# PORTING CONCEPTS FROM DNNS BACK TO GMMS

*Kris Demuynck, Fabian Triefenbach*

ELIS/MultimediaLab, Ghent University, Belgium

{Kris.Demuynck,Fabian.Triefenbach}@ugent.be

**ABSTRACT**

Deep neural networks (DNNs) have been shown to outperform Gaussian Mixture Models (GMM) on a variety of speech recognition benchmarks. In this paper we analyze the differences between the DNN and GMM modeling techniques and port the best ideas from the DNN-based modeling to a GMM-based system. By going both deep (multiple layers) and wide (multiple parallel sub-models) and by sharing model parameters, we are able to close the gap between the two modeling techniques on the TIMIT database. Since the 'deep' GMMs retain the maximum-likelihood trained Gaussians as first layer, advanced techniques such as speaker adaptation and model-based noise robustness can be readily incorporated. Regardless of their similarities, the DNNs and the deep GMMs still show a sufficient amount of complementarity to allow effective system combination.

***Index Terms***— speech recognition, Gaussian mixture models, GMM, deep neural networks, DNN, deep structures

## 1. INTRODUCTION

The acoustic model in a speech recognizer links the observed acoustics with the symbolic phonetic representations used in the lexicon. Speech is produced by modulating a limited number of articulators over time. A speech signal thus has both a spectral and a temporal component. Most acoustic models employ a finite state automaton in the form of a hidden Markov model (HMM) to describe the coarse temporal aspects of the signal. The fine temporal details and the spectral properties are handled by the observation density functions attached to the HMM states. To allow the modeling of the fine –within state– temporal structure, the spectral acoustic features (a single frame) are either augmented with time derivatives or are combined in a short window of frames. In many systems, the task of fitting a short window of frames to each state in an HMM is performed with Gaussian mixture models (GMMs).

Recently, deep neural networks (DNNs) have been shown to outperform GMMs on a variety of speech recognition benchmarks [1]. Several properties of DNNs have been accredited for their outstanding performance. First of all, DNNs are acclaimed to be more efficient at modeling high-dimensional data which is componential [2] or has a low intrinsic dimensionality [1]. Furthermore, large DNNs are likely or can even be forced to contain multiple (simple) models in parallel which are averaged to obtain the predictions [3]. This pushes the learning towards generalizing from trends instead of overfitting (memorizing) the training data. Lastly, deep structures seem to be better adept at modeling the complex correlations between long temporal patterns (a window of frames) and the long span symbolic units (context-dependent phone states) used in the HMMs [1, 4].

In the next section, we explore each of the above listed advantages further and we investigate how the same effect can be obtained with a GMM-based approach. The proposed 'deep' GMM setup uses the maximum-likelihood trained GMMs as a first layer. This assures that the resulting recognition system can still rely on the vast array of GMM-based techniques developed in the last decades to cope with complex tasks such as speaker and environment adaptation [5, 6] and model-based noise robustness [7, 8, 9]. As secondary consideration, we try to keep the overhead introduced by the modifications to the GMM-based system low. This assures that, in combination with existing techniques for fast the evaluation of Gaussians [10], the recognizer can still be deployed on commodity hardware.

The deep GMMs are evaluated on the TIMIT benchmark test in section 3. Directions for further improvements are explored in section 4. The papers ends with some conclusions.

## 2. SYSTEM DESIGN

In this section, we look at some important aspects of DNN-based modeling and try to find matching techniques for GMM-based systems.

**Sharing parameters:** If one interprets the inputs to neurons as logarithmic values, the operation performed by a neuron in a DNN can be interpreted as a "product of experts". Such models are efficient at modeling high-dimensional componential data [11]. In a multi-layer neural network, the lower layers create non-linear manifolds. These intermediate representations then form the "experts" used to model the complex output distributions. This sharing ensures that every output is sensitive to a large fraction of the parameters which in turn ensures that each parameter in the model is constrained by a large fraction of the training data.

Mixture models such as GMMs, by default, do not share parameters over "components". A simple and well known technique to achieve some amount of parameter sharing is to split the input features into multiple streams which are treated (more or less) independently. This approach will be further explored in the "going deep" subsection.

The individual mixture components can also be shared (tied) across outputs. One such technique, applied to the GMMs used in speech recognition, was presented in [10]. The approach starts with making output (state) specific Gaussians and then allowing the sharing of those Gaussians between the states. The subsequent training then reduces the amount of sharing (Gaussian tying) by zeroing those weights for which little support is seen in the training data.

**Feature conditioning:** DNNs are in theory insensitive w.r.t. linear transformations of the input space: any linear feature transformation can be counteracted by transforming the linear weights in the first layer with the inverse transform. In practice however, the learning algorithm still requires well conditioned features. The generative pre-training by means of contrastive divergence [11] used in deep

belief networks (DBN) for example thrives on correlations to find more compact and robust internal representation of the signal. This is demonstrated in [2]: when using Mel spectra instead of cepstra (no truncation of the cepstra), the error rate decreases from 22.3% to 20.7%. In the Mel spectrum case, the pre-training will see correlations both along the spectral and the temporal dimensions, resulting in more informative internal representations than when only time correlations are visible (cepstral features).

GMMs require well conditioned features: the feature set must be compact –typically around 40 components– and the individual feature components must be decorrelated. Several techniques have been devised to achieve both goals. Linear discriminant analysis (LDA) is a basic feature reduction technique [12]. A problem with LDA is that it condenses all information concerning the classes to be discerned (means and variances) into two scatter matrices only. This is appropriate if all classes have the same covariance and if the distribution of the class means can be adequately described by a single Gaussian distribution. Heteroscedastic discriminant analysis (HDA) [13, 14] improves upon LDA by taking the individual class covariances into account. The mutual information based discriminant analysis (MIDA) proposed in [15] takes both the individual class means and covariances into account. As was shown in [16], the approximations make by LDA and HDA do hurt the performance when context-independent phone states are used as classes. If a large set of context-dependent phone states are available, LDA becomes a competitive and computationally far less taxing alternative to MIDA.

Once a suitable low-dimensional sub-space has been selected, the features can be decorrelated using least squares [17] or maximum-likelihood techniques [18]. Sub-space selection and decorrelation can be combined in one operation by employing HDA with a single semi-tied covariance matrix.

**Going wide:** Both the optional pre-training with contrastive divergence and the later training with error back propagation will initially focus on the most salient input-to-output relations. In combination with the random initialization of the DNNs, this leads to the parallel discovery of more or less the same solution with minor variations in different parts of the network. Given that DNNs for speech recognition invariably take a window of frames as input, including both static and dynamic features, even makes that the same input information is present multiple times. The overall effect is that DNNs are likely to contain multiple (simple) models in parallel which are averaged to obtain the final predictions. The 'dropout' technique presented in [3] even forces the existence of multiple parallel encodings. This pushes the learning towards generalizing from trends instead of overfitting (memorizing) the training data.

Statistical model combination and multi-stream approaches are well known techniques in speech recognition. See for example [19] for an overview of some recent examples. In our approach, we want to combine the multiple streams (GMMs) into a single acoustic model. This can be accomplished by training multiple GMMs that model the same set of outputs (the context-dependent states used by the HMM). This is similar in concept to what is proposed in [19]. However, the focus in this paper is not on harvesting the complementary information present in different feature sets. We combine GMMs for two main reasons. Firstly, system combination improves the robustness of the system. Compared to a large monolithic GMM, a combination of multiple smaller GMMs has less chance of overfitting the training data. The second reason is the handling high dimensional input data. As was mentioned in the section "feature conditioning", GMMs for speech recognition typically limit the input dimensionality to 40. Feature selection techniques invariably lose some information. Splitting a high dimensional input vector into (more or less) independent streams is the most appropriate way to handle high dimensional (componential) data efficiently with GMMs.

**Going deep:** Instead of coping with the full complexity of the speech signal in a single step, DNNs distribute the classification task over successive layers where each layer handles some of the remaining complexity (undesired variability). Hence, the classification task becomes increasingly more manageable as the signal propagates through the layers [20]. This approach seems to be an excellent fit for the acoustic modeling which must make the complex mapping from long temporal patterns (a window of frames) to the context-dependent phone states in the HMM. To mimic this DNN-property, we propose to add one (or more) extra layers to a GMM-based system as follows. First, the state likelihoods at frame $t$ are converted to posterior probabilities by multiplying the likelihoods with the state priors and normalizing them to sum up to 1.0. The posterior state probabilities over a range of $L=2R+1$ frames are then combined in a log-linear model, i.e. the posteriors $p_t^{(s)}$ for state $s$ on frame $t$ are derived from the posteriors $q_t^{(s')}$ derived from the GMMs as follows:

$$p_t^{(s)} = \frac{1}{Z_t} \exp\left( \sum_{s',\delta=-R\ldots R} \lambda_{s',\delta} \log\left(q_{t+\delta}^{(s')}\right) \right)$$

with $Z_t$ a normalization constant to assure $\sum_s p_t^{(s)} = 1$. Note that we convert the posterior probabilities $p_t^{(s)}$ back to values that behave like conditional state likelihoods by dividing away the state priors before using them in the Viterbi decoding.

This setup is consistent with the gradual, layer-wise, decomposition strategy we assume DNNs use: the GMMs provide posteriors based on a short window of frames; each log-linear layer expands this view with $2R$ frames; the log-linear layers can build upon the intermediate representations made by the lower layer(s).

The connections of the log-linear model can be pruned easily by only retaining those connections that are supported by a high correlation between the target values of $p_t^{(s)}$ (the labeling of the training data) and the observed values of $q_{t+\delta}^{(s')}$. The log-linear model can also be initialized trivially by setting $\lambda_{s'=s,0}$ to 1.0 and all other connections to 0.0.

Log-linear models are not new to the world of speech recognition. They have been introduced before in the form of hidden conditional random fields [21], flat direct models [22], dynamic Bayesian networks [23], and as the soft-max final layers in MLPs. Likewise, the combination of log-linear models to form deep structures has been investigated before [24].

The introduction of the log-linear layer transforms the generative GMM-based model to a discriminative model. Unlike the discriminative training of HMM-GMM based systems which uses the extended Baum-Welch (EBW) algorithm to work out the inter-frame dependencies and hence employs training targets that are better related to the speech recognition accuracy, our current setup –similar to standard DNNs– still treats every frame independently. See section 4 for pointers of how EBW can be integrated in the setup.

Given that the resulting deep GMMs employ (frame-by-frame) discriminative training, one could opt to update the Gaussian means, covariance and mixture weights as well. For this work, we deliberately choose to not update the GMM parameters: by keeping the generative character of the GMMs, maximum likelihood (ML) adaptation techniques such as (f)MLLR [5] and other techniques that assume a generative model are expected to be still applicable.

| System | #layers | N-gram 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| baseline GMM systems | | | | | |
| base0 | 0 | 26.29 | 24.01 | 22.94 | 22.17 |
| base1 | 0 | 25.94 | 23.86 | 22.62 | 22.24 |
| base2 | 0 | 25.54 | 24.08 | 22.87 | 22.17 |
| base3 | 0 | 25.77 | 24.31 | 23.05 | 22.47 |
| base4 | 0 | 25.34 | 23.76 | 22.73 | 22.47 |
| single deep GMM systems | | | | | |
| deep0 | 1 | 23.82 | 22.66 | 21.30 | 20.84 |
| deep1 | 1 | 23.55 | 22.02 | 21.07 | 20.77 |
| deep2 | 1 | 23.02 | 22.50 | 21.55 | 21.06 |
| deep3 | 1 | 23.40 | 22.51 | 21.86 | 21.06 |
| deep4 | 1 | 23.51 | 22.54 | 21.59 | 20.78 |
| deep+wide GMM systems | | | | | |
| x1: comb. b0...4 | 0 | 24.14 | 22.53 | 20.93 | 20.29 |
| x2: x1→deep | 1 | 22.24 | 20.70 | 19.87 | 19.76 |
| x3: x2→deep | 2 | 22.28 | 20.93 | 20.28 | 19.81 |
| y1: comb. d0...4 | 1 | 21.37 | 20.78 | 19.45 | 19.36 |
| y2: y1→deep | 2 | 21.66 | 20.51 | 19.57 | 19.34 |
| z1: d0+d1 | 1 | 22.26 | 21.27 | 20.43 | 19.98 |
| z2: d0+d4 | 1 | 22.17 | 20.99 | 20.09 | 19.43 |
| fMLLR, MLP, combination | | | | | |
| f1: b1+fMLLR | 0 | 24.71 | 22.42 | 21.57 | 21.31 |
| f2: d1+fMLLR | 1 | 21.85 | 20.85 | 20.10 | 20.14 |
| f3: y2+fMLLR | 2 | 20.16 | 19.66 | 18.82 | 18.83 |
| m1: MLP | 2 | 23.18 | 22.65 | 22.11 | 22.06 |
| m2: y2+MLP | 2 | 20.85 | 20.25 | 19.12 | 18.71 |

**Table 1**. Phone error rates on the TIMIT core test set

## 3. EXPERIMENTAL VALIDATION

The deep GMMs were evaluated on the TIMIT corpus [25], a database specifically designed for phone recognition experiments. The absence of higher level linguistic resources such as lexicon and language model in combination with its small size make TIMIT an excellent choice for the development and evaluation of new approaches to acoustic modeling.

TIMIT contains recordings of 630 native American speakers, 438 males and 192 females. The data of 462 speakers were used to train the acoustic models. The test set contains 24 speakers (the so called core test set). The remaining 144 speakers were set aside as development data and were used to tune hyper parameters such as the weight of the phone N-gram. Each speaker reads 8 phonetically rich sentences and 2 calibration sentences (the SA sentences). The SA sentences were not used. TIMIT provides detailed manually verified acoustic-phonetic transcriptions for all sentences using an alphabet containing 61 symbols. The 61 symbols were mapped back to 51 symbols for the acoustic modeling and, as was proposed in [26], 39 symbols for the evaluation. The phone recognizer uses a Viterbi search in combination with an N-gram derived from the labels of the training utterances. Given that deep systems can learn phonotactic constraints by itself when being fed with a large window of frames [4, 27] while HMM+GMM systems commend a more strict separation between acoustics modeling (GMMs) and phonotactics (phone N-gram), we compare the systems over a range of phone N-grams.

SPRAAK [28] was used to create the baseline generative HMM+GMM systems. The acoustic features consist of 22 vocal tract length normalized (VTLN) and mean normalized Mel Spectra. The VTLN employs two GMMs to distinguish between male and female speech on a sentence by sentence base [29]. The seed HMM system $<base0>$ was trained using the standard recipe in SPRAAK. The 22 Mel Spectra $\phi$ are augmented with their first and second order time derivatives $\Delta, \Delta^2$. This vector is reduced to 39 dimensions by means of MIDA [15] using the context-independent phone states as classes, after which the features are decorrelated [17]. The resulting HMM system uses a shared pool of 5710 Gaussians to model the observations in 589 context-dependent tied triphone states, using GMMs containing 135 Gaussians on average per state. This seed model provides the phonetic descision tree and hence the context-dependent phone states and a corresponding labeling of all training data.

Four derivative acoustic models $<base1...4>$ were created using the techniques described in section 2. These additional systems differ in their input features only. The following feature combinations were used to represent frame $t$:

- $<b0,b1>$: $[\phi_t, \Delta_t, \Delta_t^2]$
- $<b2>$: $[\phi_t, \Delta_{t-2...t+2}]$
- $<b3>$: $[\phi_t, \Delta_{t-2,t,t+2}]$
- $<b4>$: $[\phi_t, \Delta_{t-3,t-1,t+1,t+3}]$

LDA with the context-dependent states as classes was used to reduce each of these feature vectors to 39 features. After feature decorrelation, state-specific Gaussians were made. The number of Gaussians per state was set proportional to the amount of data, with a maximum value to prevent an excessive amount of silence Gaussians. The total number of Gaussians was set to 5833, a value close to that of the seed system. The Gaussians were allowed to be shared across the states using the approach described in section 2.

Systems $<b0>$ and $<b1>$ start from the same features, but employ different techniques for feature dimension reduction (MIDA on context-independent states versus LDA on context-dependent states) and for the initialization and tying of the Gaussians (on context-independent states versus context-dependent states). Hence, when looking at the combination of systems based on $<b0>$ and $<b1>$ solely, we look at how much one can gain be combining two different encodings of the exact same input space. Systems $<b2>$ and $<b3>$+$<b4>$ on the other hand expand the scope of the input space by adding the $\Delta$ features from surrounding frames. Since the $\Delta$ features are computed using a 5 frame window themselves, the input of these systems span up to a 110ms of data. By adding only the $\Delta$ features and removing the $\Delta^2$ features from the central frame, we prevent duplication of data which would have been problematic for both the LDA-dimension reduction and the diagonal covariances in the GMMs. As discussed in section 4 and shown in [19], system combination provides better results when the contributing systems, starting from the input features, are complementary. However, in the scope of this work, we preferred a fair comparison with DNNs and hence assured that all four feature combinations are a sub-set of the features typically used by DNNs.

Once all derivative GMMs were created, a consensus segmentation was obtained by combining all 5 systems. Two additional Viterbi re-estimation iterations were performed on each system using this consensus segmentation. All system combinations employ a "product of expert" combination rule, i.e. we make a weighted average of the log likelihoods. The weights are frame independent and add up to 1.0. Systems $<b3>$ and $<b4>$ should be regarded as the two components from a larger system which takes a window of 110ms of data as input. Therefore, systems $<b3>$ and $<b4>$ are assigned halve the weight of the others when combining systems; all other systems receive an equal weight.

Deep variants of these systems (<*deep0...4*>) and of the combined system were trained using a basic MLP back-propagation algorithm (no momentum term). The input of the additional log-linear (soft-max) layer consist of $L = 11$ frames of log state posteriors. Overfitting was avoided by reducing the number of connections to 250 per output node (context-dependent states); see section 2 for more details. The order of the frames was randomized. The batch size was set to 100 frames. To obtain convergence, the learning rate was gradually decreased.

All results are listed in table 1. Going 'wide' (<*b0...4*> vs. <*x1*> and <*d0...4*> vs. <*y1*>) gains approximately 1.5% absolute. Going 'deep' (<*b0...4*> vs. <*d0...4*> and <*x1*> vs. <*x2*>) gains approximately 1.0% absolute for the 3-gram case and is largely complementary to going 'wide'. Given that the 'deep' variants automatically consider a larger window of frames and hence can learn more phonotactics, the gains are more pronounced for the lower order N-grams. Comparing <*x2*> with <*y1*> shows that keeping the system wide for at least one log-linear layer gives somewhat better results, at the cost of more parameters and a slower acoustic model. Going 'deeper' (systems <*x3*> and <*y2*>) show little to no improvements. These 3-layer systems integrate information over 310ms: 110ms of data given as input and an additional 50ms to the left and right with each log-linear layer. A more gradual increase of the 'window size' over the layers, as done in [4] may give better results overall. Also note that the first layer, the GMMs, correspond to two layers in a DNN, namely the Gaussian-Bernoulli restricted Boltzmann machine (RMB) used to handle the real-valued input data and the subsequent binary-valued RMB which makes the linear combination and the soft-max.

Comparing <*d0*>, <*y1*>, <*z1*>, and <*z2*> shows that most of the gain obtained by going 'wide' can already be achieved with two systems, even if they use an identical prepreprocessing (<*z1*>). The combination of the two most complementary deep systems as done in <*z2*> even get very close to the full 'width' combinations (<*y1*>, <*y2*>).

Given that the deep GMMs retain the maximum-likelihood trained Gaussians as first layer, model-based adaptation techniques such as fMLLR [5] for speaker adaptation should still work. This is investigated in the <*f1...3*> setups. The target labels for the fMLLR adaptation were obtained with a first recognition pass on all 8 sentences for each speaker. Given the low amount of adaptation data and the use of full transformation matrices (LDA-features cannot be readily separated in three more or less independent streams), we opted to use all data instead of discarding the phone segments with the lowest confidence scores. Both for the normal GMMs as for the deep and wide GMMs, applying fMLLR results in tangible improvements (on top of the VTLN).

A final experiment verifies whether the combination with a deep MLP, using the same features, is still beneficial. The deep MLP <*m1*> was designed identical to the <*y2*> system: two MLPs with a single hidden layer containing 2500 nodes and taking $[\phi_t, \Delta_t, \Delta_t^2]$ and $[\phi_t, \Delta_{t-5\ldots t+5}]$ as input features were converted to deep structures by adding a log-linear layer (cf. section 2, $L = 11$) to each of them. After combining the two scores, an extra log-linear ($L = 11$) layer was added. The fact that the combination still improves the results substantially indicates that, despite their similarities, deep GMMs and deep MLPs do intrinsically behave differently.

The results in table 1 can also be compared to the best published results. A result of 19.7% is reported in [3] for a large DNN with 4 hidden layers containing 4000 nodes each and trained using 'dropout'. Since this system does not use any information about speaker identity, it should be compared with <*y2*> and <*m2*>. Using discriminative features, speaker adaptation and the combination with a discriminatively trained HMM [27] lowers the error rate to 19.3% when using a 3-gram. This result should be compared with <*f3*> or <*m2*>. In both cases, the deep GMMs return comparable or better results. Furthermore, the <*y2*> and <*m3*> systems contain only 3.4mio free parameters, which is more than an order of magnitude less than the DNN systems they are compared with.

## 4. FUTURE WORK

Although reasonable design choices were made during the design of the deep GMM systems, more advanced options are known for most of the stages.

First of all, the complementary of the parallel models (streams) in the first layer(s) can be increased. A straightforward technique to attain this is to start with different acoustic features [19]. Given the use of a log-linear model for combining the streams, the concept of boosting as presented in [30] can be applied as well. Boosting can be accomplished for example by modifying the linear discriminant analysis criterion when adding a second or later streams [31] so that more weight is given to those features dimensions which carry complementary information.

Another simple modification would be the use of speaker adaptive training [32] to increase the efficiency of the speaker adaptation. Next to the Gaussian mean and variances, the GMM weights can be adapted as well [33].

Despite the fact that the acoustic models must relate sequences of frames (short windows of frames) to context-dependent phone states, they were still trained with a frame-based cross-entropy error criterion in this work. As was show in [34, 35, 36], substantially better results can be obtained when using appropriate sequence-based criteria such as minimum phone error or maximum mutual information. The resulting extended Baum-Welch (EBW) algorithm takes into account the inter-frame dependencies and hence provides training targets that are more directly related to the speech recognition accuracy. EBW can be readily applied to log-linear models and neural networks, either in the form of on-line training [37] or batch training [38].

In the current setup, overfitting is prevented by making the connection weights sparse and by having fixed model mixing weights. As was proposed in [19], one may make the model mixing weights state dependent, i.e. instead of combing the streams and then stacking $L$ frames of log state posteriors, one may keep the individual state posteriors of the $S$ streams and let the training algorithm devise a good weighting. This increases the number of trainable parameters $S$ fold and hence will require some form of regularization to prevent overfitting. In a Bayesian framework, regularization corresponds to imposing certain prior distributions on model parameters. Since we can infer reasonable initial values for the parameters, it must also be possible to devise sensible parameter specific regularization costs. Another strategy would be to inject noise into the training. Noise, for example in the form of 'dropout', acts as a powerful regularization by preventing complex co-adaptations [3].

Next to improving the systems, the approach should be evaluated on different data and more tasks, with the extension to large vocabulary continuous speech recognition being the first objective. Similar to DNNs, the deep GMMs are designed to effectively correlate the acoustic information present in relatively long (200ms) windows of speech with the phonetic sub-units used by the decoder. DNNs seem to be able to learn this complex relation even for very challenging conditions where the observed acoustics may deviate substantially from the canonical phone transcriptions found in the lexicon such as

spontaneous speech and speech by non-natives [39, 1]. Given the common design principles, we expect that deep GMMs will exhibit the same potential.

Noise robustness is another interesting aspect. Since the deep GMMs retain the maximum-likelihood trained Gaussians as first layer, model-based noise robust techniques such as VTS [7], SPLICE [8] or joint uncertainty decoding [9] should still be applicable. Furthermore, the use of a relatively long window of frames in combination with training schemes that employ some form of noise injection [3] may push the system to learn some from of island-parsing [40, 41] automatically.

## 5. CONCLUSIONS

The deep Gaussians mixture models proposed in this paper borrow several ideas from deep neural networks: an efficient use of the parameters by sharing intermediate representations (tied Gaussians), having multiple smaller systems in parallel to obtain less noisy predictions, and the layer-wise decomposition of a complex problem into smaller sub-tasks.

On other aspects, the two approaches could be regarded as antipodes. A typical DNN for TIMIT consists of 3 or more fully interconnected hidden layers containing 4000 nodes each. The layers are homogeneous, except for the use of a Gaussian-Bernoulli restricted Boltzmann machine (RMB) instead of a binary RBM as first layer. A random initialization is followed by a potent (and demanding) learning phase consisting of generative pre-training and discriminative training. The end result of this is a large homogeneous structure which relies on an effective training scheme to implicitly divide the task over the layers and nodes (sharing representations, modeling spectral versus temporal aspects, keeping multiple parallel sub-models, etc.). As such, DNNs are exceptionally well suited as a generic black-box technique. The proposed deep GMM scheme on the other hand is more a task-specific technique which combines various compact representations (tied Gaussians, sparse connection weights) over multiple heterogeneous layers. The role of the layers (focus on spectral versus temporal aspects) is assigned manually. Given the specific role of each layer, simple but effective procedures can be devised to initialize the parameters in each layer, reducing the chance of getting stuck in a bad local minimum.

Black-box methods such as DNNs are appealing since they require little task-specific expert knowledge and obviate the need for tuning the layers and interconnections or the conditioning of the input features. A well-tuned task-specific architecture on the other hand requires more work to set up but can be expected to perform better or at least offer a more compact solution. The degrees of freedom in designing and interconnecting each layer in the proposed architecture also make it easier to incorporate and/or use expert knowledge such as speaker adaptation and phonotactic constraints into the system. This paper shows, at least for the TIMIT task, how such a well-tuned task-specific architecture can be built using existing techniques and expertise. The resulting system runs much faster than real-time on commodity hardware (no GPUs needed) and yet performs as well as the best DNN systems published. Moreover, as was discussed in section 4, better design choices can be devised for many of the stages, leaving ample opportunities open to further improve the setup.

## 6. REFERENCES


[1] G. Hinton, L. Deng, D. Yu, G.E. Dahl, A-R. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T.N. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[2] A.-R. Mohamed, G.E. Dahl, and G. Hinton, "Acoustic modeling using deep belief networks," *IEEE Trans. on ASLP*, vol. 20, no. 1, pp. 14–22, 2012.

[3] G.E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *CoRR*, p. abs/1207.0580, 2012.

[4] F. Triefenbach, A. Jalalvand, K. Demuynck, and J-P. Martens, "Acoustic modeling with hierarchical reservoirs," *IEEE Trans. on ASLP*, 2013, (accepted).

[5] M.J.F. Gales, "Maximum likelihood linear transformations for HMM-based speech recognition," *Comp. Speech and Lang.*, vol. 12, pp. 75–98, 1998.

[6] R. Kuhn, J.-C. Junqua, P. Nguyen, and N. Niedzielski, "Rapid speaker adaptation in eigenvoice space," *IEEE Trans. on SAP*, vol. 8, no. 6, pp. 695–707, 2000.

[7] A. Acero, L. Deng, T. Kristjansson, and J. Zhang, "HMM adaptation using vector Taylor series for noisy speech recognition," in *Proc. ICSLP*, 2000, pp. 229–232.

[8] J. Droppo, L. Deng, and A. Acero, "Evaluation of the SPLICE algorithm on the Aurora2 database," in *Proc. EUROSPEECH*, 2001, pp. 217–220.

[9] H. Liao and M.J.F Gales, "Adaptive training with joint uncertainty decoding for robust recognition of noisy data," in *Proc. ICASSP*, 2007, pp. 389–392.

[10] K. Demuynck, J. Duchateau, and D. Van Compernolle, "Reduced semi-continuous models for large vocabulary continuous speech recognition in Dutch," in *Proc. ICSLP*, 1996, vol. IV, pp. 2289–2292.

[11] G.E. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural Computation*, vol. 14, pp. 1771–1800, 2002.

[12] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*, pp. 114–121, John Wiley & Sons, 1973.

[13] G. Schukat-Talamazzini, J. Hornegger, and H. Niemann, "Optimal linear feature transformations for semi-continuous hidden Markov models," in *Proc. ICASSP*, 1995, vol. I, pp. 369–372.

[14] N. Kumar, *Investigation of Silicon-Auditory Models and Generalization of LDA for Improved Speech Recognition*, Ph.D. thesis, John Hopkins Univ., 1997.

[15] K. Demuynck, J. Duchateau, and D. Van Compernolle, "Optimal feature sub-space selection based on discriminant analysis," in *Proc. EUROSPEECH*, 1999, vol. III, pp. 1311–1314.

[16] J. Duchateau, K. Demuynck, D. Van Compernolle, and P. Wambacq, "Class definition in discriminant feature analysis," in *Proc. EUROSPEECH*, 2001, vol. III, pp. 1621–1624.

[17] K. Demuynck, J. Duchateau, D. Van Compernolle, and P. Wambacq, "Improved feature decorrelation for HMM-based speech recognition," in *Proc. ICSLP*, 1998, vol. VII, pp. 2907–2910.

[18] M.J.F. Gales, "Semi-tied covariance matrices," in *Proc. ICASSP*, 1998, vol. II, pp. 657–660.

[19] X. Cui, J. Xue, B. Xiang, and B. Zhou, "A study of bootstrapping with multiple acoustic features for improved automatic speech recognition," in *Proc. INTERSPEECH*, 2009, pp. 240–243.

[20] D. Yu, M.L. Seltzer, J. Li, J.-T. Huang, and F. Seide, "Feature learning in deep neural networks – studies on speech recognition tasks," *CoRR*, vol. abs/1301.3605, 2013, http://arxiv.org/abs/1301.3605.

[21] Y-h. Sung and D. Jurafsky, "Hidden conditional random fields for phone recognition," in *Proc. ASRU*, 2009, pp. 107–112.

[22] G. Heigold, G. Zweig, X. Li, and P. Nguyen, "A flat direct model for speech recognition," in *Proc. ICASSP*, 2009, pp. 3861–3864.

[23] G. Zweig and S. Russell, "Speech recognition with dynamic Bayesian networks," Tech. Rep., Microsoft, 1998.

[24] D. Yu, S. Wang, and L. Deng, "Sequential labeling using deep-structured conditional random fields," *IEEE Journal of Selected Topics in Signal Processing*, vol. 4, no. 6, pp. 965–973, 2010.

[25] J. Garofolo, L. Lamel, W. Fisher, J. Fiscus, D. Pallett, and N. Dahlgren, "The DARPA TIMIT acoustic-phonetic continuous speech corpus CD-rom," Tech. report, NIST, 1993.

[26] K. Lee and H.-W. Hon, "Speaker-independent phone recognition using hidden Markov models," *IEEE Trans. on ASSP*, vol. 37, pp. 1641–1648, 1989.

[27] A.R. Mohamed, T.N. Sainath, G. Dahl, B. Ramabhadran, G.E. Hinton, and M.A. Picheny, "Deep belief networks using discriminative features for phone recognition," in *Proc. ICASSP*, 2011, pp. 5060–5063.

[28] K. Demuynck, J. Roelens, D. Van Compernolle, and P. Wambacq, "SPRAAK: An open source speech recognition and automatic annotation kit," in *Proc. INTERSPEECH*, Sept. 2008, p. 495.

[29] J. Duchateau, M. Wigham, K. Demuynck, and H. Van hamme, "A flexible recogniser architecture in a reading tutor for children," in *Proc. ITRW on Speech Recognition and Intrinsic Variation*, 2006, pp. 59–64.

[30] Y. Freund and R.E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of computer and system sciences*, vol. 55, no. 1, pp. 119–139, 1997.

[31] J. Lu, K.N. Plataniotis, and A.N. Venetsanopoulos, "Boosting linear discriminant analysis for face recognition," in *Proc. ICIP*, 2003, vol. 1, pp. 657–660.

[32] T. Anastasakos, J. McDonough, and J. Makhoul, "Speaker adaptive training: a maximum likelihood approach to speaker normalization," in *Proc. ICASSP*, 1997, vol. 2, pp. 1043–1046.

[33] X. Zhang, K. Demuynck, and H. Van hamme, "latent variable speaker adaptation of gaussian mixture weights and means," in *Proc. ICASSP*, 2012, pp. 4349–4352.

[34] B. Merialdo, "Phonetic recognition using hidden Markov models and maximum mutual information training," in *Proc. ICASSP*, 1988, vol. 1, pp. 111–114.

[35] D. Povey and P.C. Woodland, "Improved discriminative training techniques for large vocabulary continuous speech recognition," in *Proc. ICASSP*, 2001, vol. 1, pp. 45–48.

[36] D. Povey and P.C. Woodland, "Minimum phone error and I-smoothing for improved discriminative training," in *Proc. ICASSP*, 2002, vol. 1, pp. 105–108.

[37] B. Kingsbury, "lattice-based optimization of sequence classification criteria for neural-network acoustic modeling," in *Proc. ICASSP*, 2009, pp. 3761–3764.

[38] D. Kanevsky, D. Nahamoo, T.N. Sainath, B. Ramabhadran, and P.A. Olsen, "A-functions: a generalization of extended Baum-Welch transformations to convex optimization," in *Proc. ICASSP*, 2011, pp. 5164–5167.

[39] F. Seide, G. Li, and D. Yu, "Conversational speech transcription using context-dependent deep neural networks," in *Proc. INTERSPEECH*, 2011, pp. 437–440.

[40] L.D. Erman, F. Hayes-Roth, V.R. Lesser, and D.R. Reddy, "The HEARSAY-II speech understanding system: Integrating knowledge to resolve uncertainty," *Computing Surveys*, vol. 12, pp. 213–253, 1980.

[41] R. Kumaran, J. Bilmes, and K. Kirchhoff, "Attention shift decoding for conversational speech recognition," in *Proc. INTERSPEECH*, 2007, pp. 1493–1496.