framework Programme [FP7/2007-2013] under grant agreement no. 213850 SCALE.

The research leading to these results has received funding from the European Union Seventh Framework Programme EU-Bridge (FP7/2007-2013) under grant agreement N287658.

The quadratic log-linear models can be simplified by setting $\Lambda = \mathbf{0}$ [8, 10, 3]:

$$p_\theta(s|\vec{x}) = \frac{\exp(\lambda_s^\top \vec{x} + \alpha_s)}{\sum_{s'} \exp(\lambda_{s'}^\top \vec{x} + \alpha_{s'})} \tag{2}$$

The Maximum Mutual Information (MMI) criterion is adopted as the frame-level objective function:

$$F(\lambda, \alpha) = -\tau_\theta ||\lambda, \alpha||^2 + \sum_{r=1}^{R} \sum_{t=1}^{T_r} \log p_{\lambda,\alpha}(s_t|\vec{x}_t) \tag{3}$$

2.1. Log-Linear Models in Polynomial Feature Space

The feature vector $\vec{x}$ can be expanded by its degree-N polynomials $\phi_N(\vec{x})$ as:

$$\phi_N(\vec{x}) = \{x_1^{i_1} x_2^{i_2} \cdots x_D^{i_D}\}, i_d \in \{0, 1, \ldots, N\} \tag{4}$$

The posterior probability estimated by log-linear training on the degree-N polynomials could be written by:

$$p_\theta(s|\vec{x}) = \frac{\exp(\lambda_s^\top \phi_N(\vec{x}) + \alpha_s)}{\sum_{s'} \exp(\lambda_{s'}^\top \phi_N(\vec{x}) + \alpha_{s'})} \tag{5}$$

2.2. Dimensionality Reduction for Log-Linear Models

A linear feature transformation matrix $\mathbf{A} \in \mathbf{R}^{D' \times card(\phi_N(\vec{x}))}$ for reducing the dimension of $\phi_N(\vec{x})$ can be introduced in the log-linear model [3, 9], such that the posterior probability is estimated as follows:

$$p_{\theta,\mathbf{A}}(s|\vec{x}) = \frac{\exp(\lambda_s^\top \mathbf{A}\phi_N(\vec{x}) + \alpha_s)}{\sum_{s'} \exp(\lambda_{s'}^\top \mathbf{A}\phi_N(\vec{x}) + \alpha_{s'})} \tag{6}$$

which reduces the dimension of λ to D'. Setting N as 1 and 2 results in the methods proposed by [9] and [3], respectively. An iterative procedure can be used to obtain $\mathbf{A}$ and λ by alternate convex optimization of one parameter set with a fixed value of the other parameter set. To implement a gradient-based optimization, an initial value of $\mathbf{A}^{(0)}$ is needed, which is computed by a conventional linear discriminant analysis [3].

This new formulation diminishes the storage cost in comparison with the full log-linear training shown in Equation 2, especially for the log-linear models in the higher-order ($N \geq 3$) polynomial feature space (Equation 5) [3]: the number of log-linear parameters in Equation 5 is greatly reduced by the projection $\mathbf{A}$, at the negligible cost of keeping $\mathbf{A}$ itself. Moreover, the high redundancy incurred by the polynomial representation definitely means that the effective dimensionality of the feature space is much lower, which also shows the importance of dimensionality reduction (by $\mathbf{A}$) in the polynomial feature space.

However, directly training the log-linear model (Equation 6) when $N \geq 3$ is not computationally feasible for a large training corpus. One reason is that the size of $\mathbf{A}$ might still be prohibitively large. More importantly, the convergence rate is very slow [6] when both the number of free parameters and training samples are large, which makes the training procedure extremely inefficient. In the next section, a novel method will be proposed to train the log-linear models and projections in higher-order polynomial feature space.

3. Proposed Method

3.1. The Iterative Polynomial Expansion Framework

Although the cardinality of the projection matrix $\mathbf{A}$ in Equation 6 is prohibitively large, one property of polynomials might make it possible to learn the linear combinations of higher-order polynomials: the feature space spanned by the second-order expansion $vec(\vec{y}\vec{y}^\top)$, $(\vec{y} = \mathbf{B}\vec{x},\ \mathbf{B} \in \mathbf{R}^{r \times D},\ r < D)$ is still a linear combination of the feature space of $\phi_2(\vec{x}) = vec(\vec{x}\vec{x}^\top)$ by noticing $\vec{y}\vec{y}^\top = \mathbf{B}\vec{x}\vec{x}^\top\mathbf{B}^\top$, where the operator $vec(\cdot)$ stands for the vectorization of a matrix. Each element of $vec(\vec{y}\vec{y}^\top)$ is a linear combination of $vec(\vec{y}\vec{y}^\top)$ and consequently projecting the feature space $vec(\vec{y}\vec{y}^\top)$ by the linear projector $\mathbf{A}$ could be regarded as an approach to learn the (constrained) linear weights for log-linear optimization.

Repeating this procedure makes it possible to learn the optimal weights on higher-order polynomials: if the original feature vector $\vec{x}$ contains the information of 2^c-order polynomials, the resultant $\vec{y}$ will reflect that of 2^{c+1}-order polynomials as shown in Figure 1. The algorithmic description is given as follows:

- **Step 1, Initialization:** Train the conventional LDA on the original feature $\vec{x}$ and project it into a low-dimensional representation, namely $\vec{z}_0 = \mathbf{W}_{LDA}\vec{x}$. Set the number of iterations $k = 0$.
- **Step 2, Second-Order Polynomial Expansion:** Generate the second-order polynomial of $\vec{z}_k$ as $\vec{z}_k\vec{z}_k^\top$. Concatenate its vectorized representation $vec(\vec{z}_k\vec{z}_k^\top)$ with $\vec{z}_k$, which results in $\vec{y}_k = [vec(\vec{z}_k\vec{z}_k^\top)|\vec{z}_k^\top] \in \mathbf{R}^{d_k(\times 1)}$.
- **Step 3, Log-Linear Training of the Projection Matrix [9]:** Train the projection matrix $\mathbf{A}_k \in \mathbf{R}^{r_k \times d_k}$ log-linearly using Equation 7:

$$p_{\theta_k,\mathbf{A}_k}(s|\vec{y}_k) = \frac{\exp(\lambda_{s,k}^\top \mathbf{A}_k\vec{y}_k + \alpha_{s,k})}{\sum_{s'} \exp(\lambda_{s',k}^\top \mathbf{A}_k\vec{y}_k + \alpha_{s',k})} \tag{7}$$

  The outputs of this step are $\hat{\mathbf{A}}_k$ and $\hat{\lambda}_k$, the final estimators of $\mathbf{A}_k$ and λ_k after the iterative training [9].
- **Convergence Criterion:** If the MMI score (Equation 3) does not increase, terminate this algorithm. Otherwise, generate $\vec{z}_{k+1} = \hat{\mathbf{A}}_k\vec{y}_k$, increment k by 1, and return to Step 2.

3.2. Initialization of the Log-Linear Training

The log-linear training in Step 3 depends crucially on reasonable initial values of the projection matrix and log-linear weights $\mathbf{A}_k^{(0)}$ and $\lambda_k^{(0)}$. One reason why good initial values are crucial is the fact that the large scale of ASR training data makes it likely that the algorithm will suffer from a slow convergence to a (local) maximum. Therefore, $\mathbf{A}_k$ ($\in \mathbf{R}^{r_k \times d_k}$) and corresponding λ_k are initialized by setting $\mathbf{A}_k^{(0)} = [\mathbf{0}_{r_k \times (d_k - r_k)}|\mathbf{I}_{r_k \times r_k}]$ and $\lambda_k^{(0)} = \hat{\lambda}_{k-1}$. It can be inferred that this initialization guarantees that the MMI score of the new iteration is exactly identical to that of the previous iteration:

$$\lambda_k^{(0)} \left[\mathbf{0}_{r_k \times (d_k - r_k)} \,\vdots\, \mathbf{I}_{r_k \times r_k}\right] \left[\begin{array}{c} vec(\vec{z}_k\vec{z}_k^\top) \\ \hdashline \vec{z}_k \end{array}\right] = \hat{\lambda}_{k-1}\hat{\mathbf{A}}_{k-1}\vec{y}_{k-1}$$

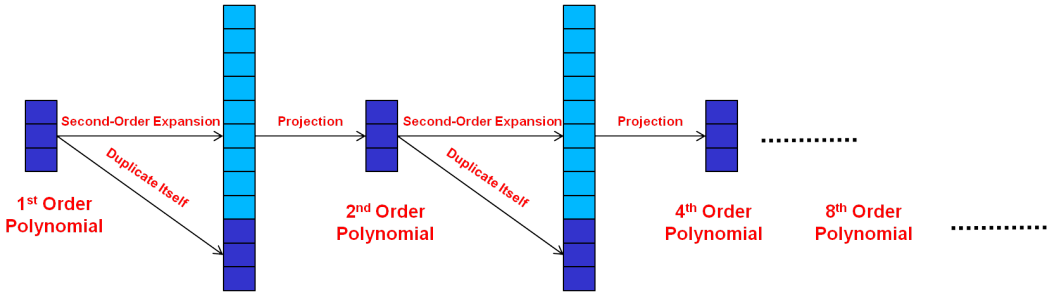

Figure 1: The flowchart of proposed iterative higher-order polynomial acoustic model training framework

Since the iterations continuously absorb more discriminative information from the higher-order polynomials, the low-dimensional reduced vector might not be able to capture such additional information and need to be augmented:

$$\mathbf{A}'^{(0)}_k = \left[\begin{array}{c|c} \mathbf{0}_{r_k \times (d_k - r_k)} & \mathbf{I}_{r_k \times r_k} \\ \hline \mathbf{0}_{(r_{k+1}-r_k)\times(d_k-r_k)} & \mathbf{0}_{(r_{k+1}-r_k)\times r_k} \end{array}\right]$$

Compared with $\mathbf{A}_k$ defined in the previous subsection, the matrix $\mathbf{0}_{(r_{k+1}-r_k)\times d_k}$ is inserted. Accordingly, $\lambda^{(0)}_k$ is also augmented by its copy, which results in $\lambda'^{(0)}_k = [\lambda^{(0)}_k | \lambda^{(0)}_k]$. It can be seen that $\lambda'^{(0)}_k \mathbf{A}'^{(0)}_k = \lambda^{(0)}_k \mathbf{A}^{(0)}_k$. Therefore, the additional rows can be expected to learn the additional information when higher-order polynomials are considered.

4. Experimental Setup

In this paper, a large-vocabulary continuous speech recognition task is performed to show the usefulness of the higher-order polynomial features trained by the proposed framework. The corpus is from the task of European Parliament Plenary Sessions (EPPS), which contains speeches of the European Parliament in British English under clean conditions. The training corpus is 40.8 hours and the evaluation corpus is 3.5 hours. The lexicon contains 54k words and there is a 4-gram language model.

The acoustic representation is composed of 16 MFCC features and 1 voicing feature. Nine consecutive frames are stacked together, and reduced by LDA to a 45 dimensional feature vector. There are 4501 triphone CART-based generalized triphone states.

5. Results and Discussions

5.1. Effectiveness of Using Higher-Order Polynomials

The Word Error Rate (WER) after each step of polynomial expansion and linear projection is shown in Table 1; the results of using first-order and second-order polynomials have already been reported in [9] and [3]. The row "1st, log-LDA" [9] refers to a setup where the feature transformation matrix has also been trained discriminatively The acoustic models with two additional iterations, meaning the usage of 4th and 8th order polynomial features, achieve 1.8% and 2.1% absolute WER reductions over the acoustic model using 2nd-order polynomial features. Even compared with a log-linear model with full covariance information [8], which achieves at WER of 20.8%, the model with 4th-order polynomial features shows superior performance. Thus, invoking higher-order polynomial features is promising, both in terms of WER and computational complexity.

Another important observation from Table 1 is the dependence of WER on the number of output dimensions with polynomial features. The 8th order polynomial features yield a WER of 19.0% if 45 output dimensions are used. If the number of output dimensions is increased to 90, WER decreases to 18.2%. This indicates that for representing the increased information of higher order polynomial features, the dimensionality of the resulting feature space needs to be increased somewhat.

Table 1: *WER of EPPS dev2007: Mixtures vs. Polynomials*

Polynomial Order	**Training Criterion**	**Feature Dim.**	**Num. Para.**	**WER (%)**
1st	ML	45	213k	28.3
1st	MMI	45	213k	25.3
1st, log-LDA [9]	MMI	45	213k	23.5
1st, 2 mixtures	MMI	45	433k	20.2
1st, 4 mixtures	MMI	45	853k	18.4
full 2nd order	MMI	1080	4972k	20.8
2nd [3]	MMI	45	263k	21.1
4th	MMI	45	311k	19.3
8th	MMI	45	359k	19.0
8th	MMI	90	611k	18.2

5.2. Analysis on the Computational Costs

For practical use of high-order polynomial features it is very important to analyze the computational costs when introducing these features in the acoustic models. In Table 1, the results obtained with increasing the number of the mixture densities with first-order polynomial features are also presented. The number of parameters in these models increases linearly with the number of mixtures. From the Table it can be seen that when using the higher-order polynomial models competitive WERs can be obtained with a smaller number of parameters that when increasing the number of mixture densities.

Figure 2 visualizes the relationship between the number of parameters and WER for the systems adopting polynomials and mixtures. It can be seen that the proposed polynomial-based method improves the ASR performance to the same extent as when using mixture densities, at a lower cost in terms

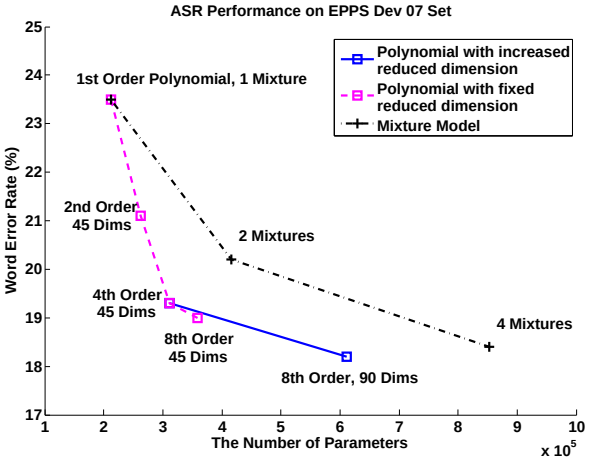

Figure 2: *EPPS dev2007: WER(%) vs. No. of parameters for polynomial features in comparison with mixture densities*

of the number of involved parameters. This suggests that the polynomial features achieve a more compact representation than the mixture models. Moreover, it is interesting to observe that the acoustic models with 8th-order polynomials and 90-dimensional features yield slightly lower WER with much fewer parameters than the mixture models with 4 densities. This confirms that using polynomial features is a very promising approach.

6. Conclusions

In this paper, we presented an iterative log-linear training framework, which can harness crucial information from the higher-order ($\geq$ 3) polynomial feature space. The main highlight is that the framework allows us to train log-linear models in the otherwise prohibitively high-dimensional feature space spanned by higher-order polynomials. By repeating the second-order polynomial expansion n times, 2^n-order polynomial features are embedded into the log-linear models. Following each doubling of the polynomial feature order, the subsequent linear projection limits the dimensionality of the reduced feature space, such that the iterative expansion does not lead to a computationally prohibitive feature space. ASR performance obtained on the EPPS corpus revealed that the use of increasingly higher order polynomials is computationally feasible and improves the log-linear acoustic models. Moreover, conquers the combinatorial complexity of high-order polynomial features by combining successive doubling of the order of polynomial features and application of linear dimension reducing transforms to the resulting features. It leads to moderate improvements in ASR performance at a significantly lower number of parameters, which indicate the potential of the approach.

7. References

[1] H. Huang, L. ten Bosch, B. Cranen, and L. Boves, "Knowledge-based quadratic discriminant analysis for phonetic classification," in *Proc. of ICASSP*, 2012.

[2] Y. Kubo, S. Wiesler, R. Schlueter, H. Ney, S. Watanabe, A. Nakamura, and T. Kobayashi, "Subspace pursuit method for kernel-log-linear models," in *Proc. of ICASSP*, 2011.

[3] M. Tahir, R. Schlueter, and H. Ney, "Log-linear optimization of second-order polynomial features with subsequent dimension reduction for speech recognition," in *Proc. of Interspeech*, 2011.

[4] P. Clarkson and P. Moreno, "On the use of support vector machines for phonetic classification," in *Proc. of ICASSP*, 1999, pp. 585–588.

[5] V. Vapnik, "An overview of statistical learning theory," *IEEE Transaction on Neural Network*, pp. 988 – 999, 1999.

[6] S. Wielser, M. Thom, G. Heigold, R. Schlueter, and H. Ney, "Investigations on features for log-linear acoustic models in continuous speech recognition," in *Proc. of ASRU*, 2009.

[7] S. Wiesler, A. Richard, Y. Kubo, R. Schlueter, and H. Ney, "Feature selection for log-linear acoustic models," in *Proc. of ICASSP*, 2011.

[8] G. Heigold, "A log-linear discriminative modeling framework for speech recognition," Ph.D. dissertation, RWTH Aachen, 2010.

[9] M. Tahir, G. Heigold, C. Plahl, R. Schlueter, and H. Ney, "Log-linear framework for linear feature transformations in speech recognition," in *Proc. of ASRU*, 2009.

[10] T. Deselaers, T. Gass, and H. N. G. Heigold, "Latent log-linear models for handwritten digit classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, pp. 1105 – 1117, 2011.