Deep Learning Approach for Audio Signal Processing Applications

Master of Engineering
Information Technology
Shafait Azam
Matriculation No: 1345243
shafait.azam@stud.fra-uas.de

***Abstract*—Deep Learning approach provides multiplayer of abstraction and information retrieval process in the field of digital signal processing. To implement signal processing systems multilayer of non-linear processing stages with suitable activation function and learning mechanism are used where each lower layer's output is considered as input for the next higher layer. This paper aims to demonstrate different types of discrete time deep neural architectures for real-time non-linear signal processing especially audio signal. It also reviews some of the most important audio signal processing applications like speech recognition, audio signal recovery and music genre classification.**

Keywords—Deep Learning, Deep Belief Network, Multi-layer perceptron, Restricted Boltzmann Machine, Back Propagation, Signal Processing.

I. Introduction

Artificial Intelligence (AI) is used to impersonate almost everything that a human can do. Deep Learning (DL) using Artificial Neural Network is ideal for many signal processing applications. Deep Neural Network (DNN) with different activation functions and weight balancing algorithms can solve complex signal processing problems. In Deep Learning, multi-level of neurons are connected to each other and build a Feed Forward Network. It means that the output of one neuron in a specific layer is treated as a input of another neuron in the next higher layer. In a basic deep-learning architecture, there can be thousands of neurons in each layer with adjustable weight vector and there may be millions of samples to train the machine. In order to adjust the weights of every neurons, learning algorithm such as back propagation algorithm is used which measures a gradient vector for every weights. If the weights are increased or decreased by a small amount, the errors changes accordingly. After that, the weight vector is adjusted depending on the gradient vector [1]. In most of the cases, a technique called stochastic gradient descent (SGD) is used which finds a well balanced set of weights quicker than far more complex optimization techniques. It is called stochastic because for small set of samples it calculates a noisy estimation of average gradient which is calculated over all samples [1]. In order to process signals, deep architectures with multi-layer of non-linear processing units are needed. All the deep learning models that have been developed so far have two common properties: each models requires an additional top layer which is used to perform discriminative task and a large amounts of unlabeled training data is needed for unsupervised pretraining step which extracts structures and regularities from the input features and it helps to reduce the test error significantly[2]. Usually, the optimal number for hidden layers and neurons are determined by heuristic algorithms [14]. If the number of hidden layers are significantly larger then the backpropagation algorithm does not work well. Because it starts at some random initial points to find the local gradient descent. It's main drawback is it often gets trapped in the poor local optima and the result is severe in the case of large and deep networks [2]. To overcome this problem in the field of signal processing deep generative models called Deep Belief Networks (DBNs) are used. DBN uses a greedy aprroach as learning algorithm which optimizes weights at time complexity. In DBNs, fast and greedy algorithm is used for learning one layer at a time and it initializes a slow learning procedure which fine tunes all the weights with the help of wake-sleep algorithm [2] [3]. The combination of pre-initialized Multi Layer Perceptron (MLP) and DBN often produces significantly better results then that with random weights [4]. A DBN is known for its attractive features such as it can be respresented as a Bayesian probabilistic generative models, the overfitting can be observed in the model and the underfitting problem can be identified by the generative pretraining step [2]. Deep Learning first gained it's popularity in the field of image processing applications with the help of Convolutional Neural Networks (CNNs) [5]. But later it was widely adopted in signal processing applications.

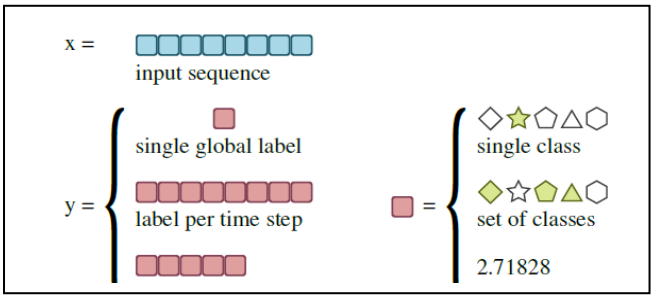

Fig. 1. Figure shows the two properties of audio signal analysis problems: (1) number of labels to be predicted (left), (2) type of each label (right) [6].

Though many deep learning approaches have been adopted from image processing, there are significant difference between audio and image. Raw audio signal is basically a one-dimensional time series signal which is different from two dimensional images. For processing purpose audio signals are transformed into two-dimensional time-frequency representations. Audio signals needed to be analysed sequentially in chronological order [6]. In the field of audio signal processing, slight changes have been made in the deep learning model to process one dimensional audio signals.

II. A PRIME STRUCTURE OF DEEP LEARNING

DBNs are well-known for their first learning greedy approach and pretraining followed by backpropagation fine-tuning. DBNs are considered as probabilistic generative models which consist of multi-layer of stochastic, latent variables. These variables have binary values and are called feature detectors [2]. In the top two layers, there are symmetric and undirected connections between them. The lower layers are designed in a top-down approach where each lower layer have direct connection with the layer above. In the lowest layer or the visible units, input data vectors are fetched. A DBN is constructed by stacking of its components which is called Restricted Boltzmann Machines (RBMs) [2]. RBMs can be depicted as bipartite graphs where all visible units are connected with all hidden units. There are no connections between visible-visible or hidden-hidden units. Adding one hidden unit with correct choice of parameters increases the likelihood significantly. With enough of this hidden units any discrete distribution can be precisely modelled [7]. The Energy function E(v,h; θ) in the RBM is defined in terms of joint distribution p(v,h; θ) where v symbolizes the visible units, h symbolizes the hidden units and θ is the model parameter [2].

$$p(v, h; \theta) = \frac{\exp(-E(v, h ; \theta))}{Z} \quad (1)$$

Here Z is denoted as the normalization factor or partition function and $Z = \sum_v \sum_h \exp(-E(v, h; \theta))$ .The marginal probability which is assigned to visible vector v is denoted by,

$$p(v; \theta) = \frac{\sum_h \exp(-E(v ,h; \theta))}{Z} \quad (2)$$

In the case of Bernoulli (visible)-Bernoulli(hidden) RBM, the energy function is defined as,

$$E(v, h; \theta) = -\sum_{i=1}^{I} \sum_{j=1}^{J} w_{ij} v_i h_j - \sum_{i=1}^{I} b_i v_i - \sum_{j=1}^{J} a_j h_j \quad (3)$$

Here W_{ij} is the symmetric interaction term between visible and hidden unit. The bias terms are denoted as b_I and a_j. Here I and J are the numbers of visible and hidden units [2]. Conditional probabilities are efficiently calculated and it can be denoted as,

$$p(h_j = 1|v; \theta) = \sigma(\sum_{i=1}^{I} w_{ij} v_i + a_j) \quad (4)$$

$$p(v_i = 1|h; \theta) = \sigma(\sum_{j=1}^{J} w_{ij} h_j + b_i) \quad (5)$$

Here $\sigma(x) = 1/(1 + \exp(x))$, which can be derived from the conditional probabilities [8].

For a Gaussian (visible)-Bernoulli(hidden) RBM, the energy function can be represented as,

$$E(v, h; \theta) = -\sum_{i=1}^{I} \sum_{j=1}^{J} w_{ij} v_i h_j + \frac{1}{2} \sum_{i=1}^{I} (v_i - b_j)^2 - \sum_{j=1}^{J} a_j h_j \quad (6)$$

Then the corresponding conditional probability [2] becomes,

$$p(h_j = 1|v; \theta)) = \sigma\left(\sum_{i=1}^{I} w_{ij} v_i + a_j\right) \quad (7)$$

$$p(v_i|h; \theta) = \aleph\left(\sum_{j=1}^{J} w_{ij} h_j + b_i, 1\right) \quad (8)$$

Here v_i is the real value and the Gaussian distribution has the mean $\sum_{j=1}^{J} w_{ij} h_j + b_i$ and the variance one. To convert real-valued stochastic variables to binary stochastic variables, Gaussian-Bernoulli RBMs can be used. RBM weights can be derived from taking the gradient of log likelihood log p(v; θ) [2].

$$\Delta w_{ij} = E_{data}(v_i h_j) - E_{model}(v_i h_j) \quad (9)$$

Where $E_{data}(v_i h_j)$ can be denoted as the expectation which was observed in the training set and $E_{model}(v_i h_j)$ can be denoted as the identical expectation under the distribution which was defined by the model. Here $E_{model}(v_i h_j)$ is difficult to compute it is replaced by running the Gibbs sampler which is initialized at the data for one full step [3]. By combining RBMs layer by layer from bottom-up fashion constructs a DBN. After learning a Gaussian-Bernoulli RBM for the applications with continuous features like speech, the activation probabilities of the hidden units can be considered as data for the training purpose of the Bernoulli-Bernoulli RBM. The activation probabilities of the second-layer can be used visible data which is then used as the input for the third-layer Bernoulli-Bernoulli RBM [2]. This type of stacking procedure increase the efficiency and it improves variational lower bound of the training data likelihood [3]. This learning procedure is unsupervised, no label is specified and maximum likelihood can be achieved. If DBN is applied to classification task such as music genre classification, the pretraining step can be combined with discriminative learning process (by adding a final layer of variables provided in the training data) which is used to fine-tune all the wights jointly to significantly improve the performance. After that the classic Backpropagation algorithm is used to adjust the DBN weights. In the case of speech recognition applications, the output layer can be phones, sub-phones or other units used in the Hidden Markov Model (HMM) systems. Although this learning procedure is expensive, it can be carried out by a single forward pass [2].

III. AUDIO SIGNAL PROCESSING STRATEGIES

Before feeding the data to the Neural Network a lot pre-processing has to be made such as extracting the right features for a specific application, choosing the right loss function, selecting the activation function, design the model with sufficient amount of hidden layers, combining the models (such as DBN with HMM) if necessary.

A. Categorize the Problem

Depending on the kind of target which is needed to be predicted from the input (time series data of audio samples), the problem can be divided into different categories as mentioned in the Fig. 1. It can be a global label or per time step can be represented by a local label. Each label can be a

real value or a set of classes. In audio signal processing applications, sequence classification can predict a language, musical chords or music genre [6]. If a label per time step is to be predicted, each time step can be confined as fixed number of audio samples so that the target sequence length is a fraction of input sequence length. Chord annotation and vocal activity detection can be categorized as the sequence labeling problem. Note onsets or the sudden change of speaker is an event detection problem. Separating a source from a combination of multiple sounds can be considered as the regression per time step problem. Audio synthesis can be considered as a regression task where utilizing a sequence of conditional variable can predict audio samples. Similarly, similarity among audio signals can be considered as regression problem as well [6].

B. Features of Audio Signal

In audio processing, building a proper feature representation and designing an applications specific classifier are considered as two different problems. DNN can be used to perform both feature extraction and classification. In a speech recognition application, the activations of the lower layer of DNNs are considered as features and the activations of the upper layers are considered as performing class-based discrimination [9]. For acoustic modeling in speech recognition DBNs are used because of their higher modeling capacity and their efficient training procedure which combines unsupervised generative leaning for feature extraction and a sufficient stage of supervised learning that fine-tunes the features [9]. A speech wave can be represented by the Linear Predictive Coding (LPC) cepstrum and the logarithmic energy sequences. In the speaker-independent word recognition experiments, time functions of dynamics-emphasized cepstrum and polynomial coefficient for energy are used which indicates that the error rate can be significantly reduced by this method. Because of this reason mel frequency cepstral coefficients (MFCCs) is used as an important features in the field of acoustic analysis [10]. Log-mel spectrum is also a popular feature in audio processing domain. To design any speech recognition system, it is very important to select the best parametric representation of acoustic data. To test the word recognition performance in a syllable-oriented continuous speech recognition system, several parametric representations of acoustic signals (such as mel-frequency cepstrum, a linear frequency cepstrum, a linear prediction cepstrum) were compared [11]. The best performance of 96.5 and 95.0 percent recognition were noticed in the case of a set of ten mel-frequency cepstrum coeffieients which was computed every 6.4 ms. For this experimental result, to represent the important aspects of the short-term speech spectrum mel-frequency cepstrum coefficients are used [11]. Mel-frequency bank is popular method of projecting frequency which is inspired by human auditory system. To make feature extraction of audio signal simpler, various methods have been proposed. Data-driven filters, usage of full-resolution magnitude spectrum, directly fetch the raw waveform representations of audio signals to the visible layer of DNN and learn data driven filters for the target tasks [6]. The lower layers of DNN are designed to impersonate the log-mel spectrum computation, though the filter parameters are learned from data [6]. Deep learning models can be trained to take complex spectrum, which includes both magnitude and phase spectrum as input features [13]. And all the operations of DNN like convolution, pooling and activation functions could be performed in the complex domain [6].

C. Loss Functions:

To design an efficient deep learning based system it is very important to select or design the proper the suitable loss function. For that the loss function is needed to be differentiable with respect to the trainable parameters of system. Mean Squared Error (MSE) is popular method to calculate the error between log-mel spectra which depicts the discrepancies between two audio frames on the basis of their spectral envelopes. But comparing two sine signals on the basis of MSE is not an ideal choice because the loss of two sine signals with identical frequency could depend on their phase difference [6]. A more effective way is to select a differentiable learning loss between time series which is built upon the celebrated dynamic time warping (DTW). It can compare time series signals of variable sizes and is susceptible to dilatations across the time dimension. Soft-DTW is a differentiable loss function and it's value and the gradient are calculated with quadratic time or space complexity [12]. Loss functions can also be application dependent. For the purpose of better result and high accuracy several loss functions can also be combined such as in the case of audio synthesis, one loss function was designed to manipulate the latent variable of Variational Autoencoder (VAE) and another was customized to create changes in the control space [6].

D. Basic Architectures for real-time Audio Signal Processing

1) Multi-layer Perceptron with external memory : DNN architecture with the adaptive non-linear filtering mechanism is suitable two process one-dimesional continuous or real time audio signals. This filters in the case of discrete time sequences can be represented as the relationship between the input sequence $\{x[t], x[t-1]\}$ and the output sequence $\{y[t], y[t-1]\}$[14]. So the general form can be denoted as,

$$y[t] = \phi\{[t], x[t-1],\ldots., x[t-M+1]\} \quad (10)$$

$$y[t] = \phi\{[t], x[t-1],.., x[t-M+1], y[t-1],.., y[t-N]\} \quad (11)$$

From (10), it can be observed that the output is a non-linear function of the input and it represents a nonlinear generalization of linear Finite Impulse Response (FIR) filter. From (11), it can be illustrated that the output signal y[t] is also a function of past output and it represents a non-linear generalization of linear Infinite Impulse Response (FIR) filter. Here (11) represents a general from which is known as Nonlinear Autoregressive Moving Average (NARMA) [14]. In both equation (10) and (11), indexes M and N are filter memory length and the couple (N, M) is the filter order.

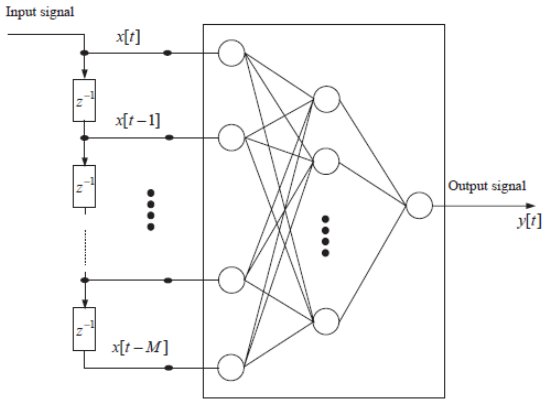

Fig. 2. MLP architecture with input TDL [14].

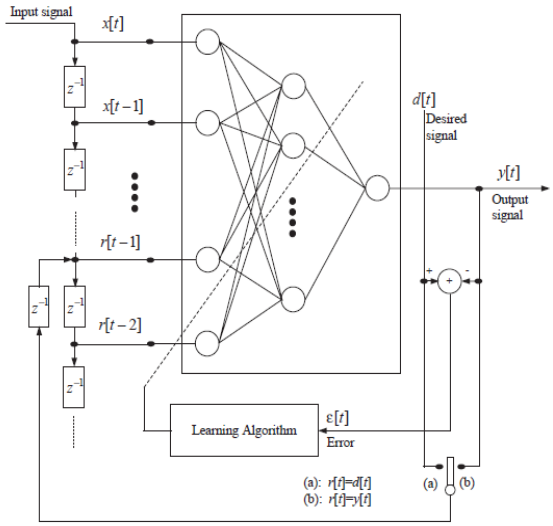

Fig. 3. MLP with input and output TDL. (a) switch for signal correction and (b) switch for error correction [14].

Here in the Fig. 1 and Fig. 2 the Tapped Delay Lines (TDL) are used to get the dynamics from the MLP network [15]. Many traditional signal processing architectures includes FIR-IIR filters and gamma memory Neural Network [14] and because of this reason the conventional TDL's is substituted by s single pole discrete-time filter. These types of networks are ideal for dynamic systems [14]. In this type of above mentioned architectures the main drawback is to determine the optimum filter order and it requires a priori knowledge about the input signals. Here in equations, (10) and (11) ϕ is a linear function and for the calculation purpose filters free parameters can be determined using a lot of methods [14]. The activity of the MLP network can be defined from the input sequence x[t] and from the target or desired signal d[t]. Each sample t has an error $\varepsilon(t)$ which is basically the difference between desired signal d[t] and the output of the network y[t]. In the case of static networks, a popular error calculation mechanism is called Least Square (LS) minimization and it can be defined by [14],

$$J(w) = \frac{1}{K}\sum_{p=1}^{K} \varepsilon(p)^2 \tag{12}$$

In the learning phase, the learning algorithm tries to minimize J(w) over a sample set K which is finite. With use of filters in Fig. 2 and Fig. 3 the error minimization follows the online procedure which means that the sequence can be considered of having infinite length and the process could be time-variant. Learning can be contemplated as a dynamical process which can be implemented with the help of finite length memory mechanism. Sliding window of T_c length [14] is used to implement the time memory limitation and then with the help of equation (12),

$$J(t, w) = \frac{1}{K}\sum_{p=t-T_c+1}^{t} \varepsilon(p)^2 \tag{13}$$

Here the window size and which type of window should be used is completely application dependent. From equation (13), it can be inferred that the learning algorithm for the static networks with external delay lines can be considered as the classic backpropagation algorithm [14]. The input sequence is fed through the sliding window (the delay line in Fig. 2 and Fig. 3) and in each learning step these input samples are translated [15]. In Fig. 3, the network has a output feedback delay line and because of this there are two procedure involved [16] and they are signal correction and (2) error correction. In the case of signal correction, the network is fed with desired signal and in the case of error correction mode delayed output signal is used. The error correction mode shows better convergence property in the case of error correction mode [16]. In practical cases, signal correction mode is first executed and after some iteration the error correction mode is switched on [14].

2) Multi-layer Perceptron with internal memory: If the internal memory of buffer is applied to the input of each neuron the the whole Neural Network could show it's dynamic behaviour. Here single layer of neurons are fully interconnected with each other like Recurrent Neural Network (RNN). But it may imposed some drawbacks such as higher complexity, taking more time to train [14]. By introducing temporal dynamics into the multi;ayer neural network some of the drawbacks can be mitigated. In Fig. 4(a), the static synapses are substitued by conventional IIR adaptive filters [14]. In Fig. 4(b), it is a local feedback recurrent multi-layer network (LF-MLN) where the output of the combined neuron is filterd by an Autoregressive (AR) filter and after that the result is fed to the activation function [14]. In Fig. 4(c), the IIR filter is modified with the feedback loop which passes through the nonlinearity and the one time step dealyed output is then filter by FIR filter. The output of the FIR filter is connected to the input which provides the activation function. In Fig. 4(d), it depicts a multilayer network where each of the neuron is connected with the FIR filter synapses and an AR filter. Because of the dynamic properties of this locally recurrent neural network with internal memory, it requires significantly smaller number of neurons per layer for a specific application [14]. Fig. 4(c) is considered as the most general architecture. To train this network gradient based algorithm called Recursive Back-propagation (RBP) is used. The on-line version of this causal

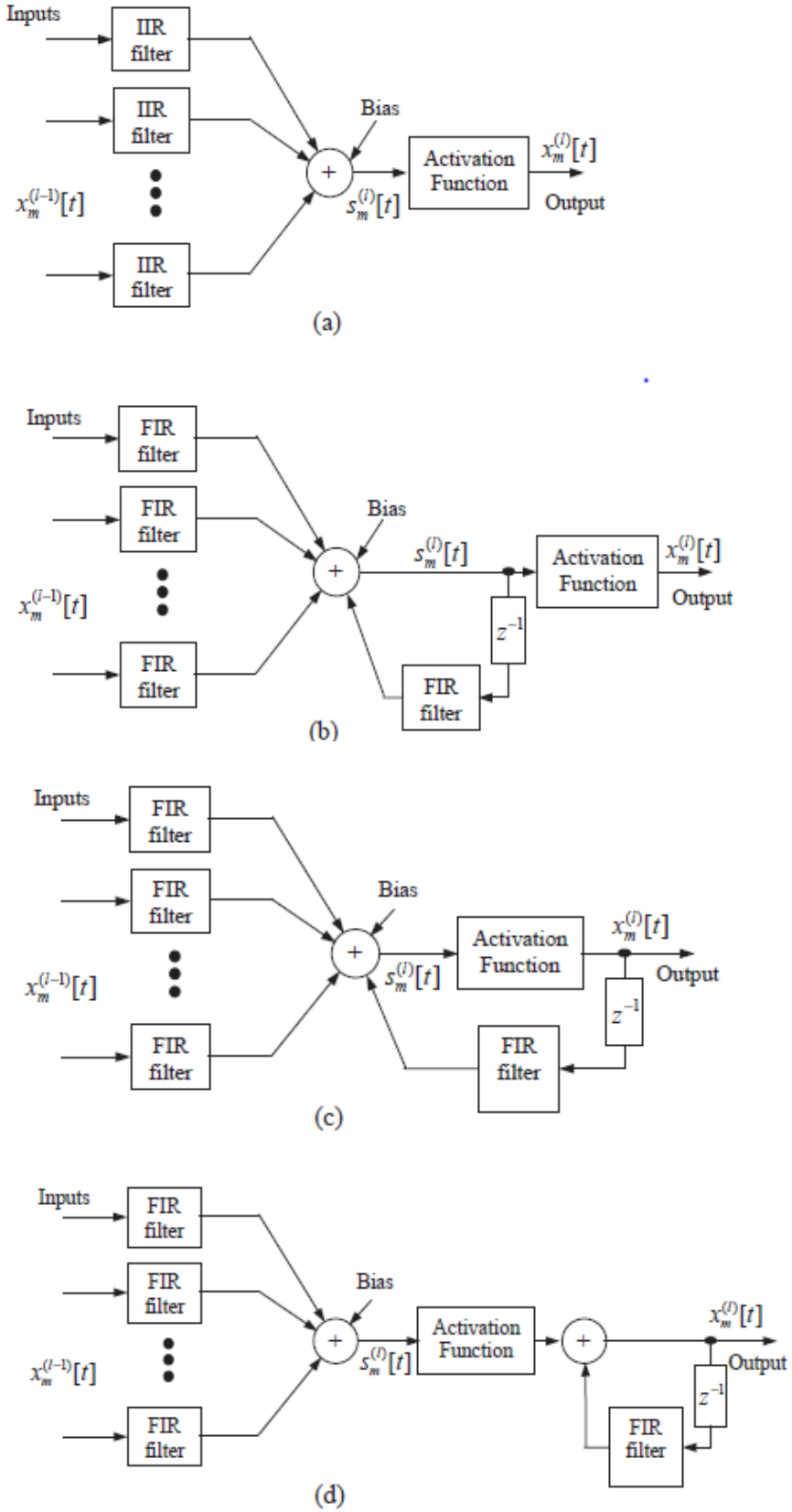

Fig. 4. Dynamic neurons which uses: (a) IIR/FIR synapses (FIR-IIR/MLP); (b) Locally Feedback Multilayer Network (LF-MLN; (c) output feedback; (d) autoregressive MLP [14].

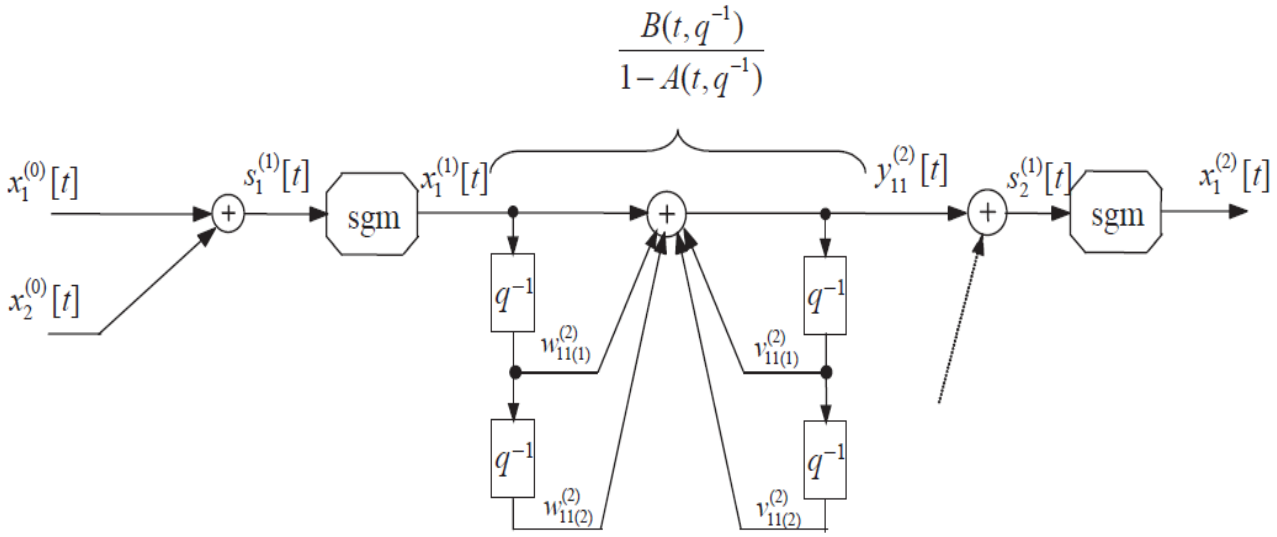

Fig. 5. A simple IIR-MLP with all the notations [14].

algorithm known as Causal Recursive Backpropagation (CRBP) showed some significant advantages in audio signal processing on-line applications [14]. The forward phase can be described as,

$$y_{nm}^{(l)}[t] = \sum_{p=0}^{L_{nm-1}^{l}} w_{nm(p)}^{(l)}\, x_m^{(l-1)}[t-p] + \sum_{p=1}^{I_{nm}^{l}} v_{nm(p)}^{(l)}\, y_{nm}^{(l-1)}[t-p] \quad (14)$$

$$s_n^{(l)}[t] = \sum_{m=0}^{N_{l-1}} y_{nm}^{(l)}[t]; \quad x_n^{(l)}[t] = \mathrm{sgm}(s_n^{(l)}[t]) \quad (15)$$

Here L_{nm-1}^{l} is the MA part and I_{nm}^{l} is the RA part. N_l denotes the number of neurons of the lth layer. $w_{nm(p)}^{(l)}$ and $v_{nm(p)}^{(l)}$ are the coefficients of MA and AR parts. In (14), the direct form I of the IIR has been utilized although other structures are possible. Direct form II structures increase efficiency by reducing the storage complexity and also reduces the computation numbers in both forward and back ward direction [14]. For the clarity purpose from (14) the following can be derived,

$$y[t] = \sum_{p=0}^{M-1} w[t]\, x[t-p] + \sum_{p=1}^{N-1} v[t]\, y[t-p] \quad (16)$$

here $y[t]$ is the output of the IIR filter, $x[t]$ is the input of the IIR filter, $w[t]$ and $v[t]$ are the coefficients of the MA and AR part respectively, $N-1$ is the order of MA part and $M-1$ is the order of AR part. By using different notation, (16) can be written as [14],

$$y[t] = \left(\frac{B(t, q^{-1})}{1 - A(t, q^{-1})}\right)x(t) \quad (17)$$

here,

$$A(t, q^{-1}) = \sum_{p=1}^{N-1} v_p[t]q^{-p}\,; B(t, q^{-1}) = \sum_{p=0}^{M-1} w_p[t]q^{-p}$$

and q^{-1} is the delay operator. For the derivation purpose of the learning algorithm let, $\varepsilon_n[t] = d_n[t] - x_n^{(M)}[t]$ and (13) can be minimized to,

$$J(\theta) = E^2 = \sum_{t=1}^{T} \sum_{n=1}^{N_M} \varepsilon_n^2[t] \quad (18)$$

Here, w and v both are represented by θ. N_M is the number of neurons in the Mth layer and T is denoted as the duration sequence [14]. So the error and the delta of the backpropagation algorithm can be defined by,

$$e_n^{(l)}[t] = -\frac{1}{2}\frac{\partial E^2}{\partial x_n^{(l)}[t]}\,;\ \delta_n^{(l)}[t] = -\frac{1}{2}\frac{\partial E^2}{\partial s_n^{(l)}[t]}$$

For the static case the following [14] equation holds,

$$\delta_n^{(l)}[t] = e_n^{(l)}[t]\mathrm{sgm}'(s_n^{(l)}[t]) \quad (19)$$

IV. Applications of Audio Signal Processing

The implementation of deep neural network in the field of audio signal processing is rapidly increasing. Because of the modern sophisticated computational power and availability of enormous datasets deep learning model can achieve high accuracy when it processes audio signal.

A. Speech Recognition

MLP has been used as the common deep learning model for speech recognition for many years. But MLP alone can not achieve high accuracy in audio signal processing. All the dominant organization have experimented on some of the Benchmark datasets of audio signal with the recently developed Deep Learning Models. By combining the discriminative training ability of the DBNs with the sequential modeling ability of the HMMs can achieve higher accuracy in signal processing applications [2]. In Fig. 6, it depicts the architecture of DBN-HMM model where DBN is used for estimating the observation probability and HMM is used for modelling sequential information [2]. An experiment

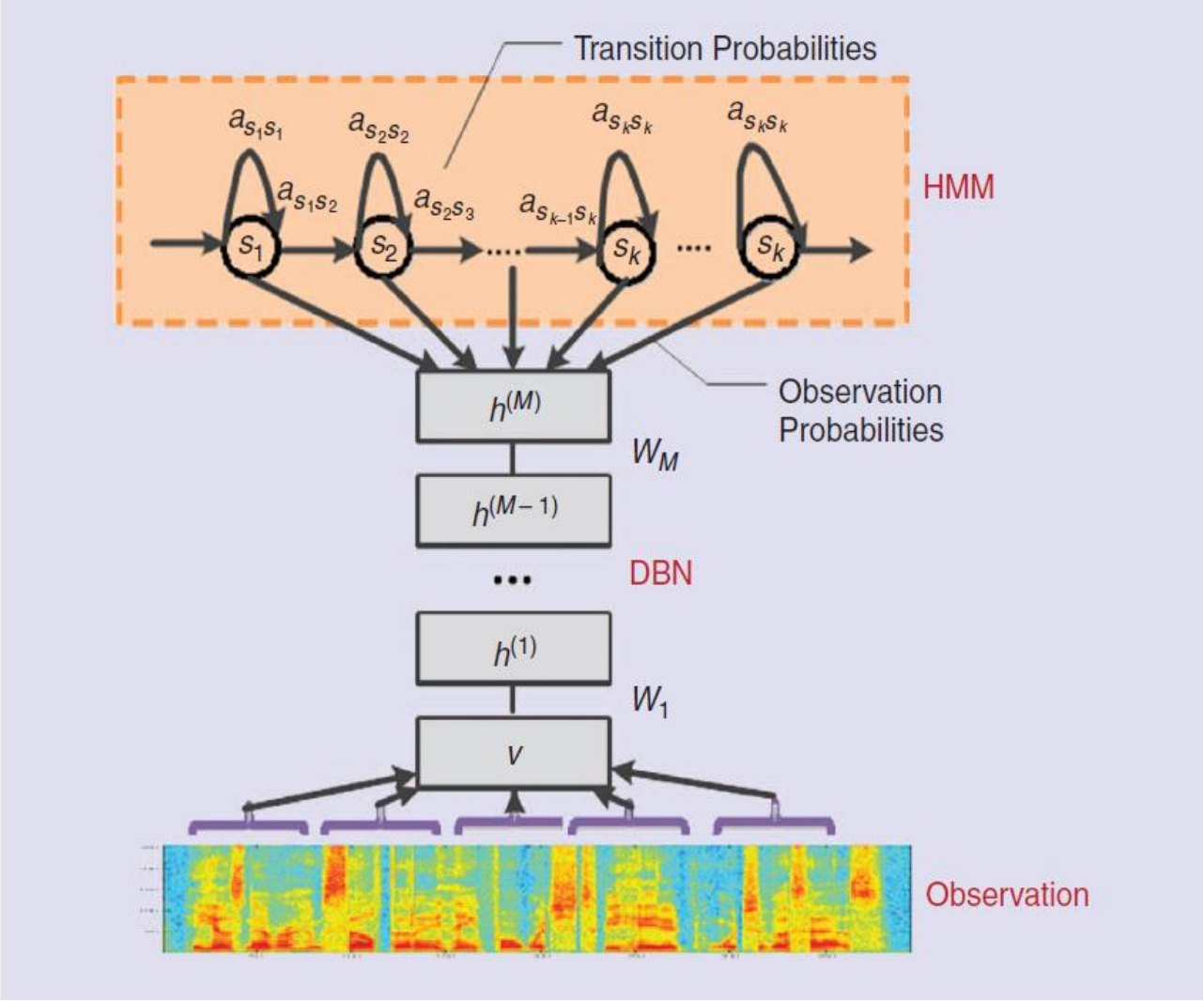

Fig. 6. The Figure shows the DBN-HMM model for speech recognition, DBN is used for successfully estimate the observation probabilities. Here, syllables, phones, subphones, monophone states and senones are considered as state values [2].

was made on the basic TIMIT corpus dataset [17], and a model was set up. Here a five layer of DBN was combined with the monophonestate. Despite of using the monophone model, the DBN-HMM structure achieved higher accuracy in phone recognition [17]. The DBN-HMM model is so powerful that it can be applied not only as a context-independent model but also as an context dependent model. A lot of experiments have been made on the challenging Bing mobile voice search data set and the DBN-HMM performs significantly well and achieved high accuracy [2]. A vector quantization technique is outperformed by the deep auto encoder if it uses the DBN pretraining. It shows poor performance if all the weights are randomly initialized in the deep auto encoder. Though DBN is very popular for audio signal processing, other types of deep models are also being used in this field such as deep-structured CRF have shown significant result in language identification, phone recognition and other speech-related tasks [2]. DBN is also used for phone recognition and it has shown significant result.

1) Deep Belief Network in Phone Recognition: DBN uses an effective training mechanism known as RBM as their core blocks. Because of certain properties, DBN is ideal for classification tasks. The hidden units that constructs the DBN block can learn to represent features. This features represents correltions with the higher order in the input data [17]. In the training procedure of the DBN, the model parameters θ is learned by RBM from which p(v|h, θ) and the prior distribution p(h| θ) can be defined. So the probability of constructing a vector v can be described as,

$$p(v) = \sum_h p(h|\,\theta)\, p(v|h,\,\theta) \qquad (20)$$

So DBN is basically an RBM which presents a prior on the top hidden layer. It is amalgamated with the recognition weights for fast approximation inference [17]. An experiment of phone recognition on the TIMIT corpus [18] dataset have been performed [17]. It is consists of a training set of 462 speaker. For model tuning a set of 50 speakers was used and the core test set consists of 24 speaker. Size of 25 ms Hamming window having 10 ms between the left edges of next frames is used to analyse the speech. The speech was represented by 12th-order MFCCs with it's first and second temporal derivatives and energy with it's first and second temporal derivatives. Then the data were normalized so that it can have a mean of zero and variance of one over the entire dataset [17]. Class labels of 183 has been used and window

size of 11 frames has been used. Using TIMIT corpus dataset in this experiment, the phone error rate (PER) for the test set has been calculated [17]. In Fig. 7, it depicts the Phone rate error for models having different size of hidden layer. Here Backpropagation-Deep Belief Network (BP-DBN) architecture is used with 2048 hidden neurons per layer [17]. From Fig. 7, it can be illustrated that adding a second layer to the model reduces the phone error rate. The reduction of PER continuous till 4th layer. After that adding another layer did

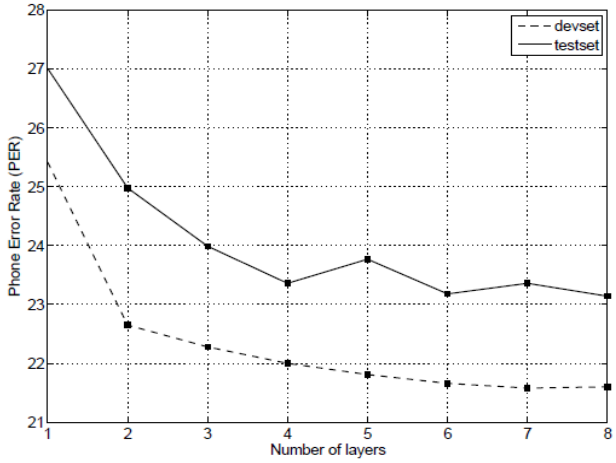

Fig. 7. PER calculation with varying hidden layers [17].

not create any significant changes in the error calculation. Here in table I, the PER for both development set and test set of 4 hidden layer DBN architectures with different hidden units per layer has been shown [17].

TABLE I. THE PER CALCULATION OF DIFFERENT LAYER SIZE [17]

Model	**Devset**	**Testset**
1024 units	21.94%	23.46%
2048 units	22.00%	23.36%
3072 units	21.74%	23.54%

TABLE II: RESULTS ON TIMIT CORE TEST SET [17]

Method	**PER**
Stochastic Segmental Models	36%
Conditional Random Field	34.8%
Large-Margin GMM	33%
Augmented Conditional Random Fields	26.6%
Recurrent Neural Nets	26.1%
Bayesian Triphone HMM	25.6%
Monophone HTMs	24.8%
Heterogeneous Classifiers	24.40%
CD-HMM	27.3%
Deep Belief Networks (DBNs)	**23.0%**

The 4 layer with 2048 units have shown significantly better performance in the core test set. A lot of experiments have been performed in the TIMIT corpus with different deep learning models. In table II, it clearly depicts the comparison of PER results of different models that have been experimented on the TIMIT corpus dataset. The BP-DBN architecture with 4 layers of 2048 hidden units per layer with the bottleneck layer of 128 units (to avoid overfitting) achieved better result than any other models mentioned in table II. The 183-way softmax has been used [17].

B. Audio Signal Recovery

By using the DNN distorted audio signal can be restored. Modern technology is growing at a rapid pace and in the media industry analog technology is being replaced by digital technology which can store streams of numerical data. Sometimes the original signal may get corrupted by environmental noise and sometimes sample sequence may get lost during transmission. To overcome such circumstances audio restoration is necessary. The back-ground noise can be created by the transducer equipment. Impulsive or environmental noise can be created due to nature surface scratches and dust particles. Sometimes portion of musical track may get missing because of the old magnetic tapes [14]. To reconstruct the missing data the signal can be modelled as autoregressive process. Non-linear prediction method using Neural Networks can successfully predict the missing portion of the signal. MLP NNs with hidden layer units having bipolar sigmoid activation function and the output layer having a linear neuron has been used. The network has been trained over different samples which have been gathered from uncorrupted signals [14] and forward and backward predictions have been performed. To learn the local characteristic of the signal, a new training stage have been added before and after the data is missing. In the subband signal reconstruction method two different filter banks have been added and they are called uniform filter bank and octave filter bank [14]. A non-linear predictor can process each subband like the Adaptive Spline Activation Function Neural Network (ASNN). ASNN can work in continuous learning mode and it behaves like simple nonlinear adaptive filters. Missing samples of an audio signal can be viewed as an extrapolation problem. For every input $x_i[t]$ of the signal forward and backward prediction algorithm have been applied. The results of these predictions have been combined with the help of weighing windows which is shown in the Fig. 8 as cross fade operation [14]. The prediction result

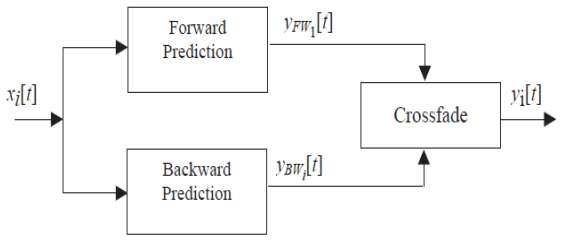

Fig. 8. The Forward-Backward Prediction Scheme [14].

then passed to the synthesis filter bank in order to achieve reconstructed signal y[t]. In Fig. 9, it shows an experimental result of constructing 45 ms of signal with the help of octave filter bank and ASNNs [14].

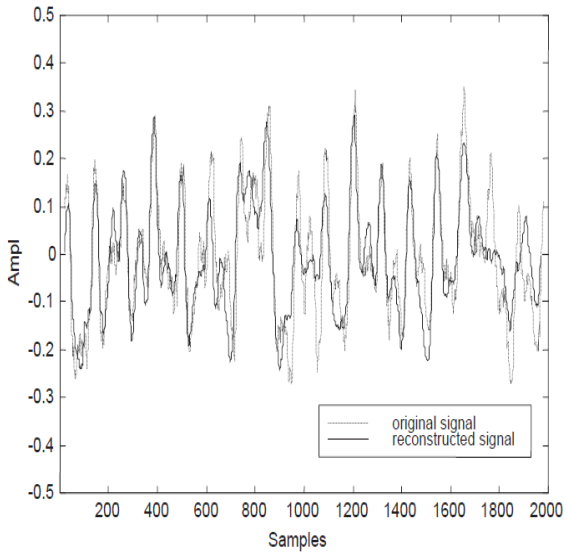

Fig. 9. The reconstruction of 2000 samples (45 ms) with the help of spline neural networks and octave filter bank [14].

C. Music Processing

Music is not like speech, even it is the combination of a variety of sound created from different sources and this will cause complex dependencies. Onset and offset detection and fundamental frequency estimation are considered as low-level analysis in music processing applications. Tempo estimation and beat tracking are known as rhythm analysis. Music genre classification, instrument detection is considered as high level analysis [6]. To find whether an onset is present in or near the centre an MLP was trained depending upon 200 ms excerpts of a constant-Q log magnitude spectrogram. The prediction result was significantly higher than using a Short Time Fourier Transform (STFT) [6]. This method was improved with a bidirectional Long Short Term Memory (LSTM) and a time difference filter. This improved method was applied to a larger dataset. RNN model which trained on spectrograms can track beats. With the help of a CNN an high level task like to predict borders between musical segments has been deployed where strongly downsampled spectrograms are used [6]. A CNN which was trained on Constant-Q and the linear-magnitude spectrograms are preprocessed by contrast normalization and then it was augmented with pitch shifting, showed good result in predicting musical chords [6]. CNN which was bult upon log-frequency spectrograms showed significant results in not only predicting chords but also represented an effective chromagram representation. CNNs are very powerful model and it has been showed that it can predict tempo from 12-second spectrogram excerpts [6]. Different deep learning model have been applied in the music processing applications to build a system which can automatically recognize chords, songs or even segment different instrument sources from a song.

1) Music Genre Classification: Songs, artists and albums can be segemented by music genre classification. Deep Belief neural networks using RBMs can solve the mullti class music genre classification problem. RBMs can be used as a pre-training step to overcome the drawback of back propagation algorithm which tends to stuck in the local minima [19]. An experiment in the GTZAN dataset showed the accuracy of NN and DBN in music classification. The GTZAN dataset contains 1000 audio tacks. Each of these track is 30 seconds long. There are total 10 genres shcu as, classical, jazz, ,metal, pop, country, blues, disco, metal, rock, reggae and hiphop. Each of the tracks is 30 second long so they can be represented as, 30 seconds * 22050 sample/second = 661500 length of vector. It is very long vector. So to traing the DBN model MFCC features is used. A hamming window of size 25 ms with 10 ms overlap is used. Then the fourier transform is applied to each frames to get the frequency component. After that a mapping has been done from frequency to mel scale (pitch scale). After the conversion, log of mel scale has been taken. Log mel scale then transformed by Discrete Cosine Transform (DCT). In the experiment MFCC features are divied into 4 same size portion and for later experimental purpose 13 * 160 = 2080 length of MFCC which represent 30 second audio file has been generated [19]. The fundamental structure of the DBN is a total of 5 layers with 3 hidden layers. Then each layer except the ouput layer is trained by RBMs and put them in the top of other like a stack. After the RBM training, forward and backward are used to fine tune the network weights [19]. The 60% of total mfcc features are used as training set and 40% of data was kept for testing set. The network was first used to classify two class and the then for three class classification.

TABLE III: 2 CLASS CLASSIFICATION RESULT [19].

Classic vs Metal	NN	DBN
Accuracy on train set	100%	100%
Accuracy on test set	97.5%	98.75%

TABLE IV: 3 CLASS CLASSIFICATION RESULT [19]

Classic, Metal and Blues	NN	DBN
Accuracy on train set	90%	91%
Accuracy on test set	70.8%	69.16%

From table III, in the two class classification both NN and DBN could classify two genres and achieved high accuracy. But in the case of three class classification problem, DBN has high accuracy in train set and lower accuracy in test set. The result is significantly worse than NN in the test set shown in table IV. It happens because of overfitting [19]. Now in the case of larger dataset the DBN performed better than NN in the test set when it is applied to three genre classification shown in table V and four genre classification shown in table VI [19]. From this experiment it can be said that DBN performs well in the presence of larger dataset.

TABLE V: 3 GENRE CLASSIFICATION RESULT WITH LARGE DATASET. [19]

Classic, Metal and Blues	NN	DBN
Accuracy on train set	92.89%	94.5%
Accuracy on test set	77.17%	77.94%

TABLE VI: 4 GENRE CLASSIFICATION RESULT WITH LARGE DATASET. [19]

Classic, Metal, Disco and Blues	NN	DBN
Accuracy on train set	94.69%	75.3%
Accuracy on test set	60.46%	61.15%

V. Conclusion

Deep Learning Models with different structures are ideal for audio signal processing. Detecting source from long distance, audio signal enhancement, audio synthesis and music genre classification using DL are some of the most popular applications in audio signal domain. DBNs are very sophisticated and powerful network which is often used to process one-dimensional on-line audio signals. DBN use RBM as learning mechanism and it also uses Backpropagation for fine tuning the weights. LSTM, CNN and RNNs are used to implement different music processing applications. DBNs with RBM as the learning algorithm can achieve high accuracy in the case of music genre classification. Though there are two and three dimensional signal like image and video, I have confined my to audio signal and described the architecture, analysis and experimentation of audio signal processing with deep learning in this paper.

VI. Acknowledgement

I would like to thank Prof. Dr. Andreas Pech for Providing me an opportunity to work on this topic and also for providing me the proper guidance to write this paper.

VII. References

[1] Lecun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, 521(7553), 436-444. doi:10.1038/nature14539

[2] Yu, D., & Deng, L. (2010). Deep learning and its applications to signal and information processing [exploratory dsp]. IEEE Signal Processing Magazine, 28(1), 145-154

[3] Hinton, G. E., Osindero, S., & The, Y. W. (2006). A fast learning algorithm for deep belief nets. Neural computation, 18(7), 1527-1554

[4] Deng, L., Seltzer, M. L., Yu, D., Acero, A., Mohamed, A. R., & Hinton, G. (2010). Binary coding of speech spectrograms using a deep auto-encoder. In Eleventh Annual Conference of the International Speech Communication Association

[5] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems, 25, 1097-1105

[6] Purwins, H., Li,B., Virtanen, T., Schlüter, J., Chang, S. Y., & Sainath, T. (2019). Deep learning for audio Signal processing. IEEE Journal of Selected Topics in Signal Processing, 13(2), 206-219

[7] Le Roux, N., & Bengio, Y. (2008). Representational power of restricted Boltzmann machines and deep belief networks. Neural computation, 20(6), 1631-1649

[8] Bengio, Y. (2009). Learning deep architectures for AI. Foundations and Trends in Machine Learning, 2(1), 1-127. doi:10.1561/2200000006

[9] Mohamed, A. R., Hinton, G., & Penn, G. (2012, March). Understanding how deep belief networks perform acoustic modelling. In 2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 4273-4276). IEEE

[10] Furui, S. (1986, April). Speaker-independent isolated word recognition based on emphasized spectral dynamics. In ICASSP'86. IEEE International Conference on Acoustics, Speech and Signal Processing (Vol. 11, pp. 1991-1994). IEEE

[11] Davis, S., & Mermelstein, P. (1980). Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. IEEE transactions on acoustics, speech and signal processing, 28(4), 357-366

[12] Cuturi, M., & Blondel, M. (2017, July). Soft-dtw: a differentiable loss function for time-series. In International Conference on Machine Learning (pp. 894-903). PMLR

[13] Wilmanski, M., Kreucher, C., & Hero, A. (2016, December). Complex input convolutional neural networks for wide angle SAR ATR. In 2016 IEEE Global Conference on Signal and Information Processing (GlobalSIP) (pp. 1037-1041). IEEE

[14] Uncini, A. (2003). Audio signal processing by neural networks. Neurocomputing, 55(3-4), 593-625. doi:10.1016/s0925-2312(03)00395-3

[15] Narendra, K. S., & Parthasarathy, K. Identification and control of dynamical systems containing neural networks. IEEE Trans. Neural Networks 1(1990), 4-27

[16] Regalia, P. (2018). Adaptive IIR filtering in signal processing and control. Routledge

[17] Mohamed, A. R., Dahl, G., & Hinton, G. (2009, December). Deep belief networks for phone recognition. In Nips workshop on deep learning for speech recognition and related applications (Vol. 1, No. 9, P. 39)

[18] Garofolo, J. S., et al. TIMIT Acoustic-Phonetic Continuous Speech Corpus LDC93S1. Web Download. Philadelphia: Linguistic Data Consortium, 1993

[19] Feng, T. (2014). Deep learning for music genre classification. Technical report, University of Illinois