Literature Study of Deep Learning Approaches In Speech Recognition

Authors: Nizar Gandy Assaf Layouss ngal@kth.se
Javier López Alberca javierla@kth.se

13/05/2013

Abstract

The purpose of this project is to make a thorough exploration of a new computational paradigm referred to as *Deep Learning* applied to Speech Recognition systems. Results in late publications have shown that Automatic Speech Recognisers based on Hidden Markov Model with deep architectures have outperformed any other traditional recognisers based on Hidden Markov Model and Gaussian Mixtures on the TIMIT database. For this reason we believe that it's worth exploring the application of deep learning techniques in the field of speech recognition and present its use in the state of the art speech recognisers.

As an introduction, this project will explore the traditional speech recognisers based on Hidden Markov Models and Gaussian Mixtures putting in evidence some of its weaknesses. In the latter sections a general introduction to deep learning algorithms and architectures will be carried on with emphasis on those used in the speech recognition field continuing to the main sections of this project where we'll explore thoroughly how these architectures are implemented and combined with Hidden Markov Models in the speech recognition field.

Contents

1 Introduction

Over the last five decades immense efforts has gone into studying technological approaches for speech recognition systems and still a general and robust solution for the problem has not been found yet. As early as 1950s, researchers built simple recognizers with credible performance for restricted tasks (such as isolated digits spoken by single speaker) but unfortunately the techniques used in these systems were not sufficient to solve the general problem. The difficulties of this problem can be described in terms of a number of characterizations of the task, including:

1. *Intra- and inter-speaker variabilities:* Is the system *speaker dependent*(i.e., optimized for a single talker), or *speaker independent*(can recognize anyone's voice)?

2. Is the system able to recognize *isolated or continuous speech?* With *isolated word recognition* system, the talker is required to say words with short pauses in between. This is the simplest case, since word boundaries are detected fairly easily and words are not strongly coarticulated. The other case is *Continuous Speech Recognition* where boundaries between words are more difficult to detect and words are strongly coarticulated.

3. *Vocabulary size and confusability:* Is the system able to recognize a vocabulary of only a few words, or can it handle large vocabularies of thousands of words? What is the potential confusability between words? In general, it is difficult to get good recognition results with a large vocabulary, and the computation time can also be an issue in this case.

4. Does the system work in *adverse conditions?* Several variables that can alter the performance of ASR systems have been identified:

   - environmental noise(e.g., in car, cockpit or factory floor).
   - distorted acoustic and speech correlated noise (room acoustics, nonlinear distortions).
   - different microphones and distance from microphone.
   - altered speaking manner (e.g., differing speaking rate, speaker stress, breath, pitch...)
   - some combination of the above (most cases).

Most of these difficulties can be summarized fairly simple: variability of the speech acoustic and variation of additive noise. However, we instinctively expect a high level of recognition performance, much as would be achieved by a human, and have very little interest in a recognizer that makes frequent mistakes. For these reasons, speech recognition must achieve a very high level of performance to be of general interest as a man-machine interface.

Although modern systems do not achieve yet recognition performance achieved by human, late published systems do fairy well. Results In [1, 2] on the TIMIT [3] database have shown a potential way to improve recognition systems by using deep architectures [4] interfaced with Hidden Markov Models to approximate the temporal variability and in this paper we'll explore the new paradigm applied in recognition systems.

2 Statistical Approach to ASR

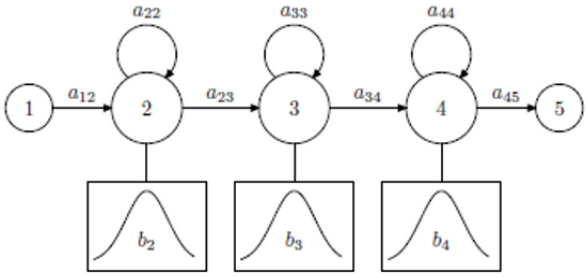
One of the most common approaches to solve the problem of *Automated Speech Recognition* is using statistical framework for inference given a certain utterance. The idea is that given a set of measurements, x, what's the probability of it belonging to a class ω_k also known as *a posteriori probability* $p(\omega_k|x)$. By using the Bayes Rule it's possible to calculate it as follows :

$$p(\omega_k|x) = \frac{p(x|\omega_k)p(\omega_k)}{p(x)} \tag{1}$$

Where x is the measurement (acoustic feature) taken from the utterance signal and ω_k is the set of classes (words or phonemes).

The measurement x typically is a *Mel Frequency Cepstral Coefficients (MFCC)* vector that aims to capture the discriminative information of the utterance signal and discard the less important information to ease in the computations. Other type

Figure 1: Continuous Density Hidden Markov Model



of feature extraction exist such as *Linear Predictive Coding (LPC)*, *Perceptual Linear Prediction (PLP)*, first order derivative, second order derivative etc...

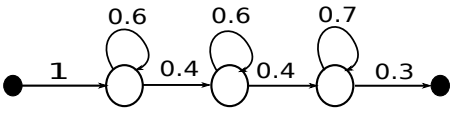
As equation (1) shows, the a posteriori probability is split in the conditional probability $p(x|\omega_k)$ where it represents the probability of the utterance given the *Acoustic Model* and the prior probability $p(\omega_k)$ where it's the probability of the class (phoneme or a word) aa priori from our knowledge of the language (known as the *Language Model*). In this paper the focus will be on the application of Deep Learning techniques to the acoustic model and for the rest of the document we'll be dealing with different forms of building acoustic models that better approximates the nature of human speech utterances.

3 HMM/GMM Aproach

Given the temporal variability in speech and its sequential nature, the most efficient approach developed so far to model it is by a Hidden Markov Model [5]. For every unit of speech (either a word or a phoneme) a hidden markov model is built. The continuous nature of speech gives rise to model it efficiently by a *Continuous Density Hidden Markov Model (CDHMM)*. Basically a CDHMM is a Hidden Markov Model where the emission probability $p(x|q_i)$ is a continuous density function as shown in Figure 1. A special case of this model is the approximation of $p(x|q_i)$ by *Gaussian Mixture Model (GMM)*.

Considering a phoneme as the basic unit of

Figure 2: Three State HMM Phoneme Model



speech, each phoneme is associated with a hidden markov model with a predefined topology that generally is tri-state phoneme topology although other topologies exist like one state representing a phoneme or more than three states.

The general process of the phone recognition then is summarized as follows:

1. *Feature extraction*: This process converts the speech signal into a sequence of acoustic vectors $\mathbb{X} = (x_1, x_2, ..., x_N)$.

2. *Training phoneme models:* A slightly modified version of Baum-Welch algorithm is used to estimate the parameters of the HMM/GMM phoneme models.

3. *Recognition phase:* Given a set of trained phoneme models, recognition is carried on by using Viterbi decoding algorithm with backpointers to recover the path.

With viterbi decoding it's possible to recover the sequence of models that gave rise to the maximum likelihood probability $p(x|\omega_k)$ where ω_k here is the phoneme model.

Although HMM's are the most efficient approach developed so far for modeling the temporal variability in speech, they suffer from several drawbacks. Several assumptions are made by using HMM's :

- Observation independence assumption: acoustic features are not correlated and conditional independence exist given a state at time t.
- First order markov model: the probability of transition to state q_i at time t only depends on the state q_j at time $t-1$ and is conditionally independent from the history of the chain.
- Probability of transition are assumed stationary (assumed to be time invariant).
- Poor discrimination due to the training algorithm, which maximizes likelihoods instead of posterior probabilities.

The drawbacks related to modeling the emission probability $p(x|q_i)$ with GMM's:

- Modeling the variability and non-linearity of speech acoustic features could end up estimating a very high number of parameters by assuming Gaussian mixtures.
- Acoustic feature vectors are assumed to be uncorrelated to reduce the parameters to be estimated in the covariance matrix of the gaussian to only the diagonal.

The assumption of uncorrelated acoustic features in the HMM/GMM is the main drawback and that's why other approaches to estimate the emission probabilities were proposed such as the hybrid model of interfacing a multi-layer perceptron (MLP) with an HMM as discussed in the next section.

4 Hybrid HMM/MLP Approach

Given the difficulties presented in section 3 of modeling the emision probabilities with GMM's, an alternative was proposed by using Multi-Layer Perceptron to model the emission probability in the HMM [6] .

Attempts in the past of using *Artificial Neural Networks (ANN)* to model the whole process of speech recognition ended in failure given that there is a main lack in the considering neural networks for the whole task: they we not tailored for time sequential inputs. For that reason, they are interfaced with an HMM to model the stationary emission probability given a state.

Advantages of using a MLP to approximate emission probabilities:

1. It provides discriminative learning.
2. It can approximate in theory any kind of nonlinear functions of the input.
3. There is no need for strong assumptions about the statistical distributions of the input features.
4. There is no need to assume uncorrelated acoustic fetures. Additionally, by presenting several sequential frames in the input it's possible to model the contextual information.

Thus considering ANN's for modeling emission probabilities constitute a promising approach. The training algorithm used is the Backpropagation –ref with the error criteria as the cross-entropy function. Labels to complete the training has to be privided, so acoustic features presented in the neural network has to be correspondingly labeled the the speech unit class they belong to. Thus, segmentation of the utterances to identify and label the speech units is necessary. Embedding MLP training in a Viterbi algorithm as presented in [6] shows that an iterative process of training with an initial segmentation is possible and eventually improve the initial segmentation. Here are provided the steps for the Viterbi alignment:

1. Starting only with the phonetic transcription of the training and cross-validation sets, these two sets of sentences are linearly segmented, respectively providing the MLP output targets for the training set as well as for the cross-validation set.
2. The MLP can then be trained (using cross-validation), which provides new weights and, consequently, new emission probabilities for the Viterbi matching.
3. Using this newly trained MLP,Viterbi matching is performed on the training and cross-validation sets, providing us with new segmentations and, consequently, with new output targets for MLP training and cross-validation.
4. This process is iterated until the score (product over all the optimal path probabilities) on the cross-validation set (and not on the training set) begins to decrease. Thus, two cross-validations take place in this process: one for the MLP itself, and one for the Viterbi matching.

The number of hidden units in the output layer is as the number of states of the HMM, thus if we consider phonemes as speech units and n phonemes each one with one state then the output layer would have n units with a softmax squashing function to approximate a probability distribution over the set of output units. If a three state phoneme model

is considered, then the output layer would have $3n$ units as output.

Theoritical and experimental results have shown that the MLP output values may be viewed as estimates of *Maximum a Posteriori (MAP)* probabilities [6] in a sense that it outputs $p(\omega_k|x)$ (or $p(q_i|x)$ as every state is associated with a phoneme). But as mentioned before, the main use of the MLP is to estimate the likelihood $p(x|q_i)$ (emission probability). A way to overcome this problem is by using Bayes Theorem and work with the *scaled likelihood* in the viterbi training algorithm:

$$\frac{p(x|q_i)}{p(x)} = \frac{p(q_i|x)}{p(q_i)} \tag{2}$$

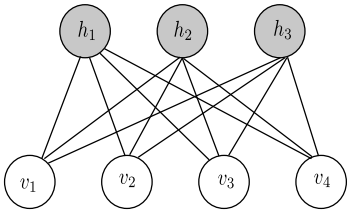
where $p(q_i)$ is the phoneme prior probability estimated from the training set and $p(q_i|x)$. Working with scaled likelihood won't hurt the recognition process as all viterbi decoding probabilities will be devided by the same constant $p(x)$. To summarize, the general algorithm for training a Hybrid HMM/MLP :

1. Extract the acoustic features from with speech signal with an initial speech unit segmentation
2. Train the MLP using the acoustic features vector as the input (one by one or concatenating several frames is possible to model context dependency of a central frame.)
3. Train the HMM using Viterbi algorithm and with the scaled likelihood emission probabilities that the MLP outputs.
4. Iterate over the whole process.

5 Deep Neural Networks

Inspired by the architectural depth of the brain, neural network researchers had wanted for decades to train deep multi-layer neural networks, but no successful attempts were reported before 2006 . With deep achitectures we refer to architectures having more than one hidden layer as is the case of MLP. It was not until Hinton et. al. at university of Toronto introduced *Deep Belief Networks (DBN)* [] exploiting an unsupervised learning algorithm for each layer, a Restricted Boltzmann Machine (RBM). Since then a whole new paradigm od training efficiently emerged called *Deep Learning*.

Figure 3: Restricted Boltzmann Machine.



5.1 Restricted Boltzmann Machine

The *Restricted Boltzmann Machine (RBM)* is a type of an *Energy Based Model* represented as an undirected graphical model that basically consists of two layers: a visible layer and a hidden layer as shown in figure 3 where the units of the visible layers are connected with weighted connections to the hidden layer and viceversa with the restrictions that units from the same layer are not connected (a special case of Boltzmnann machine where they are allowed to be connected). The hidden layer only consists of binary units, either 0 or 1, but the visible layer could be of real-valued data.

An energy function $E(v, h)$ is defined to map the visible and hidden units configuration of the RBM to an energy space and the training algorithm consists of driving the model parmeters (connections weights and units bias) such that it maps a current configuration to a locl minimum of energy state.

$$E(v, h) = -b'v - c'h - h'Wv \tag{3}$$

where b and c are the bias of the visible and hidden units respectively and W is the connections weights matrix, v and h are the values of the input units and hidden units respectively.

When presented an input unit, the hidden units binary output is computed as :

$$P(h|v) = sigm(c_i + W_{i.}v) \tag{4}$$

And the visible units could also be reconstructed given hidden units values as:

$$P(v|h) = sigm(b_j + W'_{.j}h) \tag{5}$$

A special case of the real-valued data is to consider the visble unites disributed as a normal distribution. This case is calles *Gaussian-Bernoulli*

RBM (GRBM) and is the main approach to represent real-valued data in acoustic modeling approach. In this case the reconstruction of the visible units is:

$$P(v|h) = \mathcal{N}(b_j + \sigma_j W'_{.j} h, \sigma_j^2) \quad (6)$$

Learning the standard deviations of a GRBM is problematic for reasons described in –ref so for training a GRBM the data is normalized to have a zero mean and a unit variance.

The original algorithm for training a RBM is :

$$\Delta w_{ij} = \epsilon(< v_i h_j >_{data} - < v_i h_j >_{model}) \quad (7)$$

where $< \cdot >$ refers to the expectation.

Getting an unbiased $< v_i h_j >$ sample from data can be done by 4, but getting an unbiased $< v_i h_j >_{model}$ from the model is moremuch more difficult. It's possible to do it by alternating Gibbs sampling starting from a given visible units and sampling each hidden unit sequentially for long period where as shown in —ref convergence is ensured.

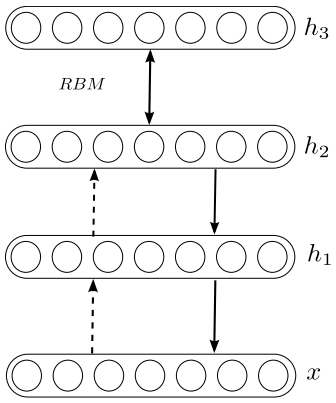
But, an efficient method used is called *Contrastive Divergence* [7] to train the RBM. Given that Gibbs sampling takes computational efforts, Contrastive Divergence –ref is used and instead of the model sample, it's changed with a reconstruction sample:

$$\Delta w_{ij} = \epsilon(< v_i h_j >_{data} - < v_i h_j >_{recon}) \quad (8)$$

The training steps are depicted as follow and is caleld *Contrastive Divergence (CD_1)*:

1. Set the visible units as the input values of the feature vector.
2. Compute the hidden units by using 4
3. Reconstruct the visible units given the hidden units using 5
4. Compute $< v_i h_j >_{data}$ by multiplying $c_1 = h' * v$.
5. Compute the hidden units by using 4, we will refer to the values of the hidden units in the second iteration as h_2
6. Reconstruct the visible units given the hidden units v_2 using 5

Figure 4: Deep Belief Network.



7. Compute $< v_i h_j > recon$ by multiplying $c_2 = h'_2 * v_2$
8. $\Delta w = \epsilon(c_1 - c_2)$
9. Repeat the process for each feature input.

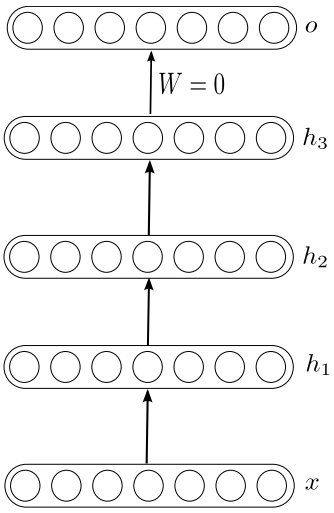
It's enough to perform only one iteration of Contrastive Divergence and that's why it's symbolized as CD_1. Experiments with more iterations didn't yield significant improvement. What happens in theses energy based generative models is that they are not only trying to encode the input but also to capture the statistical structure in the input, by approximately maximizing the log-likelihood (CD_1 is an approximation and not the excact log-likelihood).

After succesfully training an RBM, it's possible to sample from it supposing that it learned efficiently the underlying statistical structure of the data. An important note is that training the RBM is an *unsupervised learning* method as no labels/-targets are required.

5.2 Deep Belief Network

The RBM is the building block of a DBN where RBM's are stacked one on the top of the other sharing layers to build a DBN for the unsupervised pre-training. The Idea here is stack RBM's one on top of another in a way that the hidden unit of the lower RBM is the input unit of the upper level RBMas shown in figure 4.

Figure 5: Deep Neural Network.



Then training DBN consists of training stacked RBM's sequentially one after another given an input vector in the first layer. So basically each layer encodes the satistical structure of the layer directly beneath it and in a way a *Multi-Level Feature Representation* is created of the original input vector.

5.3 DBN-Deep Neural Network

After performing the unsupervised pre-training of the DBN, the undirected connections between layers in the DBN are converted to directed connections and an output layer of the desired type (logistic, sigmoidal, softmax etc..) is added as the top-level layer as shown in figure 5 to form a *Deep Feed Forward Network or Deep Neural Network* with it's connection weights set to zero. After converting a DBN into a DNN a *fine-tune* step is performed using the Backpropagation algorithm. Note that this step is used with labels as a normal Backpropagation algorithm.

6 Hybrid HMM/DNN (Deep Neural Networks) Approach

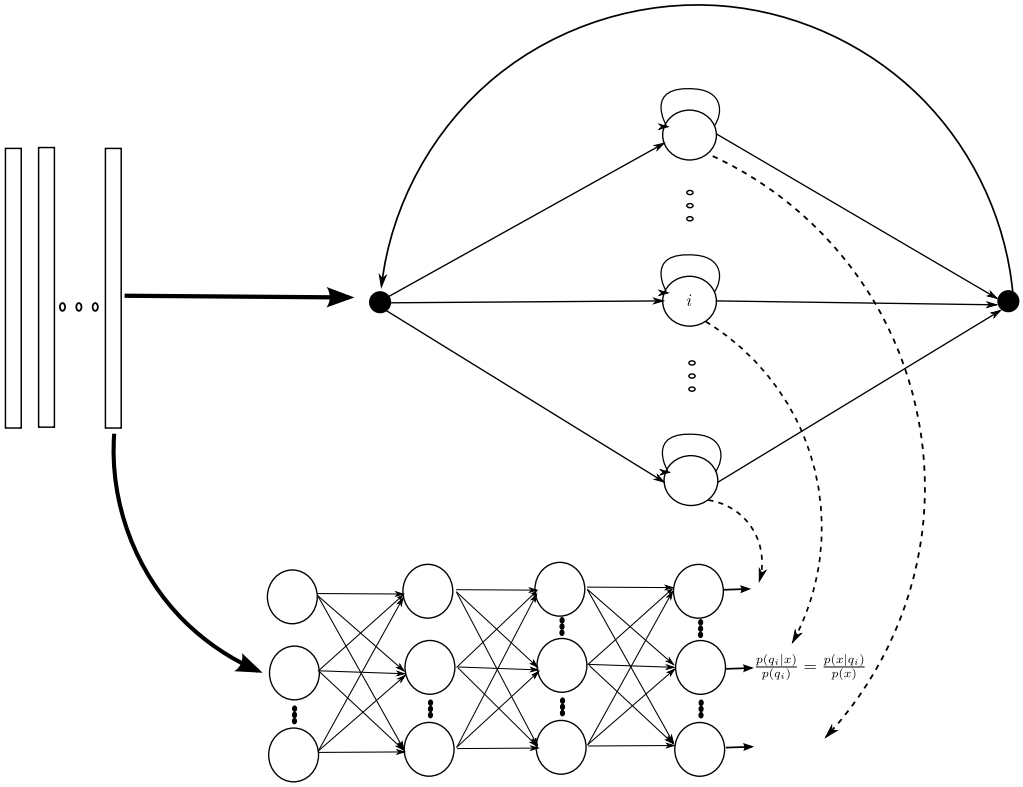
The hybrid approach of interfacing a Hidden Markov Model with a Deep Neural Network is similar to the hybrid HMM/MLP approach but with an unsupervised pre-train step before starting with the supervised training algorithm. So first an unsupervised training of the DBN is performed with the desired number of stacked RBM's with **all** the acoustic features is performed to drive the weights to initial values that the supervised training will use. Then the DBN is converted to a DNN by adding an output layer with a number of output units as the number of states of the HMM trained with the same supervised training as shown in 4.

In figure 6 shows the final architecture of the Hidden Makov Model interfaced with the Deep Neural Network. The final architecture is pretty similar to interfacing HMM with MLP but with more hidden layers. Reuslts of different architectures with the same approach of HMM/DNN on the TIMIT database are shown in table 6 .

Method	PER
Bayesian Triphone HMM-GMM	25.6%
Monophone DBN-DNN (Six Layers)	22.4%
Monophone DBN-DNNs On FBANK (Eight Layers)	20.7%
Monophone MCRBM-DBN-DNNs (Five Layers)	20.5%

Nowadays the best results of Automated Phone Recognition on the timit database were obtained by using a hybrid approach of HMM/DNN with five layers with a *Phoneme Error Rate (PER)* of 20.5%. (add results table)

Figure 6: HMM/DNN Final architecture.



7 Summary

Based on the results of phone recognition on the timit database, deep architectures seem to be promising considering that it has outperformed the traditional systems based on Gaussian Mixtures. One of the key feature in using generally a neural network is the lack of assumption of uncorrelated fetures and thus a better modelling of the emission probabilities but it's more difficult to use given that there is a need in labaled data that most of time we don't have. In deep architectures there is no need of labeled data in the pretraining phase thus giving a solution to that difficulty so with deep neural networks it's possible to pretrain with huge amount of unlabeled data and then finetune with a much smaller labeled dataset thus we have both advantages here: correlated features assumption and the use of unlabeled data in pretraining.

References

[1] Geoffrey Hinton, Li Deng, Dong Yu, George Dahl, Abdel rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara Sainath, and Brian Kingsbury. Deep neural networks for acoustic modeling in speech recognition. *IEEE Signal Processing Magazine*, 29(6):82–97, 2012.

[2] G. Dahl, D. Yu, L. Deng, and A. Acero. Context-dependent pre-trained deep neural networks for large vocabulary speech recognition. *IEEE Trans. on Audio, Speech, and Language Processing*, 20(1):30–42, 2012.

[3] J.S. Garofolo, L.F. Lamel, W.M. Fisher, J.G. Fiscus, and D.S. Pallett. Darpa timit acoustic-phonetic continous speech corpus cd-rom. nist speech disc 1-1.1. 1990.

[4] Yoshua Bengio. Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 2(1), 2009.

[5] Lawrence R. Rabiner. Readings in speech recognition. chapter A tutorial on hidden Markov models and selected applications in speech recognition, pages 267–296. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1990.

[6] Herve A. Bourlard and Nelson Morgan. *Connectionist Speech Recognition: A Hybrid Approach.* Kluwer Academic Publishers, Norwell, MA, USA, 1993.

[7] Miguel A. Carreira-Perpinan and Geoffrey E. Hinton. On contrastive divergence learning.