Structural Optimization of Deep Belief Network and Its Application for Speech Recognition

D161685
Murman Dwi Prasetio

Hiroshima University
Higashi-Hiroshima, Hiroshima

Structural Optimization of Deep Belief Network and Its Application for Speech Recognition

Abstract

Since the first time computers were created, humans have thought about how to make computers learn from experience. The term *machine learning* is basically a computer process for learning from data. Therefore, it continues to interact with data. All machine learning knowledge will definitely involve data. Data can be the same, but the algorithms and approaches vary to get optimal solutions. Machine learning is a branches of the artificial intelligence discipline which discusses the development of systems based on data. Many things are learned, but basically there are 4 main techniques learned for machine learning.

The first thing preparation for supervised learning is data. Data will usually be divided into 2 groups, namely *training data* and *testing data*. Training data is used to train algorithms to find suitable models, while testing data is used to test and determine the performance of the model obtained on the testing stage. From the model obtained, the predictions of data that are divided into two types depending on the type of output. If the prediction results are discrete. For examples gender classification took from speech (male and female output) is the classification process. While if the thread is continuous, it is called regression process. For examples of applying machine learning in life is detecting a person's disease from the existing symptoms. Another example is detecting heart disease from an electrocardiogram recording. In the case of information retrievers, language translation using a computer, by converting voice into text and spam email filters it is used for detecting and classifying sounds.Natural Language Processing (NLP) is the ability of computers to understand both written text and human speech. NLP techniques requires to capture the meaning of an unstructured text from documents or communication from the users. Therefore, NLP is the primary way that systems can interpret text and spoken language.

Computer generated communication by voice has existed for a while now. Automatically understanding speech allows for use in various applications. an utterance is our most natural form of interaction when working with people, but still cannot be reached as a reliable interface between humans and machines. Although they are catching up, even the production quality systems like Google, Apple's Siri or Amazon Alexa are far off from what a human-generated speech would sound

like recently. Building a System of Understanding oral Language (SLU) requires solving several sub-problems, each of which presents significant challenges in itself.

A similar neural network results in a model for molecular activity prediction substantially more effective than production systems used in the pharmaceutical industry. Even though training assays in drug discovery are not typically very large, it is still possible to train very large models by leveraging data from multiple assays in the same model and by using effective regularization schemes. In the area of natural language processing, I first describe restricted Boltzmann machine training algorithm suitable for voice data. Then, I introduce a new neural network generative model of parsed sentences capable of generating reasonable samples and demonstrate a performance advantage for deeper variants of the model.

Machine Learning approaches, in particular of neural network to emphasized high-capacity, scalable models that learn distributed representations of their input. The neural network consists of a number of units that have simple nonlinear transfer functions and approximate capabilities for a number of complex types of problems in comparison to a small number of calculations. Therefore, neural networks are applied for data analysis, data mining and data classification. Adequate learning cannot be done if the size of the network is too small. Conversely, over-fitting occurs in the learning data and loses the ability to generalize if the size is too large.

Therefore, the appropriate neural network structure needs to be determined for each target problem for higher neural network performance. Traditionally, neural network structure was determined through a trial and error procedure based on the experience of a neural network designer. However, a very large computational time is required by the determination process.

This dissertation demonstrates the efficiency and generality of structural optimization method of a Deep Belief Network (DBN) which consists of multiple layers Restricted Boltzmann Machines (RBMs) using several kinds of evolutionary computation methods and . DBN has succeeded in acquiring higher data analysis capability by effectively incorporating a feature extraction process which is conventionally performed by trial and error. In DBN, multiple RBMs were incorporated into the learning process as feature extractors. There were two kinds of experimental design to approach the method throughout these studies, data classification or prediction and speech recognition.

The experimental design in data classification or prediction using Caltech101 as benchmark of

image classification in many related literature. The image data of the Caltech 101 are gray-scale 30×30 grids images, and each grid is scaled in the range of $[0,1]$. Images are classified into 4 categories "airplane", "cat", "face", "dolphin". There are 65 images per a category. Here, in the structure of RBM has a weakness such as repeating learning could be consuming the time of structure of evaluation and leads to inefficiency in the structure optimization, so by the modularization structure of optimization is performed efficiently by shortening calculation time. Also, the number of hidden layers and the number of each layer units are optimized, but optimizing them simultaneously optimizes the number of layers and optimizes the number of units first because the search space is too wide.

Neural Network, although the learning degree increases as learning is done, it becomes excessive learning and loses generalization ability. Therefore, it is desirable that a network with both a learning error for learning data and a verification error for data unused for learning are low. First, although the target data is divided, simply dividing it into one for learning and verification excessively conforms to these two data, and for the other verification data. In this study, the division method was set to 1: 4: 5. We evaluate the structure by learning error / generalization ability verification error / verification error at that time.

In the area of speech recognition, its develop a more accurate acoustic model using a deep belief network. First step we are conducted how to build the simplest data-set in voices, its will be extracted into a single data matrix, and a label vector with the correct label for each data file is created. Once the data turned into an input matrix, the next step is to extract features from the raw data, as is done in many other machine learning pipelines. In this experiment, we are used the simple frequency peak detection. The technique to found the peak of signal, it is called the Short Time Fourier Transform (STFT). The Fast Fourier Transform (FFT) is applied over chunks of the input data, resulting in a 2D FFT "image", usually called the spectrogram.

Acoustic-phonetic approach has been studied in great depth for before centuries. This approach is based upon theory of acoustic phonetics and postulates. The earliest approaches to speech recognition were based on finding speech sounds and providing appropriate labels to these sounds. Three aspects of speech processing are investigated: acoustic parameterization, recognition algorithms and acoustic modeling.

DBN performs feature extraction with unsupervised learning called Pre-training and supervised

learning called Fine-tuning are performed based on the extracted features. The structural characteristics of DBNs, it can be considered that there exists a great relationship between the structure of DBM, the number of hidden layers and units constituting each layer, and the performance in data classification or prediction. Performance improvement is expected by giving an appropriate structure corresponding to input data. This model, which uses rectified linear units and dropout, improves the accuracy of classification to predict human voices (male or female) with the accuracy 90%.

Contents

List of Tables

List of figures

Is one who is devoutly obedient during periods of the night, prostrating and standing [in prayer], fearing the Hereafter and hoping for the mercy of his Lord (ALLAH SWT), [like one who does not]? Say, "Are those who know equal to those who do not know?" Only they will remember [who are] people of understanding. Quran 39:9

Acknowledgments

Al-Baqaraa 2:195, I would like to express my deep gratitude to Professor Tomohiro Hayashida and Professor Nishizaki, my research supervisors, for their patient guidance, enthusiastic encouragement and useful critiques of this research work. I would also like to thank Prof.Ichiro Nishizaki and Prof. Shinya Sekizaki, for their advice and assistance in keeping my progress on schedule. I am also grateful to the members of my committee for their patience and support in overcoming numerous obstacles I have been facing through my research

I would like to thank my colleague in System Cybernetics Laboratory for their feedback, cooperation and of course friendship. In addition, I would like to express my gratitude to the staff of Electrical Engineering A1 Building for the last minute favors. Specially, for my wife Mihrani Herdiska thank you for your support, dedication, kindness to support my study until this is going to next step. My lovely children, Ahmad Kheidera Farrelio, Kayreen Seika Talitha the best and precious that ALLAH given to me.

Finally, I wish to thank my parents for their support and encouragement throughout my study.

1 Introduction

1.1 Background and Aim

Nowadays, we are awarded with a lot of data (even unlimited), machine learning techniques become intuitive to do inference on large data. This causes machine learning to become popular because the construction of the inference model can be done automatically. Machine learning is like a "tool", just like a mathematical formula. It is one form of artificial technology (AI) that allows computers to learn directly from samples and experiences in the form of data. The traditional algorithm approach to programming depends on hard-coded rules, which determine how to solve problems, step by step. When you first hear the term "artificial intelligence", you might think of a robot that has a physical body. However, artificial intelligence is not only limited to something that has a physical body.

In context of the data mining step, it is important to choose the correct approach for tackling the task appropriately. This is often done using machine learning methods. A major difference between humans and computers has been for a long time that a human beings tend to automatically improve their way of tackling a problem. Humans learn from previous mistakes and try to solve them by correcting them or looking for new approaches to address the problem. Traditional computer programs do not look at the outcome of their tasks and are therefore unable to improve their behavior. The field of machine learning addresses this exact problem and involves the creation of computer programs that are able to learn and therefore improve their performances by gathering more data and experience.

The first scientist to create a self-learning program was A. Samuel in 1952, who created a program that became better at playing the game checkers with the number of games played[89]. In 1967, the first pattern recognition program was able to detect patterns in data by comparing new data to known data and finding similarities between them[8]. Since the 1990's machine learning is used in data mining areas, adaptive software systems as well as text and language learning fields[18]. As an example: A computer program that gathers data concerning the customers of an e-commerce shop and creates better personalized advertisements out of these pieces of information has the ability to acquire new knowledge and comes close to being artificial intelligence (AI)[103].

AI algorithms and machine learning are not new. The AI field was introduced from the 1950[86]. An IBM researcher, developed one of the earliest machine learning programs for an independent learning program to play chess[12]. In fact, he invented machine learning terms. His approach to machine learning is explained in a paper published in the IBM Journal of Research and Development in 1959. For decades, AI techniques have been widely used as a method to improve the performance of the underlying code. Today, catching up the technology of machine learning become interested due to the algorithms of machine learning are available through an open source community with a large user base. Therefore, there are more resources, frameworks and libraries that have made development easier[33].

Furthermore, machine learning systems are normally classified by their underlying learning strategies,which are often identified by the amount of inference the computer program is able to perform: Rote Learning, Learning from Instruction, Learning by Analogy and Learning from Examples

Rote learning describes the strategy that all traditional computer programs use. They do not perform any kind of inference and all their knowledge has to be directly implemented by the programmer, since the application is not able to draw any conclusions or transformations from the given information.

Learning from instruction encompasses all computer programs that are able to transform information from the given input language to an internal language. Although the knowledge on how to effectively perform this transformation is still given by the programmer, this requires little forms of inference from the side of the computer program. Therefore, this defines a separate level of learning system compared to rote learning.

In contrast to Learning from Instruction, **learning by analogy** tries to develop new skills that are almost similar to existing skills and therefore easy to adopt, by performing transformations on known information. This system requires the ability of creating mutations and combinations of a dynamic knowledge set. It creates new functionality, which were unknown to the original computer program and therefore requires a lot of inference.

Learning from Examples is nowadays one of the most commonly used learning strategies as it provides the most flexibility and enables computer programs to develop completely unknown skills or find unknown structures and patterns in data[15]. Learning from examples is a technique that is often used in classification and data mining tasks to predict the class label of new data entries

based on a dynamic set of known examples. In this work, the proposed research questions will be tackled with strategies and algorithms that belong to this category.

The following most common machine learning systems, are identically with neural network systems or evolutionary computation method. Which consists of multiple Restricted Boltzmann Machines (RBMs) and a single feed-forward Neural Network (FNN). An Neural Networks as "a mathematical model that is based on biological neural networks and therefore is an emulation of a biological neural system".

Compared to conventional algorithms, neural networks can solve problems that are rather complex, on a substantially easier level in terms of algorithm complexity. Therefore, the main reason to use Artificial Neural Networks is their simple structure and self-organizing nature which allows them to address a wide range of problems without any further interference by the programmer. Example given, a neural network could be trained on analysis the proceeding data on the journal system to classify which data containing word and number[69].

Neural Network consists of nodes, also called neurons , weighted connections between these neurons that can be adapted during the learning process of the network and an activation function that defines the output value of each node depending on its input values. Every neural network consists of different layers. The input layer receives information from external sources, such as attribute values of the corresponding data entry, the output layer produces the output of the network and hidden layers connect the input and the output layer with one another. The input value of each node in every layer is calculated by the sum of all incoming nodes multiplied with there selective weight of the interconnection between the nodes[100].

The important part in machine learning, is the problem of how a computer program notices which of its results were appropriate and which contained mistakes. An example where this poses no problems to the algorithm would be a computer program that tries to predict whether a customer in an e-commerce shop will perform a purchase or not. The data entry will afterwards be logged with the given information whether the customer bought articles or not and can then be used to evaluate the performance of the algorithm. More difficult scenarios come up in research areas with limited or no access to real-world data, such as the evaluation of document translations. This requires an additional human effort to rank given translations into classes to be able to compare the computer program results in the end.

The evaluation of classification tasks is normally done by splitting the data set into a training data set and a test data set. The machine learning algorithm is then trained on the first one, while the test data set is used to calculate performance indicators in order to evaluate the quality of the algorithm. A common problem for machine learning algorithms lies in the access to limited test and training data. Therefore, over-fitting can be a serious problem when evaluating these programs.

Afterwards, the performance indicators are averaged over all validation processes. There is no perfect indicator for every subject concerning evaluation of machine learning algorithms, since everyone has its flaws and advantages. The most important factors for evaluating the performance of a machine learning program are the following: miss-classification rate, bench-marking, the precision

value, the recall value, the F-Measure and the confusion matrix.

Instead, the machine learning system is assigned a task, and given a large amount of data to be used as an example of how this task can be achieved or from where to detect patterns. The system then learns the best way to achieve the desired output. This can be considered a narrow AI: machine learning supports intelligent systems, which are able to learn certain functions, are given a special data set to study. However, machine learning is not a simple process.

Machine learning by using various algorithms that iterative are able to learn, improve, describe and predict the data results accurately. When the algorithm digests training data, it is possible to produce a more appropriate model based on that data. The machine learning model is the output produced when you practice algorithms learning your machine with data. After training, when you provide a model with input, you will be given output. For example, a predictive algorithm will create a predictive model. Then, when you provide a predictive model with data, you will receive predictions based on the data that trains the model. Machine learning is now important for creating analytical models.

Machine learning allows models to train on data sets before deploying. Some online machine learning models and continues to adapt when new data is digested. On the other hand, another model, called the offline machine learning model, comes from the machine learning algorithm but, after use, does not change. The recurring process of this online model leads to an increase in the types of associations made between data elements. Because of its complexity and size, these patterns and associations can be easily ignored by human observations. After the model is trained, this model can be used in real time to learn from the data.

The model of machine itself can be obtained on the speech area application and data classification. Speech is the most natural ways to interact with each other to acquire information. Sometimes people are so convenient with speech that we would like to interact with our devices (gadget, computer, etc) via speech rather than having to traditional interfaces such as keyboard and pointing devices. Speech recognition is a problem in the field of study in pattern recognition which estimates the pattern matching for doing identification of various objects including part of speech.

Actually, in speech recognition there were so many paradigms according to machine learning process. The paradigms presented and elaborated include: generative and discriminative learning such as: supervised, unsupervised, semi-supervised and active learning (Bayesian learning). These learning paradigms are motivated and discussed in the context of automated speech recognition (ASR) technology and applications. We finally present and analyze recent development paradigms in machine learning with deep learning approach methodology for direct relevance to advancing ASR technology.

Recognizing voices automatically is useful for several applications. For example, it supports biometric authentication for security-relevant services like telebanking. This corresponds to the task of speaker verification]: a model is trained for the voice of each authorized person a priory (the training phase), and when a speaker demands access to the secured service, his voice is compared with the model corresponding to his additionally given identity claim (the evaluation- or test

phase). Based on the similarity of the current voice to the claimed model, access is granted or the speaker is rejected.

Automatic voice recognition also helps to make automatic speech recognition robust by adapting learned speech models to a certain speaker. Therefore, all potential speakers are enrolled in the system (i.e. models of their voices are trained), and in the evaluation phase, the current speaker's voice is compared with all enrolled models. The identity of the model's speaker being most similar to the current voice is returned in this speaker identification scenario if the similarity is not below a certain threshold.

Last, automatic voice recognition enables search engines to index spoken documents and thus improves retrieval performance and surveillance. To this end, first, all speech segments of individual speakers in the audio document have to be identified and segregated from each other and the non-speech content. Then, the number of distinguished speakers and their respective segments has to be identified simultaneously through speaker clustering i.e. grouping together the most similar segments until a certain threshold is reached. The complete process of generating this "who spoke when" index over time of the complete audio document, including the removal of non-speech content, is known as speaker diarization.

Speech is non-stationary signal where properties change quite rapidly over time. This is fully natural and nice thing but makes the use of DFT or auto-correlation as such impossible. For most phonemes the properties of the speech remain invariant for a short period of time (5-100 ms). Thus for a short window of time, traditional signal processing methods can be applied relatively successfully.

At present we have seen ASR applications that demonstrate impressive performance as a result of many major advances in signal processing, statistical modeling and machine learning algorithms including Linear prediction (LP) analysis, Fourier and Filter-bank analysis, hidden markov model hidden markov model (HMM), Gaussian mixture models (GMM), Phonetic decision tree (PDT), N-gram language models, and Deep neural networks (DNN), just to name a few, and various ASR model training techniques, including maximum likelihood and discriminative training.

The main goal of a speech recognition system is to substitute for a human listener, although it is very difficult for an artificial system to achieve the flexibility offered by human ear and human brain. Thus, speech recognition systems need to have some constraints. For instance, number of words is a constraint for a word-based recognition's system. In order to increase the performance of the recognition, the process is dealt with in parts, and researches are concentrated on those parts, this approach of splitting the process into parts provide better performance achievement for each of the parts, thus resulting in increased overall performance. Speech recognition system can be separated in different classes by describing what type of utterances they can recognize.

Isolated word recognizes attain usually require each utterance to have quiet on both side of sample windows. It accepts single words or a single utterance at a time. This is having "Listen and Non Listen state". Isolated utterance might be better name of this class.

Continuous speech recognizer allows user to speak almost naturally, while the computer deter-

mine the content. Recognizer with continues speech capabilities are some of the most difficult to create because they utilize special method to determine utterance boundaries (Keller, 1994). Continuous Speech Recognition system can be divided into two groups as Connected Word Recognition, and Conversational Speech Recognition. The former aims at performing the recognition word by word, however, the latter aims at understanding the meaning of the sentence. Therefore, Conversational Speech Recognition systems are also called Speech Understanding systems, and they require the use of complex grammar rules in the system.

Recognition accuracy of the first one is very high because the system is free from negative side effects of co-articulation. However, for continuous speech recognition, transition effects between words again cause problems. Moreover, for a Word-based recognition system, processing time and memory requirements are very high because there are many words in a language which are the bases of the reference patterns. In a phoneme-based system, while recognition accuracy decreases, it is possible to apply error correction using the ability to produce fast results with very few phoneme numbers.

Most of speech processing in fact is done in this way: by taking short windows (overlapping possibly) and processing them. The short window of signal like this is called frame. In implementation view the windowing corresponds to what is understood in filter design as window-method: a long signal (of speech for instance or ideal impulse response) is multiplied with a window function of finite length, giving finite length weighted (usually) version of the original signal.

State of the art speaker recognition systems mostly use vocal tract related speaker information represented by the spectral or cepstral features like linear prediction cepstral coefficients (LPCC) or mel frequency cepstral coefficients (MFCC). These features provide good recognition performance. They nearly represent complete vocal tract information i.e LPCC or MFCC captures the formants and their bandwidth information characterizing the vocal tract completely, but pitch is only one aspect of speaker information due to source

Windowing techniques are mainly used in the process of designing digital filters. In order to convert an impulse response of infinite duration to a Finite Impulse Response (FIR) filter design windowing is performed. Symmetrical sequences of Window functions generated for digital filter design. Those window functions are usually an odd length with a single maximum at the center. Hamming window technique is used to optimize the window to minimize the maximum (nearest) side lobe, giving it a height of about one-fifth that of the Hann window .

The core of signal processing is Fourier analysis, and the core of Fourier analysis is a simple but somewhat surprising fact: Any periodically-repeating waveform can be expressed as a sum of sinusoids, each scaled and shifted in time by appropriate constants. Moreover, the only sinusoids required are those whose frequency is an integer multiple of the fundamental frequency of the periodic sequence. These sinusoids are called the harmonics of the fundamental frequency.

Finding the Fourier series representation is called Fourier analysis; the converse, Fourier synthesis, consists of converting a set of Fourier coefficients into a waveform by explicitly calculating and summing up all the harmonics. A waveform that is created by Fourier synthesis will yield the

exact same parameters on a subsequent Fourier analysis, and the two representations the waveform as a function of time, or the Fourier coefficients as a function of frequency, may be regarded as equally valid descriptions of the function, i.e. together they form a transform pair, one in the time domain, and the other in the frequency, or Fourier domain.

When analyzing an FFT , the target of interest is the peaks that appear. These peaks occur at locations where the corresponding frequency is dominant in the audio sample. Some audio samples are cleaner and easier to identify peaks than others. Consider an audio sample of an instrument playing a single note and compare this to an audio sample of someone speaking. The voice sample is obviously more complex. So the FFT of the voice will have much more going on with many peaks. There will be many more non-zero coefficients.

Once there is a nice window to view the FFT, extracting the dominant peaks is the next step. Since different audio samples have different amplitudes, it can be helpful to normalize the data and declare a threshold with which to extract the peaks. Since the values are all positive, divide all points by the largest point in the set which results in all amplitude values to be between 0 and 1.

The signal is plotted against the Mel spectrum to mimic human hearing. Each pitch of a pure tone with an actual measured frequency f measured in Hz, a subjective pitch is measured on a scale called the 'Mel Scale'. Mel Filter Bank filters an input power spectrum through a bank of number of Mel-filters. The output is an array of filtered values, typically called Mel spectrum, each corresponding to the result of filtering the input spectrum through an individual filter[82].

After that, the signal will process to discrete cosine transforms (DCT) based speech compression is used to reduce the size of the speech information. It is used to speed up the system by removing the redundancy from audio information. Compression is the process of elimination of redundancy and duplicity. The DCT is very common when encoding video and speech tracks on computers. The DCT is very similar to the DFT but the output values of DCT are real numbers and the output vector is approximately twice as long as the DFT output. It shows a sequence of finite data points in terms of sum of cosine functions.

A non-parametric classification of English phonemes in speaker-independent continuous speech investigates by researcher in 2009[39]. The system employs a powerful and intuitive non-parametric classifier. The recognition result shows a promising increase in the percentage of correctness over the conventional HMM based phoneme recognition. In addition, applying the approximate nearest neighbour approach for the classification purpose rather than the exact one, leads to achieving a very lower training execution time compared to the HMM-based system, and also a comparable execution time for the testing. The outcome was a considerable reduction in the k-NN search space and hence the execution time, and also a slight increase in the recognition performance.

The speech feature extraction is used to reduce the dimensionality of the input vector while maintaining the discriminating power of the signal from fundamental formation of speaker identification and verification system. The number of training and test vector needed for the classification problem grows with the dimension of the given input. Hence we need feature extraction of speech signal.

A fast speaker adaptation technique dedicated to automatic speech recognition systems using artificial neural networks (ANNs) for hidden Markov models (HMMs) state probability estimation presents by[30]. With only 20 words of adaptation data, results show a 25% relative decrease of the word error rate over the speaker independent system, and a 15% decrease over the standard affine transformation adaptation approach.

an effective method for speaker identification system. Based on the wavelet transform, the input speech signal is decomposed into several frequency bands, and then the linear predictive cepstral coefficients (LPCC) of each band are calculated. In this study, the effective and robust LPCC features were used as the front end of a speaker identification system. In order to effectively utilize these multi band speech features, a multi-band 2-stage Vector Quantization (VQ) was proposed as the recognition model. Different 2-stage VQ classifiers were applied independently to each band, and then errors of all 2-stage VQ classifiers were combined to yield total error. The experimental results show that the proposed method is more effective and robust than the baseline models proposed previously[1].

Mel Frequency Scale Cepstral is based on signal decomposition with the help of a filter bank, which uses the Mel scale expressed on the Mel-frequency scale. The MFCC is the result of a discrete cosine transform of the real logarithm of the short-term energy. Mel scale cepstral analysis is very similar to perceptual linear predictive analysis of speech, where the short-term spectrum is modified based on psychophysically based spectral transformations. In this method, the spectrum is warped according to the MEL scale, where as in PLP the spectrum is warped according to the Bark scale. The main difference between Mel scale cepstral analysis and perceptual linear prediction is related to the output cepstral coefficients. The output cepstral coefficients are then computed based on this model. In contrast Mel scale cepstral analysis uses cepstral smoothing to smooth the modified power spectrum.

A key word detection method for continuous speech in noisy environment was proposed in 2010[29]. In the proposed method, the widely used energy, zero crossing, entropy and MFCCs were extracted to generate an audio feature set. Robust endpoint detection algorithm is also used which makes the feature modify its parameter by adapting to the strength of background noise. Then HMMs are used for the classifiers.

Researcher analyzed[62] the voice recognition algorithm based on HMM (Hidden Markov Model) in detail. The feature vector of each voice characteristic parameter is chosen by means of MFCC (Mel Frequency Cepstral Coefficients). The extracting algorithm of syllable parts from continuous voice signal is introduced. It shows the relationship between recognition rates and number of applying syllables and number of groups for applying syllables. The core engine of the HMM method is described, and simple syllables were used for the recognition process. In order to achieve a high recognition rate for different syllables, significant quantitative information of syllables is required. MFCC parameters were used. MFCC with a mel frequency index of 24 provides a higher recognition rate (96% per 72 syllables).

Speaker dependent recognition requires only a mel frequency index of 14 during training in

comparison to the 24 required for speaker independent recognition training. Based on the results of this study, more words can be added frequently to the database. By increasing the number of voice samples being trained, HMM can be widely applied to real life applications and, ultimately, a voice recognition system can be produced.

The current idea for improving the speech recognition system is minimalizing the noise from acoustic signal performance. The impact of noise signal in real time can caused the speech recognition system unpredictable. Acoustic-phonetic approach has been studied in great depth for more than 40 years. This approach is based upon theory of acoustic phonetics and postulates. The earliest approaches to speech recognition were based on finding speech sounds and providing appropriate labels to these sounds[111].

Recently, a number of approaches based on statistical models have been developed, and the corresponding performances in noisy environments are more favourable than those of early methods for which only magic numbers or a portion of rules are used. Typical examples of these newer approaches are those for which either the likelihood-ratio test (LRT), the hidden Markov model (HMM), or the support vector machine (SVM) are used. However, these approaches are still problematic because burst clippings, the short frames of false positives and false negatives, are often caused. The burst clippings can be removed, however, by applying rule based post-processing to the output decisions of these methods; but, although rule-based post-processing is simple, it is unreliable in a non-stationary, noisy environment and is insufficient for real-time applications. To eliminate burst clippings, researcher devised a HMM-based hangover method for which the state-duration effect of the HMM is utilized. However, its limitation is that a little success in avoiding burst clippings relies on the HMM whose weakness is in modeling state duration[23].

In this thesis, we address research questions at each level of the spoken language understanding pipeline. We show that we can improve individual system components using approaches based on neural networks. Deep belief networks (DBNs) have driven tremendous progress in computer vision in recent years. DBNs offer a powerful approach to learning increasingly complex functions from large datasets.

Computer vision saw rapid success for DBNs as many computer vision tasks easily translate to classification and regression problems. Deep learning is a machine learning technique that uses hierarchical neural networks to learn from a combination of unsupervised and supervised algorithms. Deep learning is a specific method of machine learning that incorporates neural networks in successive layers in order to learn from data in an iterative manner. Typically, deep learning learns from unlabeled and unstructured data. While deep learning is very similar to a traditional neural network, it will have many more hidden layer.

Some researcher focus on this works called Elman Network with a feedback layer which only connects to the hidden layer and they proposed the structural optimization to find the optimum characteristic parameters stated by Delgado et al [9]. After training the binary data in RBM, data from previous process can be used for training another model of first RBM significantly in hidden units this process can be repeated as much as desired for creating many layers of non-linear feature

detectors and more complex data. The RBM stack can be associated in multi-layer generative model is called deep belief net [10].

The contribution of this study are:

- We claimed the performance of the system using DBN with the Taboo Search Optimization are better.
- In addition, the novelty of this work being the investigation on "Deep Learning" approach to general voice database speech recognition on top of acoustic modeling.

1.2 Description of content

We present the following description in the chapters of this dissertation:

- Chapter 2 Provides general works of Neural Network, The development of machine learning and the fundamental theory of speech recognition techniques. Historical information is provided, and the basic properties of human speech production and perception are explained. We discuss the main main component to build the speech recognition system. We also discuss the techniques of feature extraction from waveform signal spoken into digital signal.
- Chapter 3 This chapter presents, details investigates of pattern recognition which is widely used in many application areas. We evaluated this technique experimentally using MNIST and OWN database and the proposed approaches were compared to the conventional Deep Belief Network.
- Chapter 4 Presents the experimental result and discussion on the use of different feature and classification methods in speech signal with mathematical model. By the mathematical model we shows the process from raw speech signal extracted into feature system then they will process into optimization model.
- Chapter 5 Finally, we are concluding the remarks of dissertation in this section with the summary contribution and suggestion for continuing research in and application of speech recognition.

2

Common Techniques and Related Works

2.1 Neural Network

2.1.1 An Overview of Neural Network

Neural Network is a category of Soft Computing science. Neural Networks actually adopt from the ability of the human brain that is able to provide stimulation process and output. The Output is obtained from variations in stimulation and processes that occur in the human brain. The function of neural network in general application can be applied on pattern recognition, optimizer the problem, speech recognition and classification[72].

The history of the development of Neural Network has been found since 1943 when Warren McCulloch and Walter Pitts introduced the first time the neural network model[32]. They do a combination of several simple processing units together which can provide an overall increase in computing power. In 1958, Rosenblatt continues the work and began developing a network model called perceptron. The training method allows for certain learning classification jobs with added weights on each inter-network connection[31].

The perceptron methods to classify of pattern is not entirely perfect, there are still some limitations on it. this method is unable to solve XOR (exclusive-OR) problems. However, the perceptron succeeded in becoming a basis for further research in the neural network. The study of neural networks began to develop again in the early 1980s. Researchers have discovered many fields of new interest in the domain of neural networks. Recent studies include Boltzmann machines, Hop-

field networks, competitive learning models, multilayer networks, and adaptive resonance model theory.

The basic idea of neural network starts from the human brain , where the brain contains about 10^{11} neurons. This neuron functions to process every incoming information. One neuron has 1 axon, and at least 1 dendrite. Each nerve cell is connected to another nerve, the number reaches around 10^4 synapses. Each cell interacts with each other which produces certain abilities in the work of the human brain. The human brain works can be illustrated on figure 1.1

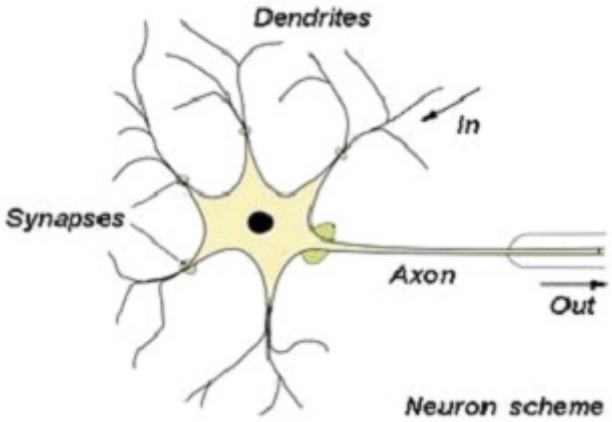

Figure 2.1.1: Biological Neuron Structural of Human Brain[34]

From the figure 2.1.1, it shown the biological neuron structural of human brain consists of *dendrites*, *axon* , and *synapses* . The function of dendrites is stimulating the impulse to receive from nerve cell body, the axon transferring the signal impulse to another part of body nerve cell. Lastly, synapses function serving as functional unit between two nerve cells[85].

2.1.2 Artificial Neural Network

The branch of artificial intelligence is quite broad, and closely related to other disciplines. This can be seen from various applications which are the result of a combination of various sciences. As well as those in medical devices in the form of applications. It has been developed that the application created is the result of a combination of artificial intelligence and medical science or more specifically in biology[67].

An artificial neural network is an information processing system that has almost the same display character as a neural network in biology. Artificial neural networks have been developed as generalizations mathematical models from biological neural networks, based on assumptions; Processing information, the *signal* is passed between the neurons via a connecting link, each connection link has a *weight* where the specific neural network is multiplied by the transmitted signal, each neuron uses the *activation function* (nonlinear) on its input network (summing the input signal and weight)

to determine the output signal[4].

The definition of artificial neural network is similar as a machine designed to pattern the way in which the brain performs a particular function. The networks are usually implemented using electronic components or simulated in a software on a digital computer. To achieve a good view, artificial neural networks use very large interconnections between computational cells called "neurons" or "processing units". As an adaptive machine, an artificial neural network is a large, parallel distributed processor composed of simple processing units that have a tendency to store experience and knowledge and make it ready for use in generally[38].

An artificial neural network is usually trained in two methods. The most common is supervised training. Each example in this training completely specifies all inputs as desired output when input is represented. Then we choose a subset of training and represent examples in a subset on the network at the same time. For each example, we compare the output produced by the network with the output we expect to produce. After all training subsets have been processed, we update the weights that connect neurons in the network. This updated model is done in the hope of reducing errors in network[92].

Another training method is unsupervised training. As with guided training, we must also enter sample inputs. But it does not provide target output for the network. It is assumed that each input comes from a different class, and network output is the identification of the class to which the input originates. The process of training is to find features that stand out in training and use them to classify input into classes and find differences. Unsupervised training is usually not used as popular as guided training. The third is the hybrid training method. A combination of guided and unsupervised training. Not guided because the target output is not specified. Called guided because at the same time, the network responds to training where the response is good or bad[53].

It is very difficult to train a network and use it immediately. The competence must be tested first. The process of testing a training is called *validation*. Training is used to train networks while validation is used to test networks that have been trained. Validation cannot be underestimated. In many fields, good *validation* is more important than good *training*.

The other opinion of understanding artificial intelligence are stated that the ability of humans to process information is the result of the complexity of the process in the brain[51]. For example, what happens to children, they are able to learn to do recognition even though they do not know what algorithm is used. The extraordinary computing power of the human brain is an advantage in the study of science. A simple artificial neural network was first introduced[28]. This study concluded that the combination of several simple neurons into a neural system would increase their computational ability. The weight in the network proposed in this theory is set to perform simple logic function. The activation function used is the *threshold function*.

The learning rule (with the reason that if 2 neurons were active at the same time, their strength would increase) was first developed in 1949[83]. Then between 1950-1960s some researchers experienced developments in observations about perceptron. The *perceptron* was introduced and began developing a network model in 1958[13]. The training method was introduced to optimize

the results of the computer. In 1960 the researcher develops the perspective by introducing network *training* rules, known as delta rules (smallest average squares)[104]. This rule will change the weight of the *perceptron* when the output produced does not match the desired target.

In 1986 the *perceptron* developed to be back-propagation, which allows networks to be processed through multiple screens[109]. In addition, several other artificial neural network models were also developed. The development that was widely discussed since 1990 was the application of artificial neural network models to solve various problems in the real world. Neural network architecture is a reflection of an artificial neural network[54].

The definition of Artificial Neural Network (ANN) can be concluded as an information processing system that has characteristics resembling to human biological neural networks. ANN is a generalized mathematical model of human cognition which includes input, output patterns and the network will be taught to provide possible prediction. Basically, ANN characteristics are determined by the pattern of relationships between neurons (it is called network architecture), the method for determining of weight in each input and activation function. The architecture of Neural Network can be seen as:

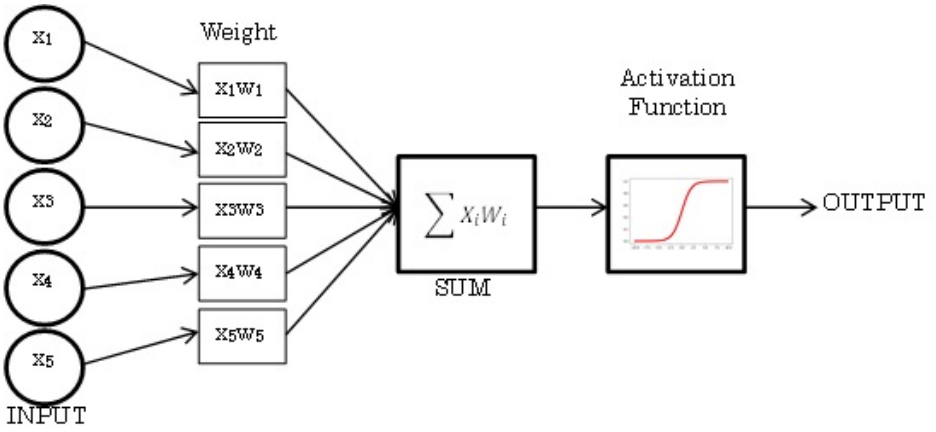

Figure 2.1.2: Neural Network Architecture

In artificial neural network model, numerical values are used as inputs to the "dendrites". Each input is multiplied by a value called weight, which simulates a real dendrite response. All results from *"dendrites"* are added and threshold in *"synapses"*[14]. Finally, the thresholds results are sent to *"dendrites"* from other neurons through *"axon"*. This sequence of events in the single neuron can be expressed in mathematical formulae as :

$$y = f\sum_{i=1}^{n} X_i.W_i \tag{2.1}$$

where X_i is the input received by a dendrite, W_i the weight associated to the i^{th} "dendrite," $f()$ a threshold activation function and y the output of the neuron.

In this network, several dendrites (x) are directly related to the output layer (y). Each input is connected to the weight (w) and produces a different output depending on the input. During the learning process, weights will be modified based on certain rules to produce precise accuracy. This model is very suitable for pattern recognition techniques seen from the level of simplicity[44].

The key factor for determining the behavior of a neuron is its activation function and connection patterns with other neurons so that neurons can send and receive signals. More specifically, in many artificial neural networks, neurons in a layer can be fully connected or not connected at all. If each neuron in a layer (ie *hidden layer*) is connected to a neuron in another layer (ie *output layer*) then each hidden unit is connected to each output unit[45].

The arrangement of neurons in layers and their connection patterns within and between layers is called network architecture. Many networks have an input layer whose activation of each unit is the same as the external input signal. Neural networks are classified as *single layers* and *multilayers*. In determining the number of layers, the input unit is not counted as a layer because the unit does not carry out the computing process. Or it could be said that the number of layers in the network is determined based on the layer that contains the weight between the connections of a collection of neurons. This is what underlies that the *weight* on the neural network contains very important information[75].

A layer is a set of *neurons* that share the same input. Each neuron in a layer has a *dendrite* that is connected to the of the neuron in the previous layer. The first layer is the input layer where neurons do not have *dendrite*. These neurons are only *placeholders* or supports so that the next layer can tap the input values the same as the next layer. The last layer is the output layer. The layer between the input layer and the output layer is called *hidden layer*. The first layer only provides input values on the network. In the next layer, neurons are assigned to identify characters from the input. The types of network architecture that are often used are: Single Layer Network and MultiLayer Network[116].

Generally, when using artificial neural networks, the relationship between input and output must be known with certainty and if the relationship is known then a model can be made or the architecture of a neural network is set by a design process. Artificial neural networks are characterized by inter-neuron connection patterns called architecture, methods of determining weights on each connection (*training* or *learning*, *algorithms*) and their activation functions[76].

Based on the level of ability, artificial neural networks can be applied to several applications that are suitable when applied to the classification of patterns, namely selecting an input data into a particular category that is applied. In addition, artificial neural networks can be applied to predictions and *self-organizing*, which is describing an object as a whole only by knowing part of another object and having the ability to process data without having to have data as a target. Furthermore, artificial neural networks are also able to be applied to the problem of optimization, namely finding the best answer or solution of a problem[105].

The characteristic model of artificial neural networks is their ability to learn. The *learning* on artificial neural networks can be interpreted as the process of adjusting the parameters of the weighting because the desired output depends on the price of the interconnect weight owned by the cell. The *learning* process will be stopped if the error or error value is considered small enough for all pairs of *learning* data. The network that is doing the *learning* process is called in the *learning* phase. In the early stages of this *learning*, it needs to be done first before testing an object[107].

2.1.2.1 Self-Organizing Map

Self Organizing Map (SOM) is a grouping method in the form of two-dimensional topography such as a map to facilitate observation of the results of grouping. SOM requires determining the learning rate, learning function, number of iterations desired in the grouping process. The Self Organizing Map method does not require objective functions such as K-Means and Fuzzy C-Means for optimal conditions in an iteration, the SOM will not stop the iteration as long as the specified number of iterations has not been reached[52].

SOM (Self Organizing Feature Map) was first introduced with ANN training techniques that use *a winner takes all basis*. Where only the winning neurons will be renewed in weight. Although using an ANN base, SOM does not use the target class, no class is assigned to each data. Characteristics like this which then make SOM can be used for ANN-based grouping purposes[64].

2.1.2.2 Multi-Layer Perceptron

Before we introduce the multi-layer perceptron , the single layer network has one connected weight layer. In this layer, the input unit can be distinguished from the output unit. Where the input unit is a unit that receives signals from the outside world while the output unit is a unit where the response from the network can be seen. In Figure 2.1.4 it is clear that the input unit is fully connected to the output unit, while the input unit with each input unit is not connected as well as between the output unit and other unconnected output units[101].

The multi-layer network has a characteristic feature of the input input layer , *hidden layer* and *output layer* . Networks with multiple layers can solve a more complex problem compared to a single network. However, the time needed to solve the problem will be longer with careful learning. By adding one or more hidden layers, the network will be able to perform better statistical extraction. The network performance improvement is caused by additional synaptic connection cell assemblies and also by additions dimension range obtained from interactions between neurons. The points of input layer enter each element of each input vector, where these signals will be used further at computational points on the first hidden layer. The output signals from this layer will be the input source for the next layer and so on in the network[47].

In general, each neuron in each layer on the network gets their input from the output of the previous layer. The collection of output signals from each neuron in the output layer represents the overall artificial neural network response to the activation pattern provided by the input points.

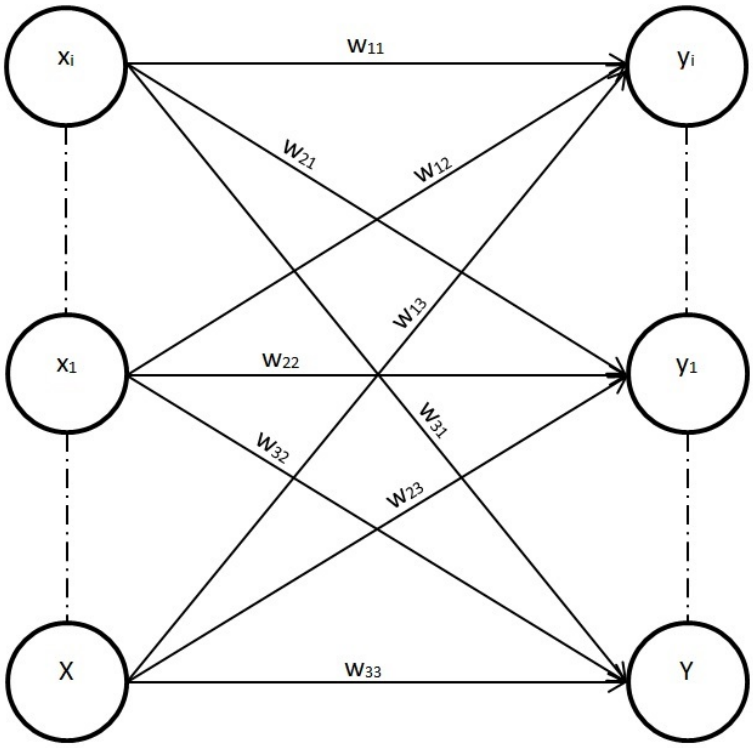

Figure 2.1.3: Single Layer Topology Network

Artificial neural networks in Figure 2.1.5 shown networks that are connected as a whole because each point on each layer is connected to each point in the next layer. But if a synaptical connection is lost, the artificial neural network is partially connected[7].

The techniques of ANN are back-propagation and feed-forward. Back-propagation algorithms for neural networks are generally applied to multi-layer perceptrons. Perceptron has at least the input part, the output part and several layers that are between the input and output. This layer in the middle, which is also known as hidden layers, can be one, two, three etc. In practice, the highest number of hidden layers is three layers. With these three layers almost all problems in the industrial world can be resolved. The last layer output from the hidden layer is directly used as the output of the neural network[108].

Training in back-propagation method involves 3 stages: feed-foward training pattern, error calculation and weight adjustment. After training the network application only uses the first stage of computing, namely feed-foward. Although the training phase is very slow, the network can produce output very quickly. The back-propagation method has been varied and developed to increase the speed of the training process.

Back-propagation was created by generalizing the Widrow-Hoff learning rule to multiple layer

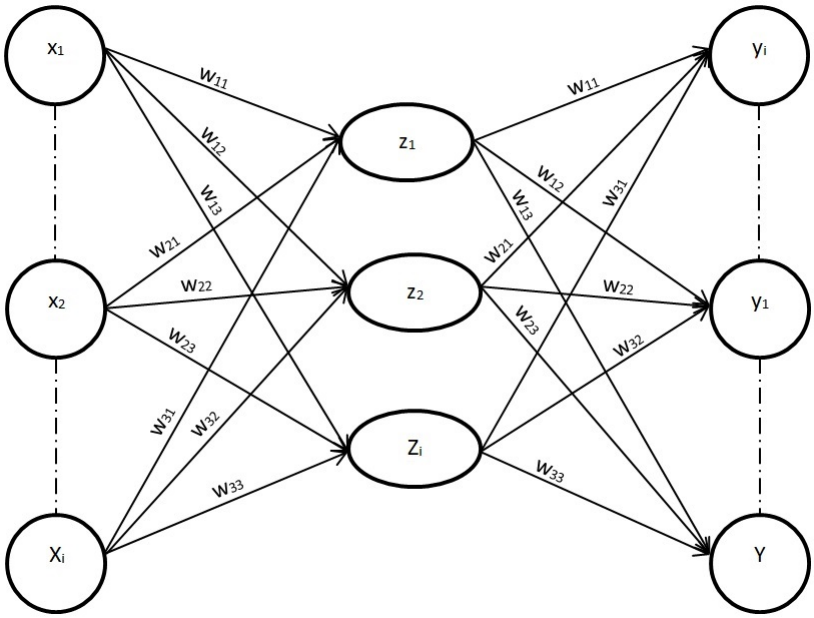

Figure 2.1.4: Multi-Layer Topology Network

networks and nonlinear differentiable transfer functions. Input vectors and the corresponding output vectors are used to train a network until it can approximate a function, associate input vectors with the specific output vectors or classify input vectors in an appropriate was as defined. Standard backp-ropagation is a gradient descent algorithm, as is the widrow-hoff learning rule. The term back-propagation refers to the manner in which gradient is computed for nonlinear multi-layer networks.

Although one network layer is very limited in learning, networks with multiple layers can learn more. More than one hidden layer may be useful for some applications, but one hidden layer is enough.

Feed-forward Neural Network (also known as multi-layer perceptrons) was first introduced in the context of Bengio's word-based language modeling in 2003[6]. The fundamental feature of a Recurrent Neural Network (RNN) is that the network contains at least one feed-back connection, so the activations can flow round in a loop. That enables the networks to do temporal processing and learn sequences, e.g., perform sequence recognition/reproduction or temporal association/prediction.

In the feed-forward step, an input pattern is applied to the input layer and its effect propagates, layer by layer, through the network until an output is produced. The network's actual output value is then compared to the expected output, and an error signal is computed for each of the output nodes[9].

Since all the hidden nodes have, to some degree, contributed to the errors evident in the output layer, the output error signals are transmitted backwards from the output layer to each node in

the hidden layer that immediately contributed to the output layer. This process is then repeated, layer by layer, until each node in the network has received an error signal that describes its relative contribution to the overall error.

Once the error signal for each node has been determined, the errors are then used by the nodes to update the values for each connection weights until the network converges to a state that allows all the training patterns to be encoded. The Back-propagation algorithm looks for the minimum value of the error function in weight space using a technique called the delta rule or gradient descent[66]. The weights that minimize the error function is then considered to be a solution to the learning problem.

2.1.2.3 Recurrent Neural Network

Recurrent neural network architectures can have many different forms. One common type consists of a standard Multi-Layer Perceptron (MLP) plus added loops. These can exploit the powerful non-linear mapping capabilities of the MLP, and also have some form of memory. Others have more uniform structures, potentially with every neuron connected to all the others, and may also have stochastic activation functions[77].

For simple architectures and deterministic activation functions, learning can be achieved using similar gradient descent procedures to those leading to the back-propagation algorithm for feed-forward networks. When the activations are stochastic, simulated annealing approaches may be more appropriate. The following will look at a few of the most important types and features of recurrent networks.

The simplest form of fully recurrent neural network is an MLP with the previous set of hidden unit activations feeding back into the network along with the inputs:

Note that the time i has to be discretized, with the activations updated at each time step. The time scale might correspond to the operation of real neurons, or for artificial systems any time step size appropriate for the given problem can be used. A delay unit needs to be introduced to hold activations until they are processed at the next time step.

The above diagram shows a RNN being unrolled (or unfolded) into a full network. By unrolling we simply mean that we write out the network for the complete sequence. For example, if the sequence we care about is a sentence of 5 words, the network would be unrolled into a 5-layer neural network, one layer for each word. The formulas that govern the computation happening in a RNN are as follows: x_t is the input at time step t. For example, x_1 could be a one-hot vector corresponding to the second word of a sentence. h_t is the hidden state at time step t. It's the "memory" of the network. h_t is calculated based on the previous hidden state and the input at the current step as formula 2.2. The function f usually is a nonlinearity such as tanh or ReLU. h_{t-1}, which is required to calculate the first hidden state, is typically initialized to all zeroes. y_i is the output at step t. For example, if we wanted to predict the next word in a sentence it would be a vector of probabilities across our vocabulary [77].

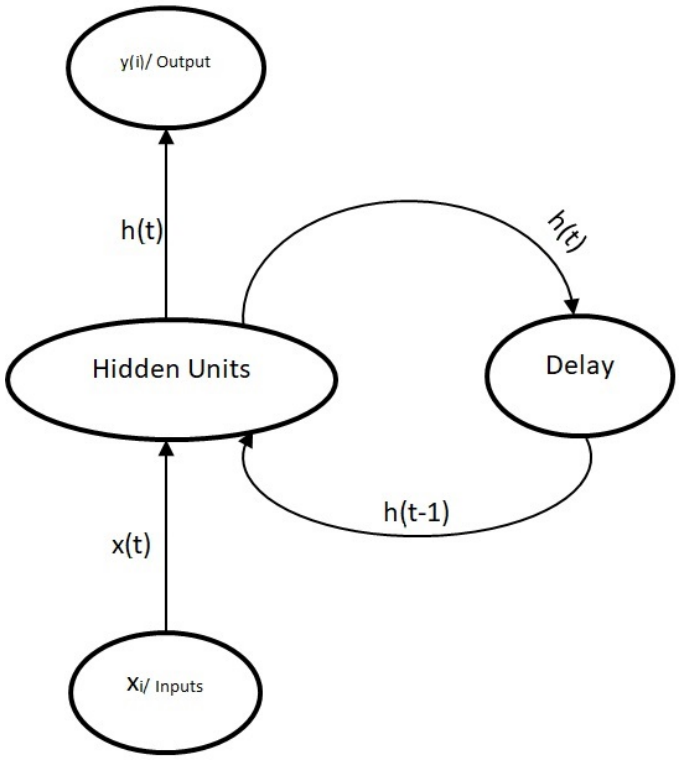

Figure 2.1.5: Recurrent Neural Network Architecture

If the neural network inputs and outputs are the vectors x(i) and y(i), the three connection weight matrices are W_{IH}, W_{HH} and W_{HO}, and the hidden and output unit activation functions are f_H and f_O, the behaviour of the recurrent network can be described as a dynamical system by the pair of non-linear matrix equations:

$$h(t) = f_H\left(W_{IHH}x\left(t\right) + W_{HH}h\left(t-1\right)\right) \tag{2.2}$$

$$y(t) = f_O\left(W_{HO}h\left(t\right)\right) \tag{2.3}$$

In general, the state of a dynamical system is a set of values that summarizes all the information about the past behaviour of the system that is necessary to provide a unique description of its future behaviour, apart from the effect of any external factors. In this case the state is defined by the set of hidden unit activations h(t). Thus, in addition to the input and output spaces, there is also a state space. The order of the dynamical system is the dimensionality of the state space, the number of hidden units.

Since one can think about recurrent networks in terms of their properties as dynamical systems, it is natural to ask about their *Stabilityy*, *Controllability* and Observability: *Stability* concerns the

boundedness over time of the network outputs, and the response of the network outputs to small changes (e.g., to the network inputs or weights). *Controllability* is concerned with whether it is possible to control the dynamic behaviour. A recurrent neural network is said to be controllable if an initial state is steerable to any desired state within a finite number of time steps. Observability is concerned with whether it is possible to observe the results of the control applied. A recurrent network is said to be observable if the state of the network can be determined from a finite set of input/output measurements.

2.1.2.4 Neural Network in Speech Recognition History

Language models are an important component in many word and language processing applications including voice recognition. The purpose of the language model is to estimate the probability of each sentence given as below:

$$P(W) = P(w_0, w_1, w_2, \ldots, w_N) = N \prod_{i=1}^{N} P(w_i | w_{i1}, \ldots w_1, w_0) \tag{2.4}$$

The probability of a word sequence $W = w_0, w_1, w_2, \ldots w_N$ can be decomposed into cascading probabilities using the chain rule. The overall probability can be written as products of conditional probabilities for each word given its history. where N is the valid length of word sequence W, w_0 is always the symbol of sentence start and w_N is always the sentence end symbol. Language models are trained on a set of training corpus to estimate the probability of $P(w_0, w_1, w_2, \ldots, w_N)$. However, for any applications with even a moderate vocabulary size, the number of parameters for this model is prohibitively large and impractical to store and compute all combinations of word w_i and history $(w_{i-1}, \ldots, w_1, w_0)$. Hence, the history $(w_{i-1}, \ldots, w_1, w_0)$ is normally grouped to an equivalent class.

The speaker adaptation techniques was studied for automatic voice recognition systems using artificial neural networks (ANN) in the estimation of hidden Markov estimation models[94]. With the 20 word trial adaptation data, the results show a relative reduction of 25% in the word error rate above the speaker independent system and a 15% decrease in the standard affine transformation adaptation approach.

Text-independent speech recognition using artificial neural networks was conducted an experiment in 1994[94] . Speech data collected from three different speakers said thirteen different words. In the trial of the three speakers the word spoken was repeated ten times. Talk data is then processed before for signal conditioning. A total of 12 feature parameters were obtained from the Cepstral coefficient through Linear Predictive Coding (LPC). This feature parameter then functions as an input to the neural network for the speaker classification. A standard two-layer neural network is trained to identify the different feature sets associated with the appropriate speaker. The network is tested for invisible words remaining in independent text mode. The results are very promising with more than 90% voice recognition accuracy.

The Artificial Neural Network is used as a research tool to achieve the introduction of automatic speech[106]. Small vocabulary containing words YES and NOT selected. Spectral features using cepstral analysis are extracted per frame and imported into advanced feed neural networks, which use backward propagation with momentum training algorithms. Networks are trained to recognize and classify words that fall into each category. The output of the neural network is loaded into the pattern search function, which matches the input sequence with a set of target word patterns. The level of variability in input speech patterns limits vocabulary and affects network reliability.

The results of the first phase of this work are satisfactory and thus the application of artificial neural networks along with cepstral analysis in isolated word recognition is quite promising. This system provides satisfactory results. It's strong enough to take into account the independent input of the speaker. Although the encouraging success of the current system is achieved based on limited vocabulary, the system can be extended to a larger vocabulary by expanding the number of subnets used in architecture. The main solution is to increase the number of features extracted at each frame at the cost of additional processing time.

The theory of combinations system using LPC and Neural Network was proposed an environmental sounds recognition system using for characterization and a backpropagation artificial neural network as a verification method[21]. The verification percentage was 96.66% although the number of feature vectors was small; specifically, two feature vectors were used. The lowest percentages were obtained for noisy sound sources, like car, motorcycles, and airplanes.

In 2012, the researcher explored how a back-propagation neural network (BNN) can be applied to isolated-word speech recognition[20]. Simulation results show that a BNN provides an effective approach for small vocabulary systems. The recognition rate reaches 100% for a 5-word system and 94% for a 10-word system. The general techniques developed can be further extended to other applications, such as sonar target recognition, missile seeking and tracking functions in modern weapon systems, and classification of underwater acoustic signals. The choice of feature vector plays an important role in the performance of the BNN. The recognition rate may decrease drastically or the system may not converge at all if the features are not correctly chosen. The feature vector chosen in the experiments consisted of the LPC coefficients, short time energy, zero-crossing rate and voiced/unvoiced classification. It worked well and provided good results. However, predictions cannot be made about the likely performance of the methods in these areas until they are actually tested.

In their experiment The convolutional neural network (CNN) can be regarded as a variant of the standard neural network[2]. Instead of using fully connected hidden layers, the CNN introduces a special network structure, which consists of alternating it is called convolution and pooling layers. In the pattern recognition using the Convolutional Neural Network (CNN), the input of data need to be organized as a number of feature maps to be fed into the CNN layers. CNNs run a small window over the input image at both training and testing time, so that the weights of the network that looks through this window can learn from various features of the input data regardless of their absolute position within the input.Weight sharing, or to be more precise in our present situation,

full weight sharing refers to the decision to use the same weights at every positioning of the window. CNNs are also often said to be local because the individual units that are computed at a particular positioning of the window depend upon features of the local region of the image that the window currently looks upon.

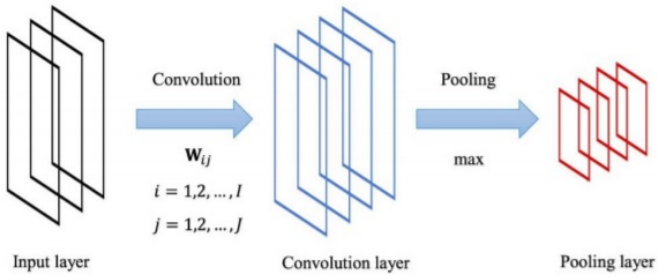

Figure 2.1.6: Convolutional Neural Network Architecture

The mapping can be represented as the well known convolution operation in signal processing. assuming input feature maps are all one dimensional, each unit of one feature map in the convolution ply. A pooling operation is applied to the convolution ply to generate its corresponding pooling ply. The pooling ply is also organized into feature maps, and it has the same number of feature maps as the number of feature maps in its convolution ply, but each map is smaller.

The purpose of the pooling ply is to reduce the resolution of feature maps. This means that the units of this ply will serve as generalizations over the features of the lower convolution ply, and, because these generalizations will again be spatially localized in frequency, they will also be invariant to small variations in location. This reduction is achieved by applying a pooling function to several units in a local region of a size determined by a parameter called pooling size. It is usually a simple function such as maximization or averaging. The pooling function is applied to each convolution feature map independently.

2.2 A Machine Learning Development

Machine learning is a collection of algorithm that would allow a computer to discover automatically or semi automatically the relation between input data and expected output. A machine learning algorithm can be used to predict a value output based on a given input[95].

Machine learning allows for learning of obvious and subtle relations among elements of given data[61]. The learned relations are called models, and can be used to predict missing data, to group data in clusters, or to identify the class of an instance. A model has the ability to represent objects, properties of objects, and relations between objects. This is especially in data mining where people want to have models of complex relations shown in easy understand diagrams or sentences like output of rules. These models can be updated automatically whenever there is need for such updates[5].

Machine learning can be accomplished in a supervised or an unsupervised way. In supervised learning, the system receives a dataset with different example parameter values and decisions /classification, from which it infers a mathematical function, which automatically maps an input signal to an output signal. So, it figures out what it is supposed to do. The whole processes of supervised learning shown in Figure 2.2.1.

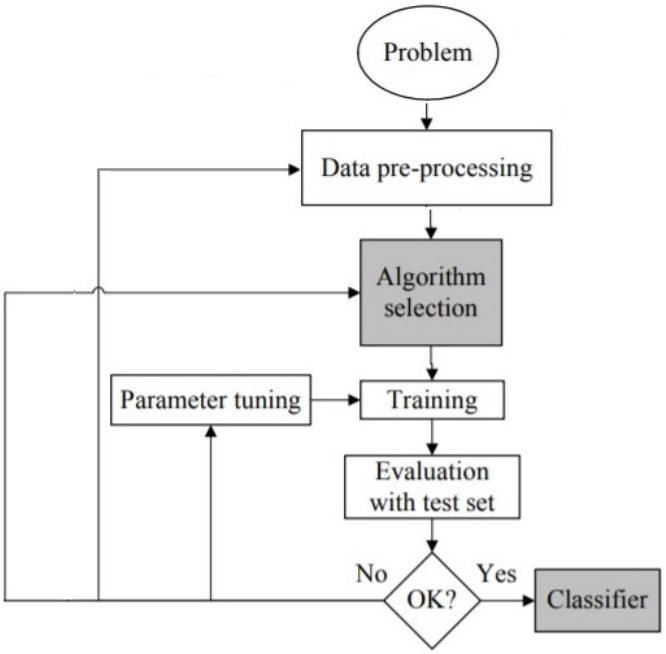

Figure 2.2.1: Convolutional Neural Network Architecture[99]

On the figure 2.2.1 above, the first step in generalize of the supervised learning process is the data preparation and data pre-processing. Depending on the circumstances, researchers have a number of methods to choose from to handle missing data. Hodge Austin have recently introduced a survey of contemporary techniques for outlier (noise) detection. These researchers have identified the techniques advantages and disadvantages.

Instance selection is not only used to handle noise but to cope with the in-feasibility of learning from very large datasets. Instance selection in these datasets is an optimization problem that attempts to maintain the mining quality while minimizing the sample size. It reduces data and enables a data mining algorithm to function and work effectively with very large datasets. There is a variety of procedures for sampling instances from a large dataset.

The choice of feature subset is the identification process and removing as many irrelevant and redundant features as possible. In theory it is explained to reduce data dimensions and allow data mining algorithms to operate faster and more effectively. The fact that many features depend on each other often greatly affects the accuracy of the supervised ML classification model. This prob-

lem can be overcome by building new features from basic feature sets. This technique is called feature / transformation construction. These newly generated features can lead to more concise and accurate classifications. In addition, the feature contributes to a better understanding of the resulting classifier, and a better understanding of the concepts learned.

After going through pre-processing data, then the next step is Algorithm Selection for learning algorithms must be used is an important step. After initial testing is considered satisfactory, classifiers (mapping from examples that are not labeled to class) are available for routine use. Classifying evaluations are most often based on predictive accuracy (the percentage of correct predictions divided by the total number of predictions). There are at least three techniques used to calculate the accuracy of classifiers. One technique is to divide the training set by using two thirds for training and the third for estimating performance. In another technique, known as cross validation, the training device is divided into several equal and equal subsets and for each subset the classifiers are trained on the unification of all other subsets. The average error rate of each subset is therefore an estimate of the error rate of the classifier. Validation of leave one is a special case of cross validation. All test subsets consist of one instance. This type of validation is, of course, more computationally expensive, but is useful when the most accurate estimation of the level of error is classified.

A common method for comparing supervised ML algorithms is to perform statistical comparisons of the accuracy's of trained classifiers on specific datasets. If we have sufficient supply of data, we can sample a number of training sets of size the data, run the two learning algorithms on each of them, and estimate the difference in accuracy for each pair of classifiers on a large test set. The average of these differences is an estimate of the expected difference in generalization error across all possible training sets of size data, and their variance is an estimate of the variance of the classifier in the total set.

Unsupervised learning is a group of Machine Learning algorithms and approaches that work with this kind of "no-ground-truth" data. This is pure 'learning by doing' or trial-and-error. Compared to supervised learning, unsupervised methods perform poorly in the beginning, when they are un-tuned, but as they tune themselves, performance increases. It can be argued that using unsupervised learning, a classifying system should be able to set up hypotheses that no human can figure out, due to their complexity.

Since the introduction of unsupervised pre-training, many new schemes for stacking layers of features to build "deep" representations have been proposed. Most have focused on creating new training algorithms to build single-layer models that are composed to build deeper structures. considered in the literature are sparse-coding, RBMs, sparse RBMs, sparse auto encoders, denoising auto-encoders, "factored" and mean-covariance RBMs, as well as many others. Thus, amongst the many components of feature learning architectures, the unsupervised learning module appears to be the most heavily scrutinized.

Some work, however, has considered the impact of other choices in these feature learning systems, especially the choice of network architecture. For instance, have considered the impact of changes to the "pooling" strategies frequently employed between layers of features, as well as dif-

ferent forms of normalization and rectification between layers. Similarly, in their experiment have considered the impact of coding strategies and different types of pooling, both in practice and in theory. Our work follows in this vein, but considers instead the structure of single layer networks before pooling, and orthogonal to the choice of algorithm or coding scheme.

Many common threads from the computer vision literature also relate to our work and to feature learning more broadly. For instance, we will use the K-means clustering algorithm as an alternative unsupervised learning module. K-means has been used less widely in "deep learning" work but has enjoyed wide adoption in computer vision for building code-books of "visual words", which are used to define higher level image features. This method has also been applied recursively to build multiple layers of features. The effects of pooling and choice of activation function or coding scheme have similarly been studied for these models. In their study, for instance, demonstrate that "soft" activation functions ("kernels") tend to work better than the hard assignment typically used with visual words models.

To evaluate classifier performance given by a machine learning scheme, either a special testing set or a cross validation technique may be employed. A test set contains pre-classified examples different to those in the training set, and is used only for evaluation, not for training. If data are scarce, it is sensible to use cross validation in order not to waste any data, which could be useful to enhance classifier performance; all data are used both for training the classifier and for testing its performance. More examples does not necessarily mean better classifier performance. Even though the classifier becomes better on the training set it could actually perform worse on the test data. This is due to the over-fitting of the classifier transfer function, so that it its too tightly to the training data and the border between classes is jagged rather than smooth, unlike how it usually should be.

Machine Learning is a scientific discipline that focuses on the problem of making accurate predictions for a given set of data based on the past observations, this section describes machine learning techniques that are widely used in different domains, among them there are Naive Bayes classifier, Support Vector machines, Neural Network, Deep Learning, Tagging, Parsing and Extraction, all of them are used for statistical classification of the given data.

There are several techniques of machine learning such as Naive Bayes Classifier, Support Vector Machine (SVM),Neural Network that can be approached with deep belief network. Naive Bayes Classifier (NBC) is a probabilistic classifier that is based on Bayes's theorem with naive independent assumptions. The independence among assumptions implies that all attributes from a given training data are independent from each other. Furthermore, it requires a small amount of training data for the classification. In addition, the following advantages of Bayesian classifier: "simplicity, learning speed, and classification speed and storage space". Naive Bayes classifier should be preferred when the dataset size is small. Also, the model of NBC is robust and self-correcting, i.e. when there are changes in data, the same happens with the result.

2.2.1 A brief History of Deep Belief Network

Historically , the concept of in deep learning came from the research of artificial neural networks. MLP's neural network with many hidden layers is indeed a good example of a model with deep architecture. Back propagation, which was discovered in the 1980s, has become a well-known algorithm for studying the weight of this network. Unfortunately back propagation alone does not function well in practice to study networks with more than a small number of hidden layers (see interesting reviews and analyzes in[24]. The presence of local optima that permeates the non-objective function convex from deep tissue is the main source of difficulty in learning. Refinement is based on local descent gradients, and usually starts at some random starting point. This is often trapped in a bad local optima, and the severity increases significantly when the network depth increases.

DBN[19] is a type of DNN that performs data classification or data prediction with high-precision by performing feature extraction by using a network in which a plurality of Restricted Boltzmann Machines (RBM) are concatenated. RBM is a type of graphical un-directed energy graph model. An un-directed graph is graph, i.e., a set of objects (called vertices or nodes) that are connected together, where all the edges are bidirectional. An un-directed graph is sometimes called an un-directed network. In contrast, a graph where the edges point in a direction is called a directed graph. When drawing a un-directed graph, the edges are usually described as lines between pairs of vertices, as illustrated in the following figure 2.2.2.

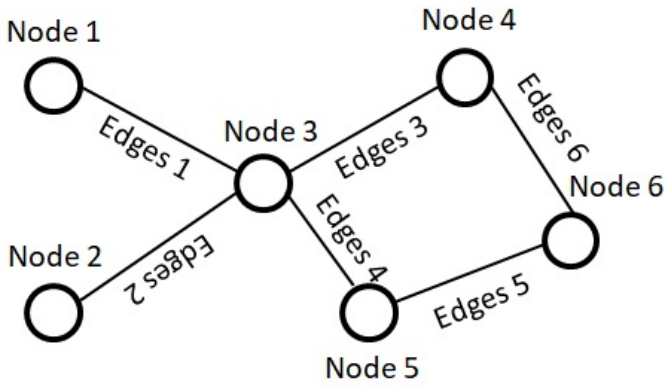

Figure 2.2.2: Un-directed Simple Graph Model

As illustrated figure 2.2.3 for RBM a pair (bipartite graph) of nodes from each of the two groups of units (usually referred to as "visible" and "hidden" units respectively) processing in simple way, who have symmetrical connections between them (node 1 and node 2 as a hidden layer and node 3-5 as a visible layer); and there is no connection between nodes in the group. In contrast, the "restricted" Boltzmann machine may have connections between hidden units. This limitation allows for training algorithms to be more efficient than those available for the general class of Boltzmann

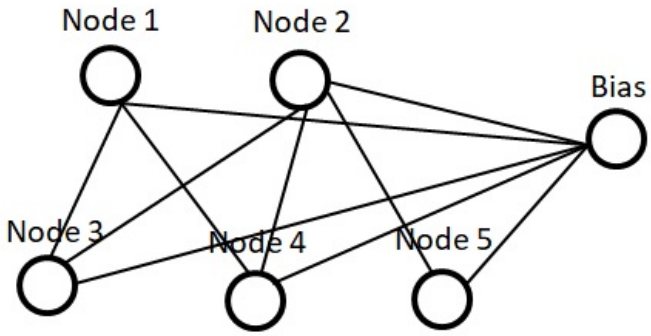

Figure 2.2.3: Undirected Simple Graph Model

machines, specifically the gradient divergence algorithm[65].

A Boltzmann machine is formed from Markov Random Field (MRF) with several nodes in the graph called hidden variables that can be indirectly observed but contribute to the joint distribution of the model. The vertices of the graph are then divided into "visible" and "hidden" variables. The visible node is usually the only node that we are interested in directly model hidden nodes and the dependencies they represent, and the easiest way to find the value of the node that is visible is to find marginal above the hidden node[27]. Using the Gibbs distribution formulae can be defined by:

$$p(v) = \sum_{v,h}^{i} p(v,h) = \frac{1}{z} \sum_{v,h}^{i} e^{-E(v,h)} \tag{2.5}$$

where Z can be defined:

$$z = \sum_{v,h}^{i} exp^{-E(v,h)} \tag{2.6}$$

Then, the maximum likelihood learning algorithm can train the network by simply alternating between updating all the hidden units in parallel and all the visible units in parallel:

$$\frac{\partial logP(v)}{\partial W_{ij}} = \langle v_i h_j \rangle_{\circ} - \langle v_i h_j \rangle_{\sim} \tag{2.7}$$

the energy function of distribution can be calculated as a Bernoulli (visible)-Bernoulli (hidden) in RBM:

$$E(v,h;\theta) = -\sum_{i=1}^{I}\sum_{j=1}^{J} w_{ij} v_i h_j - \sum_{i=1}^{I} b_i v_i - \sum_{j-1}^{J} c_j h_j \tag{2.8}$$

where w_{ij} represents the symmetric interaction term between visible unit v_i and hidden unit h_j, b_i and c_j the bias terms. Then, I and J are the numbers of visible and hidden units.

The conditional probabilities can be efficiently calculated as:

$$p(v_i = 1|\mathbf{h}) = \sigma(b_i + \sum_j W_{ij} h_j) \tag{2.9}$$

The conditional probability $p(v_i = 1|\mathbf{h})$ of the visible element conditioned by the hidden element is given by Bayes' theorem.

$$p(h_j = 1|\mathbf{v}) = \sigma(c_j + \sum_i v_i W_{ij}) \tag{2.10}$$

Similarly, the conditional probability $p(h_j = 1|\mathbf{v})$ of hidden elements conditioned with visible elements, where $\sigma(x) = 1/(1 + exp(-x))$.

Basicly the DBN works in multilayer stacked belief networks. Where its begin with the RBM theorem stacked layer against each other to for other deep belief network. When we begins to train the DBN for learning a layer of features from the visible units, using Contrastive Divergence (CD) algorithm. Then, the next step is to treat the activations (sigmoid) of previously trained features as visible unites and learn features of features in a second hidden layer. Finally, the whole DBN is trained when the learning for the final hidden layer is achieved.

The contrastive Divergence (CD) as known as greedy layer learning is a pre-training algorithm that aims to train each layer of a DBN in a sequential way, feeding lower layers' results to the upper layers. In terms of computational units, deep structures such as the DBN can be much more efficient than their shallow counterparts since they require fewer units for performing the same function[73].

However, training deep structures can be difficult because there may be a high dependency on all layer parameters, namely the relationship between the image and pixel parts. To overcome this problem, it is suggested that we must do two things. The first step is to adapt the bottom layer to give good input to the top layer final settings (the more difficult part). Next we need to adjust the top layer to use the top layer final setting[58].

A greedy multi-layered training was introduced only to overcome this problem. This can be used to train DBN in a layer-wise sequence where each layer consists of RBM, and it is confirmed to bring better generalizations by initializing the local minimum (or local criteria) which helps to formulate a high representation of the level of input abstraction to the network.

Among the greedy layer training subset (not including semi-supervision training that adapts supervised and unattended parts of training objectives), generally unattended layer training has better performance than supervised layer training. This is because the supervised method might, so to speak, "too greedy" and throw away some useful information in the hidden layer[60].

The model distribution of greedy algorithm between observed vector x and l with hidden layer h_z is as follows:

$$P\left(x, h^1, ..., h^l\right) = \left(\prod_{z=0}^{l-2} P\left(h^z|h^{z-1}\right) P\left(h^{l-1}|h^l\right)\right) \tag{2.11}$$

where the distribution for visible units conditioned on hidden units of a RBM block at level k is represented by $P\left(h^z|h^{z-1}\right)$, and the visible-hidden joint distribution of top-level RBMs is represented by $P\left(h^{l-1}|h^l\right)$. The greedy layer algorithm pseudo code for DBN can be generalized as following:

Step 1 Let raw input x be the first RBM layer that we want to train, x = h(0).

Step 2 Use the resulting representation from the first layer as an input data to the second layer. This representation can either mean activation data $P\left(h^1\right) = (1|h^0)$.

Step 3 Then train the second layer as a RBM and keep the mean activations from first layer as training data of the visible layer in this RBM.

Step 4 Repeat step 2 and step 3 for each iteration feed upwards either the mean activations.

Step 5 Finally, adapt the fine-tuning on all parameters of the unsupervised network to transform it into classifiers by adding an extra logistic regression classifier and training by gradient descent on a supervised training criterion.

Boltzmann machines and related models are traditionally trained by stochastic gradient ascent rather than in batch mode as is usual in speech processing. In the parameter updating process, a contrastive divergence (CD) learning is highly successful and is becoming the standard learning method to train the RBM parameters. The purposes of CD to approximate the second term in the log-likelihood gradient sample from the RBM distribution. First let us calculate the gradient of the log likelihood with respect to the model parameters of Gaussian-Bernoulli RBM (GBRBM), given a training ∂J and $\partial\theta$, the gradient of the log likelihood function is :

$$\frac{\partial J}{\partial \theta} = -\langle \frac{1}{\sum_h e^{-E}} \sum_h \frac{\partial E}{\partial \theta} e^{-E} \rangle_q + \frac{1}{Z} \sum_v \sum_h \frac{\partial E}{\partial \theta} e^{-E} \tag{2.12}$$

$$= -\sum_v \sum_h \frac{\partial E}{\partial \theta} p(h|v) q(v) + \sum_v \sum_h \frac{\partial E}{\partial \theta} p(v,h) \tag{2.13}$$

$$\equiv -\langle \frac{\partial E}{\partial \theta} \rangle_{data} + \langle \frac{\partial E}{\partial \theta} \rangle_{model} \tag{2.14}$$

which is zero at a critical point of the likelihood function, just as in the case of a Gaussian Markov random field. We will refer to the correlations $\langle \partial J \rangle_{data}$ *and* $\langle \partial \theta \rangle_{model}$ as the data statistics and the model statistics. Here, $\langle \cdot \rangle_{data}$, $\langle \cdot \rangle_{model}$ is $p_{data} = p(h|v)q(v)$ and $p_{model}(v,h) = p(v,h)$. The first term $\langle \cdot \rangle_{data}$ in the expression (2.12) can be calculated relatively easily, but for the second term $\langle \cdot \rangle_{model}$, every $\mathbf{v}$, *bfh* since the sum must be taken against, the state quantity explodes exponentially due to the increase in the number of units, so calculation is often very difficult. Therefore, by using the CD (Contractive Divergence) method, $p(v,h)$ sampling is performed and an average is taken to obtain an expected value approximately. Calculate specific parameters b_i, c_i, W_{ij}. Training with hidden units to train Boltzmann machines with hidden units, we use the EM algorithm. First, cal-

culate w_{ij}: $\partial E/\partial W_{ij} = -v_i h_j$, w_{ij} defined as:

$$\frac{\partial J}{\partial W_{ij}} = \langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{model} \tag{2.15}$$

$$= \frac{1}{N}\sum_k v_i^k \sigma(c_j + \sum_i v_i^k W_{ij}) + \frac{1}{N}\sum_k \hat{v}_i^k \sigma(c_j + \sum_i \hat{v}_i^k W_{ij}) \tag{2.16}$$

which is b_i can be represented $\partial E/\partial b_i = -v_i$:

$$\frac{\partial J}{\partial b_i} = \langle v_i \rangle_{data} - \langle v_i \rangle_{model} \tag{2.17}$$

$$= \frac{1}{N}\sum_k v_i^k - \frac{1}{N}\sum_k \hat{v}_i^k \tag{2.18}$$

Then c_j represents as $\partial E/\partial c_j = -h_j$:

$$\frac{\partial J}{\partial c_j} = \langle h_j \rangle_{data} - \langle h_j \rangle_{model} \tag{2.19}$$

$$= \frac{1}{N}\sum_k \sigma(c_j + \sum_i v_i^k W_{ij}) - \frac{1}{N}\sum_k \sigma(c_j + \sum_i \hat{v}_i^k W_{ij}) \tag{2.20}$$

In summary, the update formula is as follows:

$$\frac{\partial J}{\partial W_{ij}} = \frac{1}{N}\sum_k v_i^k \sigma(c_j + \sum_i v_i^k W_{ij}) + \frac{1}{N}\sum_k \hat{v}_i^k \sigma(c_j + \sum_i \hat{v}_i^k W_{ij}) \tag{2.21}$$

$$\frac{\partial J}{\partial b_i} = \frac{1}{N}\sum_k v_i^k - \frac{1}{N}\sum_k \hat{v}_i^k \tag{2.22}$$

$$\frac{\partial J}{\partial c_j} = \frac{1}{N}\sum_k \sigma(c_j + \sum_i v_i^k W_{ij}) - \frac{1}{N}\sum_k \sigma(c_j + \sum_i \hat{v}_i^k W_{ij}) \tag{2.23}$$

In the machine learning literature, Boltzmann machines are principally used in the unsupervised training of another type of generative model known as a sigmoid belief network or deep belief network (DBN). The hidden variables in a DBN can be thought of as providing causal explanations of data so that calculating the posteriors of the hidden variables in the DBN can be viewed as a sort of high-level feature extraction. The standard (mean-field) approximation used for posterior calculations in a DBN is implemented by a feed forward neural net (where the top level softmax layer usually used to make recognition decisions is missing). Suppose then that a DBN can be trained in an unsupervised way on a given data population in such away that the high level feature extractors learned in the course of training are useful for recognizing patterns in the data. Under such circumstances, it would be reasonable to use the posterior calculations in the DBN as an initialization for backpropagation training of a feed-forward neural network designed to discriminate between these patterns.

When speech signal was analyzes using the DBN to classify the feature of acoustic signal. The common techniques to extract of acoustic signal are used in Gaussian mixture models (GMMs) and hidden Markov models (HMMs), linear or non-linear dynamical systems and conditional random fields (CRFs). This strategy has been successfully deployed in speech recognition, notably at Microsoft. Interestingly, Microsoft has also recently reported excellent speech recognition results obtained with unaided backpropagation in deep neural nets (that is, without using a DBN-based initialization) so the question of whether deep belief networks will play an important role in speech recognition in the future remains to be settled.

The advantages of using deep architectures for modeling the data and the challenges we face in training the deep architectures. The depth of an architecture is defined as the longest path between any inputs and any outputs of the system. For example, in neural networks, the depth of the architecture is the number of the hidden layers plus one[115]. Also, The unsupervised training on this area procedure its allows us to use all of our data in the process of training and it does not require training criterion to be labeled.

2.2.2 Deep Learning Architecture

Deep learning refers to a rather wide class of machine learning techniques and architectures, with the hallmark of using many layers of non-linear information processing stages that are hierarchical in nature. Depending on how the architectures and techniques are intended for use, for example this theory can be applied on synthesis/generation or recognition/classification, one can categorize most of the work in this area into three types: Generative deep architectures, discriminative deep architectures, Hybrid deep architectures. The detail about these techniques can be explained in section 2.2.2.1, 2.2.2.2 and 2.2.2.3.

2.2.2.1 Generative Deep Architecture

Generative models can often be represented as a graphical model [57]: this theory is visualized as a graph where nodes represent random variables and arcs say something about the type of dependence that exists between random variables. The combined distribution of all variables can be written in terms of products that involve only nodes and neighbors in the graph. Some random variables in the graphical model can be observed, and others cannot (called hidden variables). The sigmoid belief network is a generative multi-layer neural network proposed and trained using a variational approach. In a sigmoid belief network, the units (typically binary random variables) in each layer are independent given the values of the units in the layer above.

Other types of generative models that stand out are Boltzmann machines in or DBM[48]. DBM contains many hidden variable layers and does not have connections between variables in the same layer. This is a special case of a general Boltzmann machine (BM), which is a network of symmetrically connected units that make stochastic decisions about whether to live or die. Despite having a very simple learning algorithm, general BM is very complex to learn and very slow in learning. In DBM, each layer captures complex and high-level correlations between hidden feature activities in

the layers below. DBM has the potential to study internal representations that become increasingly complex, it is highly desirable to solve the problem of object recognition and speech. Furthermore, high-level representations can be built from a large supply of non-labeled sensory inputs and very limited data labels which can then be used to only slightly refine the model for a particular task at hand.

Recurrent Neural Network (RNN) is considered as a class of generative architecture in when they are used to model and produce data sequentially (for example[98]. "Depth" of an RNN can be as large as the length of the input data set. The RNN is very powerful for modeling sequence data (for example, speech or text), but until now they have not been used widely because they are very difficult to train properly due to the "gradient disappearance" problem. Recent advances in overcoming some of these difficulties by using Hessian free optimization as information in estimating the curvature of stochastic values. In a recent work[68], RNNs that are trained with Hessian free optimization are used as generative architectures in character level (LM) language modeling tasks, where fenced connections are introduced to allow current input characters to predict transitions from one latent state vector to one next. The generative RNN model is proven to be able to produce sequential text characters. Recently, Bengio was ete.al.[26] have explored new optimization methods in generative RNN training that change the stochastic gradient gradient and show this modification can outperform the Hessian free optimization method. Mikolovetal[72] has reported excellent results on the use of RNNs for LM. Recently, Mesnil et al[70] reported the success of RNN in understanding oral language.

As an example of a variety of different types of deep models, there has been a long history of voice recognition research in which the mechanism of human speech production goes through a process of exploitation to build dynamic and deep structures in probabilistic generic models. In particular, the learning described in[70] concerning generalization and conventional shallow and conditional HMM structures by applying dynamic constraints, in the form of polynomial trajectories, in the HMM parameter. Variants of this approach have been developed using different learning techniques for HMM parameters that vary in time and with extended applications for speech recognition resistance[36]. In general, in HMM theory it also forms the basis for parametric speech synthesis.

Graphic models can consist of many hidden layers to characterize complex relationships between variables in the generation of speech. Armed with powerful graphical modeling tools, to solve very difficult problems from single channels, multi-speaker speech recognition, where mixed speech is a variable that is seen while unmixed speech becomes represented in a new hidden layer within the generative architecture that is deep speech architecture the depth has been successfully applied[114].

Generative graphics models are indeed a powerful tool in many applications because of their ability to embed domain knowledge. However, in addition to the disadvantages of using non-distributed representations for classification categories, they are also often implemented with incorrect estimates in conclusion, learning, prediction, and topological design, all arising from the

constancy inherent in these tasks for most real world . application. This problem has become one of the main tasks in the process[65], which provides an interesting direction to make generative graphic models that are potentially more useful in future for practicing the theory.

Finally, deep generative theory can be found periodically in for human movement modeling, and for natural language and parsing of natural scenes[91]. This model is very interesting because the learning algorithm is able to automatically determine the optimal model structure. This is in stark contrast to other in-depth architectures such as DBN where only parameters are studied while architecture needs to be determined in advance. In particular, as reported in[110], recursive structures commonly found in images of natural landscapes and in natural language sentences can be found using the prediction architecture of maximum boundary structures. Not only are the units contained in the images or sentences identified but also the way these units interact with each other to form the whole.

In the expanded technical scope of signal processing, the signal is endowed with not only the traditional types such as audio, speech, image and video, but also text, language, and document that convey high-level, semantic information for human consumption. In addition, the scope of processing has been extended from the conventional coding, enhancement, analysis, and recognition to include more human-centric tasks of interpretation, understanding, retrieval, mining, and user interface. Many signal processing researchers have been working on one or more of the signal processing areas defined by the matrix constructed with the two axes of signal and processing discussed here. The deep learning techniques discussed in this article have recently been applied to quite a number of extended signal processing areas.

The traditional neural network or MLP has been in use for speech recognition for many years. When used alone, its performance is typically lower than the state-of-the-art HMM systems with observation probabilities approximated with Gaussian mixture models (GMMs). Recently, the deep learning technique was successfully applied to phone recognition and large vocabulary speech recognition tasks by integrating the powerful discriminative training ability of the DBNs and the sequential modeling ability of the HMMs.

More specifically, the work of, a five layer DBN was used to replace the Gaussian mixture component of the GMM-HMM and the mono-phone state was used as the modeling unit. Although mono- phones are generally accepted as a weaker phonetic representation than triphones, the DBN-HMM approach with mono-phones was shown to achieve higher phone recognition accuracy than the state-of-the-art triphone GMM-HMM systems.

The technique of [110] was improved in the later work reported by using the CRF instead of the HMM to model the sequential speech data and by applying the maximum mutual information (MMI) training technique successfully developed in speech recognition to the resultant DBN-CRF training. The sequential discriminative learning technique developed jointly optimizes the DBN weights, transition weights, and phone language model and achieved higher accuracy than the DBNHMM phone recognizer with the frame-discriminative training criterion implicit in the DBN's fine tuning procedure implemented.

The DBN-HMM was extended from the monophone phonetic representation to the triphone or context-dependent counterpart and from phone recognition to large vocabulary speech recognition. Experiments on the Bing mobile voice search dataset collected under the real usage scenario demonstrate that the triphone DBN-HMM significantly outperforms the state-of-the-art HMM system. Three factors contribute to the success: the use of triphones as the DBN modeling units, the use of the best available triphone GMM-HMM to generate the senone alignment, and the tuning of the transition probabilities. Experiments also indicate that the decoding time of a five-layer DBN-HMM is almost the same as that of the state-of-the-art triphone GMMHMM.

A type of deep auto-encoder developed originally for image feature coding was explored on the speech feature coding problem. The goal is to extract bottleneck speech features by compressing the high-resolution speech spectrogram data to a pre-defined number of bits with minimal reproduction error. DBN pre-training is found to be crucial for high coding efficiency. When the DBN pretraining is used, the deep auto-encoder is shown to significantly outperform a traditional vector quantization technique. If weights in the deep auto-encoder are randomly initialized the performance is substantially degraded.

Further, the most recent work makes use of the DCN architecture to perform frame-level phone classification. Higher accuracy than DBN is reported, especially after a ⊠fine-tuning technique developed in other study is exploited. While the preliminary work has not developed parallel implementation of the basic learning algorithm in the DCN architecture, active research is currently underway to enable high scalability of learning DCN via parallelization.

The convolutional structure is further imposed on DBN and is applied to audio and speech data for a number of tasks including music artist and genre classification, speaker identification, speaker gender classification, and phone classification, with strong results presented. The recent work makes use of speech sound waves as the raw input feature to an RBM with a convolutional structure as the classifier. The use of rectifier linear units in the hidden layer, it is possible to automatically normalize the amplitude variation in the waveform signal, thus overcoming the difficulty encountered in the earlier attempt of using the same raw feature in the HMM based approach.

In addition to RBM, DBN, and DCN, other deep models have also been developed and reported in the literature. For example, the deep-structured CRF, which stacks many layers of CRFs, have been successfully used in the task of language identification, phone recognition, sequential labeling in natural language processing and confidence calibration in speech recognition.

As another example, a new type of HMM is introduced in which a set of hidden basis vectors and associated weights and precision matrices are jointly optimized. This can be considered as a generative deep architecture where the hidden basis, together with the associated weights, gives an intermediate representation of the speech signal. The work explores making HMM training more generalizable to unknown data, achieved by the developed Bayesian sensing framework to realize model regularization.

The original DBN and deep auto-encoder were developed and demonstrated with success on the simple image recognition and dimensionality reduction (coding) tasks (MNIST). It is interest-

ing to note that the gain of coding efficiency using the DBN-based autoencoder on the image data over the conventional method of principal component analysis is very similar to the gain reported in the speech data over the traditional technique of vector quantization.

A modified DBN is developed where the top-layer model uses a third-order Boltzmann machine. This type of DBN is applied to the NORB database a 3-dimensional object recognition task. An error rate close to the best published result on this task is reported. In particular, it is shown that the DBN substantially outperforms shallow models such as SVMs.

Two strategies to improve the robustness of the DBN are developed. First, sparse connections in the first layer of the DBN are used as a way to regularize the model. Second, a probabilistic denoising algorithm is developed. Both techniques are shown to be effective in improving the robustness against occlusion and random noise in a noisy image recognition task.

DBNs have also been successfully applied to create compact but meaningful representations of images for retrieval purposes. On this large collection image retrieval task, deep learning approaches also produced strong results.

Use of conditional DBN for video sequence and human motion synthesis. The conditional DBN makes the DBN weights associated with a fixed time window conditioned on the data from previous time steps. The computational tool offered in this type of temporal DBN may provide the opportunity to improve the DBN-HMMs towards efficient integration of temporal-centric human speech production mechanisms into DBN-based speech production model.

A very interesting piece of recent work, where the authors from Stanford propose and evaluate a novel application of deep networks to learn features over both audio and video modalities. Cross modality feature learning is demonstrated better features for video can be learned if both audio and video information sources are available at feature learning time. The authors further show how to learn a shared audio and video representation, and evaluate it on a fixed task, where the classifier is trained with audio-only data but tested with video-only data and vice-versa. The work concludes that deep learning architectures are effective in learning multimodal features from unlabeled data and in improving single modality features through cross modality learning.

Research in language, document, and text processing has seen increasing popularity recently in the signal processing community, and has been designated as one of the main focus areas by the society's audio, speech, and language processing technical committee. There has been a long history of using (shallow) neural networks in language modeling (LM) an important component in speech recognition, machine translation, text information retrieval, and in natural language processing. Recently, deep neural networks have been attracting more and more attention in statistical language modeling.

An LM is a function that captures the salient statistical characteristics of the distribution of sequences of words in a natural language. It allows one to make probabilistic predictions of the next word given preceding ones. A neural network LM is one that exploits the neural network ability to learn distributed representations to reduce the impact of the curse of dimensionality.

A distributed representation of a symbol is a vector of features which characterize the meaning

of the symbol. With a neural network LM, one relies on the learning algorithm to discover meaningful, continuous-valued features. The basic idea is to learn to associate each word in the dictionary with a continuous-valued vector representation, where each word corresponds to a point in a feature space. One can imagine that each dimension of that space corresponds to a semantic or grammatical characteristic of words. The hope is that functionally similar words get to be closer to each other in that space, at least along some directions. A sequence of words can thus be transformed into a sequence of these learned feature vectors. The neural network learns to map that sequence of feature vectors to the probability distribution over the next word in the sequence.

The distributed representation approach to LM has the advantage that it allows the model to generalize well to sequences that are not in the set of training word sequences, but that are similar in terms of their features, i.e., their distributed representation. Because neural networks tend to map nearby inputs to nearby outputs, the predictions corresponding to word sequences with similar features are mapped to similar predictions.

The above ideas of neural network LM have been implemented in various studies, some involving deep architecture. Temporally factored RBM was used for language modeling. Unlike the traditional N-gram model the factored RBM uses distributed representations not only for context words but also for the words being predicted. This approach is generalized to deeper structures.

In the popular work on natural language processing, the other researcher developed and employed a convolutional DBN as the common model to simultaneously solve a number of classic problems including part-of-speech tagging, chunking, named entity tagging, semantic role identification, and similar word identification. More recent work reported in further developed a fast purely discriminative approach for parsing based on the deep recurrent convolutional architecture called Graph Transformer Network. A similar multi-task learning technique with DBN is used to attack a machine transliteration problem, which may be generalized to a more difficult machine translation problem.

The most interesting recent work on applying deep learning to natural language processing study, where a recursive neural network is used to build a deep architecture. The network is shown to be capable of successful merging of natural language words based on the learned semantic transformations of their original features. This deep learning approach provides an excellent performance on natural language parsing. The same approach is also demonstrated by the same authors to be successful in parsing natural scene images.

Deep learning is an emerging technology. Despite the empirical promising results reported so far, much needs to be developed. For example, recent published work shows that there is vast room to improve the current optimization techniques for learning deep architectures. While the current learning strategy of generative pre-training followed by discriminative fine-tuning seems to work well empirically for many tasks, it fails to work for some other tasks that we have explored (e.g., language identification). For these tasks, the features extracted at the generative pre-training phase seem to describe the underlining speech variations well but do not contain sufficient information to distinguish between different languages. A learning strategy that can extract discriminative features

is expected to provide better solutions. Extracting discriminative features may also greatly reduce the model size needed in the many current deep learning systems.

Further, effective and scalable parallel algorithms are essential to train deep models with very large data, as in many common information processing applications such as speech recognition and machine translation. The popular mini-batch stochastic gradient technique is difficult to be parallelized over computers. The common practice nowadays is to use graphical processing units (GPUs) to speed up the learning process. However, single machine GPU processing is not practical for large datasets, which is typical in speech recognition and similar applications. To make deep learning techniques scalable to thousands of hours of speech data, for example, theoretically sound parallel learning algorithms or novel architectures need to be developed. The DCN architecture presented in this paper is a promising direction toward the scalability goal, but much more work is needed in this area.

Finally, we discuss applications of DBN and deep auto encoder to document indexing and information retrieval. It is shown that the hidden variables in the last layer not only are easy to infer but also give a much better representation of each document, based on the word-count features, than the widely used latent semantic analysis. Using the compact code produced by deep networks, documents are mapped to memory addresses in such a way that semantically similar text documents are located at nearby address to facilitate rapid document retrieval. This idea is explored for audio document retrieval and some class of speech recognition problems with the initial exploration.

2.2.2.2 Discriminative Deep Architecture

The number of discriminatory techniques in analog signal processing and information applies to shallow architectures such as HMM (for example,[97]) or Conditional Random Field (CRF)[112]. CRF is a probabilistic framework for labeling and segmenting structured data, such as sequences, trees. The underlying idea is that of defining a conditional probability distribution over label sequences gave a particular observation sequence rather than independence assumptions required by HMMs. Because the CRF is defined with a conditional probability on the input data as well as on the output label, it is a shallow discriminatory intrinsic architecture. Recently, structured CRFs have been developed by stacking output in each layer under the CRF, along with the original input data, to higher layers. Various useful structured CRF versions are used for telephone recognition[113], identification of oral languages, and natural language processing. However, at least for telephone recognition tasks, CRF performance is structured deep, which is purely discriminatory (non-generative), has not been able to match the hybrid approach involving DBN, which we will take in the near future.

In the article[74] a very good review of the other major discriminatory models that exist in speech recognition is based primarily on traditional neural networks or Multilayer Perceptron (MLP) architecture using backpropagation learning with random initialization. This argues for the importance of both increasing the width of each layer of neural network and increasing depth. Specifically, the DNN model class forms the basis of the popular "tandem" approach, in which neural net-

works studied discriminatively are developed in the context of computing discriminant emission probabilities for HMMs. In some representative works in this field, the tandem approach produces discriminatory features for the HMM by using activities from one or more hidden layers of neural networks in various ways of combining information, which can be considered as discriminatory in-depth architectural forms[79].

RNNs have been successfully used as a generative model when the "output" is taken to be the predicted input data in the future, they can also be used as a discriminative model where the output is an explicit label sequence associated with the input data sequence. Note that such discriminative RNNs were applied to speech a long time ago with limited success. For training RNNs for discrimination, pre-segmented training data are typically required. Also, post-processing is needed to transform their outputs into label sequences. It is highly desirable to remove such requirements, especially the costly presegmentation of training data. Often a separate HMM is used to automatically segment the sequence during training and to transform the RNN classification results into label sequences[84]. However, the use of HMM for these purposes does not take advantage of the full potential of RNNs.

Another type of discriminative deep architecture is a convolutional neural network (CNN), with each module consisting of a convolutional layer and a pooling layer. These modules are often stacked up with one on top of another, or with a DNN on top of it, to form a deep model. The convolutional layer shares many weights, and the pooling layer subsamples the output of the convolutional layer and reduces the data rate from the layer below. The weight sharing in the convolutional layer, together with appropriately chosen pooling schemes, endows the CNN with some "invariance" properties. Nevertheless, the CNN has been found highly effective and been commonly used in computer vision and image recognition. More recently, with appropriate changes from the CNN designed for image analysis to that taking into speech specific properties, the CNN is also found effective for speech recognition[25].

It is useful to point out that time-delay neural networks (TDNN) developed for early speech recognition area special case of the CNN when weight sharing is limited to one of the two dimensions [56]. It was not until recently that researchers have discovered that time is the wrong dimension to impose "invariance" and frequency dimension is more effective in sharing weights and pooling outputs[56]. It is also useful to point out that the model of hierarchical temporal memory (HTM) is another variant and extension of the CNN. The extension includes the following aspects: Time or temporal dimension is introduced to serve as the "supervision" information for discrimination (even for static images);both bottom-up and top-down information flow is used, instead of just bottom-up in the CNN; and a Bayesian probabilistic formalism is used for fusing information and for decision making[41].

Finally, the learning architecture developed for bottom-up, detection-based speech recognition proposed in and developed further since 2004, notably in[93] using the DBN–DNN technique, can also be categorized in the discriminative deep architecture category. There is no intent and mechanism in this architecture to characterize the joint probability of data and recognition targets

of speech attributes and of the higher-level phone and words. The most current implementation of this approach is based on multiple layers of neural networks using back-propagation learning. One intermediate neural network layer in the implementation of this detection-based framework explicitly represents the speech attributes, which are simplified entities from the "atomic" units of speech developed in the early work of[96]. The simplification lies in the removal of the temporally overlapping properties of the speech attributes or articulatory-like features. Embedding such more realistic properties in the future work is expected to improve the accuracy of speech recognition further.

2.2.2.3 Hybrid Architecture

The term "hybrid" for this third category refers to the deep architecture that either comprises or makes use of both generative and discriminative model components. Inn the many existing review, the generative component is exploited to help with discrimination, which is the final goal of the hybrid architecture[19]. There are two points how to hybrid help with discrimination processing. Firstly, The optimization viewpoint where generative models can provide excellent initialization points in highly nonlinear parameter estimation problems. Secondly, The regularization perspective where generative models can effectively control the complexity of the overall model.

Another example of the hybrid deep architecture is developed in[19], where again the generative DBN is used to initialize the DNN weights but the fine tuning is carried out not using frame-level discriminative information (e.g., cross-entropy error criterion) but sequence level one. This is a combination of the static DNN with the shallow discriminative architecture of CRF. Here, the overall architecture of DNN–CRF is learned using the discriminative criterion of the conditional probability of full label sequences given the input sequence data. It can be shown that such DNN–CRF is equivalent to a hybrid deep architecture of DNN and HMM whose parameters are learned jointly using the full-sequence maximum mutual information (MMI) between the entire label sequence and the input vector sequence.

The generative ability of the DBN model facilitates the discovery of what information is captured and what is lost at each level of representation in the deep model,as demonstrated in[45] A related work on using the discriminative criterion of empirical risk to train deep graphical models can be found in[43]. A further example of the hybrid deep architecture is the use of the generative model of DBN to pretrain deep convolutional neural networks (deep DNN). Like the fully-connected DNN discussed earlier, the DBN pretraining is also shown to improve discrimination of the deep CNN over random initialization.

The final example given here of the hybrid deep architecture is based on the idea and work of[40], where one task of discrimination (speech recognition) produces the output(text) that serves as the input to the second task of discrimination (machine translation). The overall system, giving the functionality of speech translation translating speech in one language into text in another language is a two stage deep architecture consisting of both generative and discriminative elements. Both models of speech recognition (e.g.,HMM) and of machine translation are generative

in nature. But their parameters learned for discrimination. The framework described in enables end-to-end performance optimization in the overall deep architecture using the unified learning framework initially published in[102]. This hybrid deep learning approach can be applied to not only speech translation but also all speech and possibly other information processing tasks such as speech information retrieval,speech understanding, cross lingual speech/text understanding and retrieval, etc.

2.3 Speech Recognition Techniques

The speech recognition is defined as the process of considering the spoken word as an input speech and matches it with the previously recorded speeches on basis of various parameters. This can be done by various methods. It is a process of automatically recognizing who is speaking on the basis of features of speaker of the speech signal.

2.3.1 Methodology of Feature Extraction in Speech Signal

The first step in segmentation of continuous speech signal is to digitalize the signal. The short-term energy function for the digitalized signal is calculated. FFT (Fast Fourier Transformation) of the energy function is found. Length of the window is approximately adjusted to the length of the signal so that obtained result is closer to the number of phonemes in the taken speech signal. Various windowing techniques are applied to the continuous speech signal and its performance is evaluated. The resultant FFT is raised to the power spectral so that the magnitude spectrum calculated is brought to the optimized value. Next step, the signal is plotted against the Mel spectrum to mimic human hearing. Each pitch of a pure tone with an actual measured frequency f measured in Hz, a subjective pitch is measured on a scale called the 'Mel Scale'. Mel Filter Bank filters an input power spectrum through a bank of number of Mel-filters. Then, processes the digitalize signal into discrete cosine transform to perform the spectogram.

2.3.1.1 Windowing

In signal processing and Communication Systems a filter removes the unwanted signal and allows the desired signal. Different forms of filters available based on applications. Low pass filters allows only low frequency band. High pass filters allows only high frequency. Digital filters are broadly classified in to two types. They are Finite Impulse Response (FIR) filter and Infinite Impulse Response (IIR) filter. FIR system has finite impulse response where as IIR system has infinite impulse response. While implementation of these filters, FIR filter has no feedback and it is also called non-recursive filter. Because of this, FIR filter structure can easily be implemented as compared to the IIR filter.

A function which is a mathematical function that is zero-valued outside the chosen interval is considered as window function. FIR filters can be designed using, Fourier series method, Frequency sampling method and Window methods are used. FIR filters implementation using Fourier

series method encounters some problems i.e. If the Fourier series is truncated abruptly, it results in oscillations in the pass band and stop band. These oscillations are due to slow convergence of the Fourier series, This Phenomenon is called Gibb's Phenomenon. We can overcome this by using an appropriate window function. For a discrete-time FIR filter, the output is a weighted sum of the current and a finite number of previous values of the input.

The basic approach of the windowing techniques is to multiply the sequence $x[n]$ by a 'window sequence' $w[n]$ that is non-zero only for n=0,..., L-1, where L, the length of the window, is smaller than the length N of the sequence $x[n]$. The simple way to express windowing function is:

$$x_w[n] = x[n] \times w[n] \tag{2.24}$$

Windowing techniques are mainly used in the process of designing digital filters. In order to convert an impulse response of infinite duration to a Finite Impulse Response (FIR) filter design windowing is performed. Symmetrical sequences of Window functions generated for digital filter design. Those window functions are usually an odd length with a single maximum at the center. For spectral analysis5, Windows for DFT/FFT are formed by removing the right-most coefficient of an odd-length, symmetrical window. Truncated sequences are known as periodic. When the truncated sequence is periodically extended, the deleted coefficient is commendably restored (by a virtual copy of the symmetrical left-most coefficient). Window technique consists of a function called window function which is nothing but if some interval is chosen, it returns with finite non-zero value inside that interval and zero value outside that interval.

Hamming window is used to optimize the window to minimize the maximum (nearest) side lobe, giving it a height of about one-fifth that of the Hann window. The formulae can be written as:

$$w(n) = \alpha + \beta \cos(\frac{2\pi n}{N-1}) \tag{2.25}$$

where, $\alpha = 0.54$, $\beta = 1 - \alpha = 0.46$

By using windowing functions, you can further enhance the ability of an FFT to extract spectral data from signals. Windowing functions act on raw data to reduce the effects of the leakage that occurs during an FFT of the data. Leakage amounts to spectral information from an FFT showing up at the wrong frequencies. The people who first studied the effect thought of the spectral information as "leaking" into adjacent frequency values.

The figure 2.3.1 is the the window function of performances in Hamming windowing and its called raised sine-squared, where the sampled signal values are multiplied by the Hanning function, and the result is shown in the figure. Note that the ends of the time record are forced to zero regardless of what the input signal is doing.

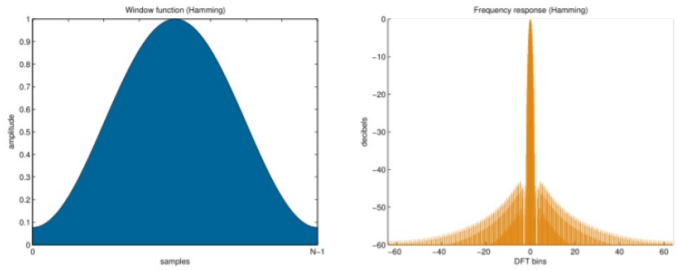

Figure 2.3.1: The window function and spectral leakage for the Hamming window

2.3.1.2 Fast Fourier Transform

The history of the Fast Fourier Transform (FFT) is quite interesting. It starts in 1805, when Carl Friedrich Gauss tried to determine the orbit of certain asteroids from sample locations. Thereby he developed the Discrete Fourier Transform (DFT), even before Fourier published his results in 1822. To calculate the DFT he invented an algorithm which is equivalent to the one of Cooley and Tukey. However, Gauss never published his approach or algorithm in his lifetime. It appeared that other methods seemed to be more useful to solve this problem. Probably, that is why nobody realized this manuscript when Gauss' collected works were published in 1866. It took another 160 years until Cooley and Tukey reinvented the FFT[37].

A discrete Fourier transform (DFT) can be thought of as a function that decomposes an audio signal into a set of coefficients for a range of basis functions [91]. The basis functions summed together will give an approximation of the original function. We can consider this as the analogy of a bunch of intertwined cables. Picture that there is a large pile of Christmas lights all tangled together on the floor. We have no idea how many different sets there are or of which lengths. Now suppose there is this exciting new machine that you can throw the pile of lights in, it unties them all for you, and then spits them out one by one in order of size. The machine essentially is a Fourier transform, where we input our audio signal, and out comes the frequency's coefficients to place on the basis functions that make up the signal. DFT is NOT the same as the DTFT. Both start with a discrete-time signal, but the DFT produces a discrete frequency domain representation while the DTFT is continuous in the frequency domain. These two transforms have much in common, however. It is therefore helpful to have a basic understanding of the properties of the DTFT. The DFT Formula defined as follows:

$$\widehat{x_k} = \frac{1}{N}\sum_{j=0}^{N-1} x_k e^{-2\pi i \frac{jk}{N}}, \quad k = 0, \ldots, N-1 \tag{2.26}$$

This size of an audio signal is determined by how many points make up the wave. If the sample rate was 1000, then each second would be composed of 1000 points. Based on the size of the audio signal, the Fourier transform will produce that many coefficients for the basis functions which

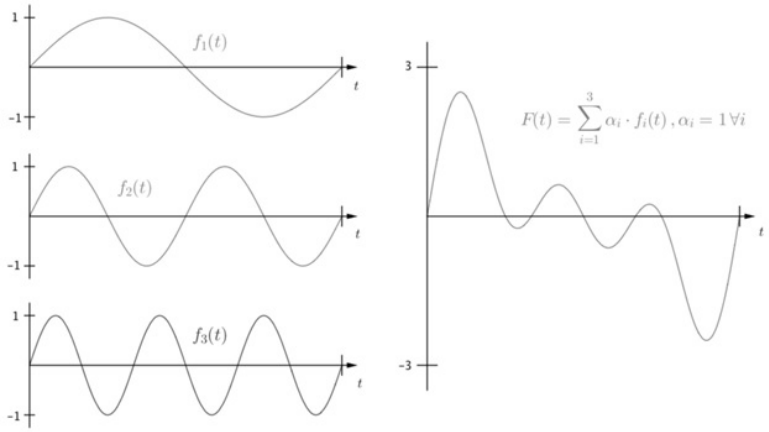

Figure 2.3.2: Single Wave with different Phase

are composed of sine and cosine functions. The basic formulae of Fast Fourier Transform can be written as:

$$X_k = \sum_{n=0}^{N-1} x_n . e^{-i 2\pi k n / N} \tag{2.27}$$

- X_k = Amount of frequency k (a complex number where the real part is the amplitude and the imaginary part is the phase).
- N = the number of time samples (length of the audio signal).
- x_n = value of the time signal at time n.
- k = current frequency under consideration.

The Fourier transform takes data from a time domain and moves it into a frequency domain. So the inverse Fourier transform does just the opposite. It moves the frequency spectrum back into a time domain. The inverse transform is found with the following summation:

$$X_n = \frac{1}{N} \sum_{k=0}^{N-1} x_k . e^{-j 2\pi n k / N} \tag{2.28}$$

The main difference is that the values x_n are now found at time n, the explanation of abbreviation same in equation 2.10. It is important to remember that when we analyzing an FFT, the magnitude of the coefficients are used which means some information is lost along the way such as the phase. Whenever an inverse transform is taken, it must be performed on the complex symmetric data. An inverse transform on the magnitude values will not produce the same results. We must also make sure that our data is symmetric complex conjugates and that the first value is a real number. The data can be very sensitive when reverting back to a time domain so caution must be taken when manipulating the coefficients of the FFT.

When analyzing an FFT, the target of interest is the *peaks* that appear. These *peaks* occur at locations where the corresponding frequency is dominant in the audio sample. Some audio samples are cleaner and easier to identify *peaks* than others. Consider an audio sample of an instrument playing a single note and compare this to an audio sample of someone speaking. The voice sample is obviously more complex. So the FFT of the voice will have much more going on with many *peaks*. There will be many more non-zero coefficients. The reason these *peaks* are of interest is that they identify which frequencies are most prevalent in the audio sample. Recall the objective of this thesis is to determine a method of converting someone's vocal frequencies to a targets voice. In order to do this, it is desired to find key characteristics of a persons voice and these large *peaks* help identify which frequencies are fundamental to their voice. The random signal speech sample of FFT when normalized with the peak procedures shown in Figure 2.3.2. Once the *peaks* have been extracted from both the source and target, the next step is to identify a way of mapping the pattern samples.

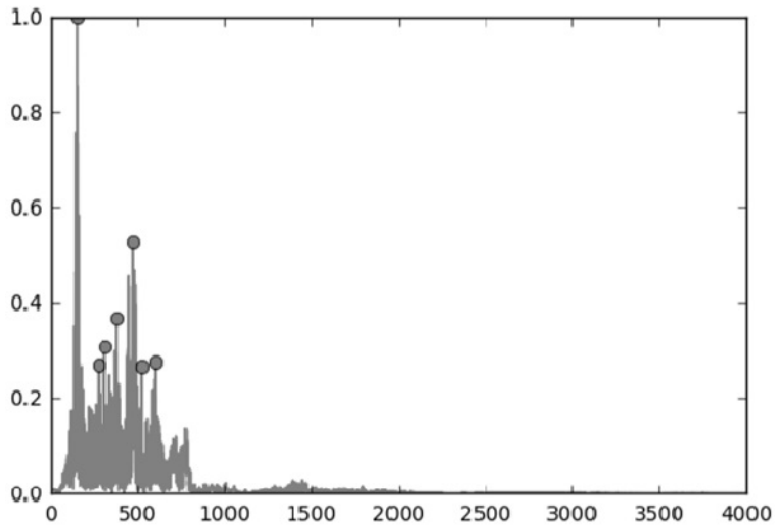

Figure 2.3.3: Random Signal FFT with Peaks Procesess

The pattern-matching approach: It involves two essential steps namely, pattern training and pattern comparison. The essential feature of this approach is that it uses a well-formulated mathematical framework and establishes consistent speech pattern representations, for reliable pattern comparison, from a set of labeled training samples via a formal training algorithm.

The job of the pattern-matching module is to combine information (probabilities) from the acoustic model, the language model, and the word lexicon to find the 'optimal' word sequence, i.e., the word sequence that is consistent with the language model and that has the highest probability among all possible word sequences in the language i.e., best matches the spectral feature vectors of the input signal.

A prototype implementation of a speech recognition system for embedded applications. The recognition system is comprised of a feature extractor and a classifier. The feature extractor is based on a 64-point Fast Fourier Transformation (FFT); the classifier is based on discrete density Hidden Markov Models (HMM) with a variable codebook size. Training as well as classification is implemented using the Viterbi algorithm. The prototype is implemented on a digital signal processor (DSP) of type TMS320C40 from Texas Instruments. The recognition rate and the performance are experimentally evaluated using a test vocabulary of 20 words. The recognition is implemented in three consecutive steps: feature extraction, vector quantization and probability calculation (classification).

The recognition including these three steps was measured for a typical word of test vocabulary, using a codebook size c = 32 and a number of states N = 5. 103 feature vectors were generated for these words that are equivalent to an utterance time of 0.6 s. The total time required to recognize these words is 738 ms. A prototype of an ASR system for command and control applications has been reported. It allows online recognition with limited memory and runtime. A recognition rate of 99 % was achieved by using a test vocabulary of 20 words.

Multi Pattern Viterbi Algorithm (MPVA) to jointly decode and recognize multiple speech patterns for automatic speech recognition (ASR). The MPVA is a generalization of the Viterbi Algorithm (VA) to jointly decode multiple patterns for a given standard Hidden Markov Model (HMM). Unlike Constrained Multi Pattern Viterbi Algorithm (CMPVA), the MPVA does not require the Multi Pattern Dynamic Time Warping (MPDTW) algorithm. The MPVA algorithm has the advantage that it can be extended to connected word recognition (CWR) and continuous speech recognition (CSR) problems. It also gives an improved speech recognition performance over the earlier techniques.

Using only two repetitions of noisy speech patterns (-5 dB SNR, 10% burst noise), the word error rate using the MPVA decreases by 28.5 percent, when compared to using individual decoding. MPVA is a generalization of single pattern Viterbi decoding for HMM. A single optimum state sequence is determined for the K set of patterns jointly. The formulation includes the local continuity constraints in determining the optimum path through the $(K + 1)$ dimensional grid. Based on this algorithm, the calculated ASR accuracy is significantly improved over that of single pattern VA. This technique is outperforming CMPVA technique in the presence of noise. The MPVA formulation has the generality of being applicable to many other problems, where robustness of HMM based pattern matching is required.

The effectiveness of perceptual features for performing isolated digits and continuous speech recognition. The proposed perceptual features are captured and codebook indices are extracted. Expectation maximization algorithm is used to generate HMM models for the speeches. Speech recognition system is evaluated on clean test speeches and the experimental results reveal the performance of the proposed algorithm in recognizing isolated digits and continuous speeches based on maximum log likelihood value between test features and HMM models for each speech. Performance of these features is tested on speeches randomly chosen from "TI Digits-1", "TI Digits-2"

and "TIMIT" databases. This algorithm is tested for VQ and combination of VQ and HMM speech modeling techniques. Perceptual linear predictive cepstrum yields the accuracy of 86% and 93% for speaker independent isolated digit recognition using VQ and combination of VQ_ HMM speech models respectively. This feature also gives 99% and 100% accuracy for speaker independent continuous speech recognition by using VQ and the combination of VQ_ HMM speech modeling techniques.

The development and advances in automatic speech recognition for the Speak4it voice search application. With Speak4it as real-life example, the effectiveness of acoustic model (AM) and language model (LM) estimation (adaptation and training) on relatively small amounts of application field-data is shown. Algorithmic improvements concerning the use of sentence length in LM, of non-contextual features in AM decision trees, and of the Teager energy in the acoustic front-end is introduced. The combination of these algorithms, integrated into the Watson recognizer, yields substantial accuracy improvements. LM and AM estimation on field-data samples increases the word accuracy from 66.4% to 77.1%, a relative word error reduction of 32%.

Template based approach matching: It is unknown speech compared against a set of pre-recorded words (templates) in order to find the best match. This has the advantage of using perfectly accurate word models. Template based approach to speech recognition has provided a family of techniques that have advanced the field considerably during the last six decades. It also has the disadvantage that pre-recorded templates are fixed, so variations in speech can only be modeled by using many templates per word, which eventually becomes impractical.

Knowledge based approach: An expert knowledge about variations in speech is hand coded into a system. This has the advantage of explicit modeling variations in speech; but unfortunately such expert knowledge is difficult to obtain and use successfully. Thus this approach was judged to be impractical and automatic learning procedure was sought instead. Vector Quantization (VQ) is often applied to ASR. It is useful for speech coders, i.e., efficient data reduction. Since transmission rate is not a major issue for ASR, the utility of VQ here lies in the efficiency of using compact codebooks for reference models and codebook searcher in place of more costly evaluation methods. For IWR, each vocabulary word gets its own VQ codebook, based on training sequence of several repetitions of the word.

Statistical based approach: The variations in speech are modeled statistically, using automatic, statistical learning procedure, typically the Hidden Markov Models, or HMM. The approaches represent the current state of the art. The main disadvantage of statistical models is that they must take priori-modeling assumptions, which are answerable to be inaccurate, handicapping the system performance. In recent years, a new approach to the challenging problem of conversational speech recognition has emerged, holding a promise to overcome some fundamental limitations of the conventional hidden Markov model (HMM) approach. This approach is a radical departure from the current HMM-based statistical modeling approaches. For text independent speaker recognition left right HMM is used for identifying the speaker from simple data. HMM has advantages based on Neural Network and Vector Quantization. The HMM is a popular statistical tool for model-

ing a wide range of time series data. In speech recognition area HMM has been applied to speech classification.

A weighted hidden Markov model HMM algorithm and a subspace projection algorithm are used to address the discrimination and robustness issues for HMM based speech recognition. Word models were constructed for combining phonetic and phonemic models. Learning Vector Quantization (LVQ) method showed an important contribution in producing highly discriminative reference vectors for classifying static patterns. The ML estimation of the parameters via FB algorithm was an inefficient method for estimating the parameters of HMM. To overcome this problem a corrective training method that minimized the number of errors of parameter estimation was developed.

A novel approach was used for a hybrid connectionist HMM speech recognition system based on the use of a Neural Network as a vector quantisation. It showed the important innovations in training the Neural Network. The vector quantization approach showed much of its significance in the reduction of word error rate. MVA method was obtained from modified Maximum Mutual Information (MMI). Various methods are used for estimating a robust output probability distribution (PD) in speech recognition based on the discrete Hidden Markov Model (HMM). An extension of the viterbi algorithm made the second order HMM computationally efficient when compared with the existing viterbi algorithm. A general stochastic model that encompasses most of the models proposed in the literature, pointing out similarities of the models in terms of correlation and parameter time assumptions, and drawing analogies between segment models and HMMs is presented.

The trajectory folding phenomenon in HMM model is overcome by using continuous density HMM which significantly reduced the word error rate over continuous speech signal. A new hidden Markov model integrating the generalized dynamic feature parameters with model structure was developed. It was evaluated using maximum-likelihood (ML) and minimum-classification-error (MCE) pattern recognition approaches. The authors have designed the loss function for minimizing error rate specifically for the new model, and derived an analytical form of the gradient of the loss function. The K-means algorithm is also used for statistical and clustering algorithm of speech based on the attribute of data.

The K in K-means represents the number of clusters the algorithm should return. As the algorithm starts K points known as cancroids are added to the data space. The K-means algorithm is a way to cluster the training vectors to get feature vectors. In this algorithm the vectors are clustered based on attributes into k partitions. It uses the k means of data generated from Gaussian distributions to cluster the vectors. The objective of the k-means is to minimize total intra-cluster variance.

The interactions of front-end feature extraction and back-end classification techniques in HMM based speech recognizer. The goal was to find the optimal linear transformation of Mel-warped short-time DFT information according to the minimum classification error criterion. These transformations, along with the HMM parameters, were automatically trained using the gradient de-

scent method to minimize measure of overall empirical error count. The discriminatively derived state-dependent transformations on the DFT data were then combined with their first time derivatives to produce a basic feature set.

Experimental results showed that Mel-warped DFT features, subject to appropriate transformation in a state-dependent manner, were more effective than the Mel-frequency cepstral coefficients that have dominated current speech recognition technology. The best error rate reduction of 9% is obtained using the new model, tested on a TIMIT phone classification task, relative to conventional HMM. Compared to all three classifiers, THMM produced the lowest error rate and is the new efficient way of utilizing the input data. Mel-warped DFT features, subject to appropriate transformation in a state dependent manner, are more effective than the MFCCs that have dominated current speech recognition technology.

The Most of the present systems are based on statistical modeling, both at the acoustic and linguistic levels. Noise resistance has become one of the major bottlenecks for practical use of speech recognizers. The models that are presently investigated for increasing recognition performance are presented. The robustness of the systems must be enhanced for the use in adverse conditions like telephone, environmental noise, etc. The on-going efforts toward enhancing the quality of the models used at the acoustic level is presented which will contribute to the development of ASR systems in new application.

The ATRASR large vocabulary speech recognition system developed for ATR. A feature vector consists of 12 MFCCs, and log power is extracted from frames of 20 ms with 10 ms frame shift of data recorded with 16 kHz sampling rate. Cepstral mean subtraction (CMS) is applied. Clean speech Japanese gender-dependent acoustic models are trained using dialogue speech from the ATR travel arrangement task corpus and 25 hours read speech of phonetically balanced sentences. Phoneme-based HMMs with 2086 states generated by the MDL-SSS algorithm with diagonal covariance matrices are used.

The system uses a multi-class composite bi gram language model and word tri-gram language models for rescoring. The lexicon size is 55k words. For testing the noise-reduction system, a small database in the cafeteria at ATR was recorded, using the PDA microphone array and a close-talking microphone as reference. Two male speakers and two female speakers read 102 utterances each from the ATR basic travel expression corpus (BTEC) test set- 01. The reverberation time in the cafeteria was about 1 s. The average signal-to-noise ratio (SNR)for each speaker was listed. The frequency range is 50 Hz - 8 kHz.

A multi channel speech input device for general purpose PDAs for hands-free speech recognition was presented. The hands-free interface consisted of a real-time implementation of a combination of a robust generalized side lobe canceller and an MMSE estimator for log Mel-spectral energy coefficients of clean speech. Based on a small experimental database, it was found that both noise-suppression methods have similar performance and that the joint system highly improves the word accuracy of a large vocabulary speech recognizer.

The evaluating recognition at phone level is important since the words are always represented

by the concatenation of phones units. The behavior of speaker-independent phone recognition in continuous speech based on the technique of HMM was investigated on the selection of an optimal model topology in order to achieve a robust phone recognition system which accomplishes the tradeoff between model size and data training. Correct phone recognitionrate of 69.33 percent and accuracy rate of 63.05 was obtained.

The recognition of cochlear implant like spectrally reduced speech (SRS) using Mel frequency cepstral coefficient (MFCC) and hidden Markov model (HMM)-based automatic speech recognition (ASR). The SRS was synthesized from sub band temporal envelopes extracted from original clean test speech, whereas the acoustic models were trained on a different set of original clean speech signals of the same speech database. Changing the bandwidth of the sub band temporal envelopes had no significant effect on the ASR word accuracy.

In addition, increasing the number of frequency sub bands of the SRS from 4 to 16 improved the system performance significantly. Furthermore, the ASR word accuracy attained with the original clean speech can be achieved by using the 16, 24, or 32 sub band SRS. The experiments were carried out using the TI digits speech database and the HTK speech recognition toolkit.

The design and implementation of English digits speech recognition system using Matlab (GUI) based on the Hidden Markov Model (HMM), which provides a highly reliable way for recognizing speech. The system is able to recognize all English digits from Zero through Nine by translating the speech waveform into a set of feature vectors using Mel Frequency Cepstral Coefficients (MFCC) technique. Two modules called the isolated words speech recognition and the continuous speech recognition were developed.

Both modules were tested in both clean and noisy environments and showed a successful recognition rates. In clean environment and isolated words speech recognition module, the multi-speaker mode achieved 99.5% whereas the speaker independent mode achieved 79.5%. In clean environment and continuous speech recognition module, the multi-speaker mode achieved 72.5% whereas the speaker-independent mode achieved 56.25%.

However in noisy environment and isolated words speech recognition module, the multi-speaker mode achieved 88% whereas the speaker- independent mode achieved 67%. In noisy environment and continuous speech recognition module, the multi-speaker mode achieved 82.5% whereas the speaker independent mode achieved 76.67%.

Learning based approach: To overcome the disadvantage of the HMMs machine, learning methods could be introduced such as neural networks and genetic algorithm programming. In these machine-learning models explicit rules or other domain expert knowledge need not be given. They can be learned automatically through emulations or evolutionary process.

A back-propagation neural network (BNN) can be applied to isolated-word speech recognition. Simulation results show that a BNN provides an effective approach for small vocabulary systems. The recognition rate reaches 100% for a 5-word system and 94% for a 10-word system. The general techniques developed can be further extended to other applications, such as sonar target recognition, missile seeking and tracking functions in modern weapon systems and classification

of underwater acoustic signals.

The choice of feature vector plays an important role in the performance of the BNN. The recognition rate may decrease drastically or the system may not converge at all if the features are not correctly chosen. The feature vector chosen in the experiments consisted of the LPC coefficients, short time energy, zero-crossing rate and voiced/unvoiced classification. It worked well and provided good results. However predictions cannot be made about the likely performance of the methods in these areas until they are actually tested.

The use of synergistically integrated systems of microphone arrays and neural networks for robust speech recognition in variable acoustic environments, where the user must not be encumbered by microphone equipment. Existing speech recognizers work best for "high-quality close-talking speech". Performance of these recognizers is typically degraded by environmental interference and mismatch in training conditions and testing conditions. It is found that the use of microphone arrays and neural network processors can elevate the recognition performance of existing speech recognizers in an adverse acoustic environment, thus avoiding the need to retrain the recognizer, a complex and tedious task.The results showed that a system of microphone arrays and neural networks can achieve a higher word recognition accuracy in an unmatched training/testing condition than that obtained with a retrained speech recognizer using array speech for both training and testing, i.e., a matched training / testing.

A system of microphone arrays (MA) and neural networks (NN) for robust speech recognition. The system expand the power and advantages of existing ARPA speech recognizers to practical acoustic environments where users need not be encumbered by hand-held, body-worn, or tethered microphone equipment, and must have freedom of movement. Examples include Combat Information Centers, large group conferences, and mobile hands busy eyes-busy maintenance tasks. Use of MA provides auto directive sound pickup that is higher in quality than conventional microphones used at distances. NN processors learn and compensate for environmental interference, and to adapt the testing condition to the training condition. Recognition performance in hostile acoustic environments can thereby be elevated without the need to retrain the recognizer.

The concepts of impulse sampling, Fourier transforms, data windowing, homomorphic filtering, speech coding and classification techniques via MATLAB and NeuralWorks. Applications involving speech coding and phonetic classification were introduced as educational tools for reinforcing signal processing concepts learned in senior level communication classes at the U.S. Naval Academy. These software tools allow sampling an analog speech signal; find the pitch and formant frequencies, and phonetically classifying voice data. The speech coding algorithms used involve digital filtering, data windowing, and spectral analysis. The application provided the means of some of the aspects of diverse signal processing theory in a graphical and procedural manner.

The synergism of web and phone technologies has led to a new innovative voice web network. The voice web requires a voice recognition and authentication system incorporating a reliable speech recognition technique for secure information access across the Internet. In the experiment, a total number of 200 vowel signals from individuals with different gender and races were recorded. The

filtering process was performed using the wavelet approach to denoise and to compress the speech signals.

An artificial neural network, specially the probabilistic neural network (PNN) model, was then employed to recognize and to classify vowel signals into the respective categories. A series of parameter settings for the PNN model in classifying speech signal of vowels was investigated, and the results obtained were analyzed and discussed. Accurate speech recognition requires models that can account for a high degree of variability in the speech signals. The results indicated that the performance of the PNN network was influenced by the smoothing parameter. A small value of smoothing parameter that set the PNN to function as a nearest neighbor classifier yielded the best result.

A Multi Layer Perceptron (MLP) neural network in the log spectral domain has been employed to minimize the difference between noisy and clean speech. By using this method, as a pre-processing stage of a speech recognition system, the recognition rate in noisy environments has been improved. The application of the system was extended to different environments with different noises without retraining HMM model.

The feature extraction stage was trained with a small portion of noisy data, which was created by artificially adding different types of noises from the NOISEX-92 database to the TIMIT speech database. The proposed method suggests four strategies based on the system capability to identify the noise type and SNR. Experimental results show that the proposed method achieves significant improvement in recognition rates. A new nonlinear noise reduction algorithm motivated by the MMSE criterion in the log spectral domain was developed for environment-robustness speech recognition.

The system was developed by using a MLP neural network in the log spectra domain. Experimental results show that this method improves the recognition accuracy in different cases for the TIMIT task and its improvement is greater than that of MBSS. New approach has several key attributes. Only a small portion of the clean speech and the corresponding noisy speech is sufficient for the method to work. The new approach can improve the recognition accuracy without any extra information about noise such as distribution. It creates a trade-off between system requirements and improvement of recognition accuracy, and knowing the noise type and SNR lead to higher improvement.

A new speech recognition system design in 2010 according to the visible characteristics of speech. It is based on multiple neural networks to distinguish different speakers. Pulse Coupled Neural Network (PCNN) was input into the spectrogram for producing the corresponding time series icon as the feature parameters of speech. Then the feature parameters were input into the Probabilistic Neural Networks (PNN) for training PNN to realize speech recognition. The simulation results show higher speech recognition rate if speaker speech signal was extracted by Pulse Coupled Neural Network (PCNN).

A novel approach for implementing isolated speech recognition. While most of the literature on speech recognition (SR) is based on hidden Markov model (HMM), the present system is im-

plemented by radial basis function type neural network. The two phases of training and testing in a radial basis function type neural network has been described. All classifiers use linear predictive cepstral coefficients. It is found that the performance of radial basis function type neural networks is superior to the other classifier multi layer perceptron neural networks.

The promising results obtained through this design show that this new neural networks approach could compete with the traditional speech recognition approaches. Promising results were obtained both in the training and testing phases due to the exploitation of discriminative information with neural networks. It is found that RBF trains and tests faster than MLP. The radial basis function neural network architecture has been shown to be suitable for the recognition of isolated words.

Recognition of the words is carried out in speaker dependent mode. In this mode the tested data presented to the network are same as the trained data. The 16 linear predictive cepstral coefficients with 16 parameters from each frame improve a good feature extraction method for the spoken words, since the first 16 in the cepstrum represent most of the formant information. It is found that the performance of RBF classifier is superior to MLP classifier.

The speech theories and some methodological concerns about feature extraction and classification techniques widely used in speech recognition system. The isolated word speech recognition is compared with phoneme-based counterpart. Isolated-word ASR for fixed vocabularies was successfully implemented using HMM, ANN and SVM but suffers from lack of adaptability to other languages and increases in complexity as the size of vocabulary increase. Conversely, phonemes, the smallest unit of human speech sounds is apparently more feasible to represent the basic building block for cross-language mapping.

The phoneme-based approach has potential to overcome the lack of available training data. It is investigated to achieve a more generic speech recognizer. Isolated word speech recognition has high recognition rate in most of the applications. This depends on the complexity of the system such as speaker independent or dependent, vocabulary size, clean or noisy speech and read or spontaneous word. This type of system suffers from limitation in vocabulary size. As the demand for greater words to be recognized arises, this method is no longer valid and appropriate.

The artificial intelligence approach: The artificial intelligence approach attempts to mechanize the recognition procedure in the same way a person applies his intelligence in visualizing, analyzing, and finally making a decision on the measured acoustic features. Expert system is used widely in this approach. The Artificial Intelligence approach is a hybrid of the acoustic phonetic approach and pattern recognition approach.

In this, it exploits the ideas and concepts of acoustic phonetic and pattern recognition methods. Knowledge based approach uses the information regarding linguistic, phonetic and spectrogram. Some speech researchers developed recognition system that used acoustic phonetic knowledge to develop classification rules for speech sounds. While template based approaches have been very effective in the design of a variety of speech recognition systems; they provided little insight about human speech processing, thereby making error analysis and knowledge-based system enhance-

ment difficult.

On the other hand, a large body of linguistic and phonetic literature provided insights and understanding to human speech processing. In its pure form, knowledge engineering design involves the direct and explicit incorporation of expert's speech knowledge into a recognition system. This knowledge is usually derived from careful study of spectrograms and is incorporated using rules or procedures.

Another difficult problem is the integration of many levels of human knowledge phonetics, phonotactics, lexical access, syntax, semantics and pragmatics. Alternatively, combining independent and asynchronous knowledge sources optimally remains an unsolved problem. In more indirect forms, knowledge has also been used to guide the design of the models and algorithms of other techniques such as template matching and stochastic modeling.

This form of knowledge application makes an important distinction between knowledge and algorithms. Algorithms enable us to solve problems. Knowledge enables the algorithms to work better. This form of knowledge based system enhancement has contributed considerably to the design of all successful strategies reported. It plays an important role in the selection of a suitable input representation, the definition of units of speech, or the design of the recognition algorithm itself.

Genetic algorithm (GA) is used to replace the steepest descent method (SDM) for the training of BPNN such that a global search of optimal weight in neural network can improve the performance of speech recognition. The non specific speaker recognition, which is trained by SDM, the recognition rate to recognize Chinese speech was made with MFCC parameter with recognition rate up to 91%. If BPNN is trained by genetic algorithm, higher recognition, to solve the problem with local optimum, GA was adopted with SDM to improve MSE convergence.

Besides increasing GA speed, it also improved system recognition rate up to 95%. Under the condition of adopting only MFCC parameters, speech recognition rate still has room for improvement. Cepstrum coefficient or LPC parameter together with pitch parameter are other modes of calculating features to improve recognition rate.

Stochastic Approach: Stochastic modeling entails the use of probabilistic models to deal with uncertain or incomplete information. In speech recognition, uncertainty and incompleteness arise from many sources; for example, confusable sounds, speaker variability, contextual effects, and homophones words. Thus, stochastic models are particularly suitable approach to speech recognition. The most popular stochastic approach today is hidden Markov modeling. A hidden Markov model is characterized by a finite state Markov model and a set of output distributions.

The temporal variability's are transition parameters in the Markov chain models, while spectral variability's are the parameters in the output distribution model. These two types of variability's are the essence of speech recognition. Compared to template-based approach, hidden Markov modeling is more general and has a firmer mathematical foundation.

Two novel HMM based techniques that segregate a speech segment from its concurrent background are discussed. The first method can be reliably used in clean environments while the sec-

ond method, which makes use of the wavelets denoising technique, is effective in noisy environments. These methods have been implemented and shown superiority over other popular techniques, thus, indicating that they have the potential to achieve greater levels of accuracy in speech recognition rates.

The application of two biometric techniques, face and speaker identification for use on mobile devices. It has been found that combining speaker and face identification technologies can have a dramatic effect on person identification performance. In one set of experiments, a 90% reduction in equal error rate in a user verification system was achieved when integrating the face and speaker identification systems. In preliminary experiments examining the use of static and dynamic information extracted from video, it was found that dynamic information about lip movement made during the production of speech could be used to complement information from static lip images in order to improve person identification. Degradation in speaker identification rates in noisy conditions can be mitigated through the use of noise compensation techniques and/or missing feature theory. Noise compensation involves the adjustment of acoustic models of speech to account for the presence of previously unseen noise conditions in the input signal. Missing feature theory provides a multi-modal face and Speaker identification mechanism for ignoring portions of a signal that are so severely corrupted as to become effectively unusable. It was demonstrated that a multi-biometric approach could reduce the equal error rate of a user verification system on a hand-held device by up to 90% when combining audio and visual information.

Dynamic information captured from a person's lip movements can be used to discriminate between people, and can provide additional benefits beyond the use of static facial features. The problem of robust speaker identification for hand held devices was addressed and showed the benefits of the posterior union model and the universal compensation techniques for handling corrupted audio data.

Genetic algorithm (GA) was first used to replace Steepest Descent Method (SDM) and make a global search of optimal weight in neural network. The improved GA is then used to train the ANN. We can find that the performance of speech recognition was improved by the later method. The experiment showed that if BPNN is trained by GAs, higher recognition rate is attained. Through out SDM in BPNN speech recognition system, attempting to recognize Chinese speech,the recognition rate up to 91% was achieved. If GA is adopted for training of ANN, the recognition rate can be improved up to 95%. It is shown that the improved GA reveals more excellent learning performance than GA (with two-point crossover) by the experiments. However, the drawback of GA (or improved GA) used to train the ANN is that it will increase training time.

In application such as a voice controlled car audio system, voice commands by the driver are corrupted by audio out of loudspeaker. The proposed method has been implemented on OMAP 2420 TIDSP C55x, with a performance of under 59 Mega Cycles for complete system and is tested in real time. In the car environment the voice commands sent to Car Speech Interface system are corrupted by presence of car audio signal.

The system works by using an acoustic echo canceller to cancel the acoustic echo captured by

the car Microphone. The AEC uses two adaptive filters to cancel individual audio channels to give better performance. Speech activity detector was used to find the activity regions in Mic signal. A proposed FSD module was used to check false alarms due to residual music component by exploiting the fact that correlation between music residual and car audio signal is significant. The proposed system showed a best case CER improvement of around 30.3%.

2.3.1.3 Mel-Scale Filter Bank

The MFCC is a representation defined as the real cepstrum of a windowed short-time signal derived from the fast Fourier transform of the speech signal. In the MFCC, a nonlinear frequency scale is used, which approximates the behavior of the auditory system. The discrete cosine transform of the real logarithm of the short-time energy spectrum expressed on this nonlinear frequency scale is called the MFCC.

The MFCC is the result of a discrete cosine transform of the real logarithm of the short-term energy. Mel scale cepstral analysis is very similar to perceptual linear predictive analysis of speech, where the short-term spectrum is modified based on psycho physically based spectral transformations. In this method, the spectrum is warped according to the MEL scale, where as in PLP the spectrum is warped according to the Bark scale. The main difference between Mel scale cepstral analysis and perceptual linear prediction is related to the output cepstral coefficients. The output cepstral coefficients are then computed based on this model. In contrast Mel scale cepstral analysis uses cepstral smoothing to smooth the modified power spectrum. This is done by direct transformation of the log power spectrum to the cepstral domain using an inverse Discrete Fourier Transform (IDFT). The MFCC has good performances in speech recognition (available from: Feature Extraction).

The main point to understand about speech is that the sounds generated by a human are filtered by the shape of the vocal tract including tongue, teeth etc. This shape determines what sound comes out. In any automatic speech recognition system to extract features i.e. identify the components of the audio signal that is good for identifying the linguistic content and discarding all the other stuff which carries information like background noise, emotion etc. If we can determine the shape accurately, this should give us an accurate representation of the phoneme being produced. The shape of the vocal tract manifests itself in the envelope of the short time power spectrum, and the job of MFCCs is to accurately represent this envelope.

An audio signal is constantly changing, so to simplify things we assume that on short time scales the audio signal doesn't change much. This is why we frame the signal into 20-40ms frames. If the frame is much shorter we don't have enough samples to get a reliable spectral estimate, if it is longer the signal changes too much throughout the frame. The next step is to calculate the power spectrum of each frame. This is motivated by the human cochlea (an organ in the ear) which vibrates at different spots depending on the frequency of the incoming sounds. Depending on the location in the cochlea that vibrates (which wobbles small hairs), different nerves fire informing the brain that certain frequencies are present.

The periodogram spectral estimate still contains a lot of information not required for Automatic Speech Recognition (ASR). In particular the cochlea can not discern the difference between two closely spaced frequencies. This effect becomes more pronounced as the frequencies increase. For this reason we take clumps of periodogram bins and sum them up to get an idea of how much energy exists in various frequency regions. This is performed by our Mel filterbank: the first filter is very narrow and gives an indication of how much energy exists near 0 Hertz. As the frequencies get higher our filters get wider as we become less concerned about variations. We are only interested in roughly how much energy occurs at each spot. The Mel scale tells us exactly how to space our filterbanks and how wide to make them. Let $s[n]$, denoted N samples of speech waveform. The discrete Fourier transform (DFT) $X[k]$ of speech signal as follows as:

$$X[k] = \sum_{n=0}^{N-1} s[n] e^{-j2\pi nk/N}, 0 \leq k \leq N \tag{2.29}$$

Where M filters $(m = 1, ..., M)$ is the number of filters we want, the first filterbank will start at the first point, reach its peak at the second point, then return to zero at the 3rd point. The second filterbank will start at the 2nd point, reach its max at the 3rd, then be zero at the 4th etc. A formula for calculating these is as follows by:

$$H[m,k] = \begin{cases} 0, & if \;\; k < f[m-1] \\ (k - f[m-1]) / (f[m] - f[m-1]), & if \;\; f[m-1] \leq k \leq f[m] \\ (f[m+1] - k) / (f[m+1] - f[m]), & if \;\; f[m] \leq k \leq f[m+1] \\ 0, & if \;\; k > f[m+1] \end{cases} \tag{2.30}$$

which satisfies, $\sum_{m=1}^{M} H[m,k] = 1, \quad k = 0, 1, ..., N-1$

The Mel scale relates perceived frequency, or pitch, of a pure tone to its actual measured frequency. Humans are much better at discerning small changes in pitch at low frequencies than they are at high frequencies. Incorporating this scale makes our features match more closely what humans hear.

The formula for converting from frequency to Mel scale is:

$$M(f) = 1125 \, ln(1 + f/700) \tag{2.31}$$

To go from Mels back to frequency:

$$M^{-1}(m) = 700 \left(e^{(m/1125)} - 1 \right) \tag{2.32}$$

For speech recognition, normally, the number M of filters is from 10 to 20 and the MFCC produced from the first few filters are the most effective in recognition. Once we have the filterbank energies, we take the logarithm of them. This is also motivated by human hearing: we don't hear loudness

on a linear scale. Generally to double the percieved volume of a sound we need to put 8 times as much energy into it. This means that large variations in energy may not sound all that different if the sound is loud to begin with. This compression operation makes our features match more closely what humans actually hear.

The implementation of MFCC can be defined as:

Step1 Assume the signal frequency at 16kHz

Step2 Frame the signal into 20-40 ms frames. 25ms is standard.

Step3 Take the Discrete Fourier Transform of the frame and compute the Mel-spaced filterbank., using the equation 2.25. This is called the Periodogram estimate of the power spectrum. We take the absolute value of the complex fourier transform, and square the result.

Step3 Take the log of each of the 26 energies from step 2.

Step4 Take the Discrete Cosine Transform (DCT) of the 12 log filterbank energies to give 12 cepstral coefficents.

For furher reading, some researcher proposed an environmental sounds recognition system using LPC- Cepstral coefficients for characterization and a back propagation artificial neural network as verification method. The verification percentage was 96.66% although the number of feature vectors was small; specifically two feature vectors were used. The lowest percentages were obtained for noisy sound sources, as car, motorcycles and airplanes.

In other, proposed a key word detection method for continuous speech in noisy environment. In the proposed method, the widely used energy, zero crossing, entropy and MFCCs were extracted to generate an audio feature set. Robust endpoint detection algorithm is also used which makes the feature modify its parameter by adapting to the strength of background noise. Then HMMs are used for the classifiers. Experiments were made under different types of noises and the results show that this method is more accurate and more anti-noise than traditional methods. This method was used in a student management system to recognize some key words.

The feature vector of each voice characteristic parameter is chosen by means of MFCC (Mel Frequency Cepstral Coefficients). The extracting algorithm of syllable parts from continuous voice signal is introduced. It shows the relationship between recognition rates and number of applying syllables and number of groups for applying syllables. The core engine of the HMM method is described, and simple syllables were used for the recognition process. In order to achieve a high recognition rate for different syllables, significant quantitative information of syllables is required. MFCC parameters were used. MFCC with a mel frequency index of 24 provides a higher recognition rate (96% per 72 syllables). Speaker dependent recognition requires only a mel frequency index of 14 during training in comparison to the 24 required for speaker independent recognition training. Based on the results of this study, more words can be added frequently to the database. By increasing the number of voice samples being trained, HMM can be widely applied to real life applications and, ultimately, a voice recognition system can be produced.

2.3.1.4 Discrete Cosine Transform

DFT-based algorithms are the most active and popular among the number of transform based algorithms which are proposed in the past for single channel speech enhancement. One of the famous spectral subtraction algorithms which was extended to the Fourier transform by Boll has became a very popular method. Fourier domain is another important area of speech enhancement. In this more noise reduction takes place it also reduced the level of tonal noise as compared to spectral subtraction. The fast Fourier transform is the property of DFT which is used reduces the computation load.

Another widely used transform method is Karhunen Loeve Transform (KLT), which has been applied to speech enhancement and it is used in one of the subspace speech enhancement algorithms. The main drawback of KLT-based algorithms is the high computational complexity. DCT provide higher energy compaction as compared to DFT. Its performance is very similar to KLT. There is no fast Fourier transform method possible in KLT. But it has high energy compaction. Therefore DCT is widely used instead of KLT and also it has fast Fourier transform algorithm. Unlike the DFT the DCT coefficient are real and there is no phase component. Therefore, DCT should be a good choice for speech enhancement.

In other opinion, DCT is ortogonal transform and is expressed as a sequence of finite data point in terms of sum of cosine function oscillating at different frequencies. DCT is equivalent to DFT to approximately twice the length oprating on real data with even symmetry. Since its introduction in 1974, it has been developed to secure data compression via transform coding techniques, e.g. image (JPEG), and speech (MPEG). The original for defining the DCT was that its basis set provided a good approximate to the eigenvectors of the class of matrices that constitutes the autocovarieance matrix of a first order Markov process, with the result that DCT had a better performance than DFT.

The same principle governs the usefullness of DFT and other transforms used for signal compression the smoother a function is the fewer terms in its DFT or DCT are required to represent it accurately giving more compression. However, the implicit periodicity's of DFT means that a discontinuity usually occurring at the boundaries of any random segment of the signal is unlikely to have the same value at both the left and the right boundaries. In contrast a DCT where both the boundaries are always even yield a continuous extension at the boundaries. The DCT is equivalent to the KLT of a first oder Markov process as correlation coefficient ρ tends to approach unit.

The other opinion about DCT, two new types of DCT's, which are konown as even discrete cosine transform (EDCT)and odd discrete cosine transform (ODCT). In other hand, the symmetric version of DCT where basis set approaches of eigenvectors of KLT of first order Markov process as block size N tends to infinity. Finally, there are exits eight types of DCT and classified them in even and odd transforms. Thus, there are four odd and four even versions of DCT's, which are numbered from I to IV with letter E and O indicating even or odd transform, respectively.

Since cosine are both periodic and have even symmetry it can be represented as shown in Figure 2.3.4 with four examples of symmetric periodic extension of a four point sequence. The original

finite-length sequence is shown in each subfigure as the samples with solid dots. These sequences are all periodic and also have even symmetry, In each case finite length sequence is easily extracted as the first four points of one period. For convenience, the periodic sequences obtained by replicating with 16 each of the four subsequences in figure 2.3.4 (a), (b), (c), (d) are denoted as $\widetilde{x_1}[n]$, $\widetilde{x_2}[n]$, $\widetilde{x_3}[n]$, $\widetilde{x_4}[n]$ respectively. It is noted that $\widetilde{x_1}[n]$ has period $(2N-2) = 6$ and even symmetry about both n=0 and $n = (N-1) = 3$. The sequence $\widetilde{x_2}[n]$ has period $(2N) = 8$ and even symmetry about half sample point n = $-\frac{1}{2}$ and $\frac{7}{2}$. The sequence $\widetilde{x_3}[n]$ has period $(4N) = 16$ and even symmetry about n=0 and n=8. The sequence $\widetilde{x_4}[n]$ and even symmetry about the half sample point n = $-\frac{1}{2}$ and n = $\left(2N - \frac{1}{2}\right) = \frac{15}{2}$.

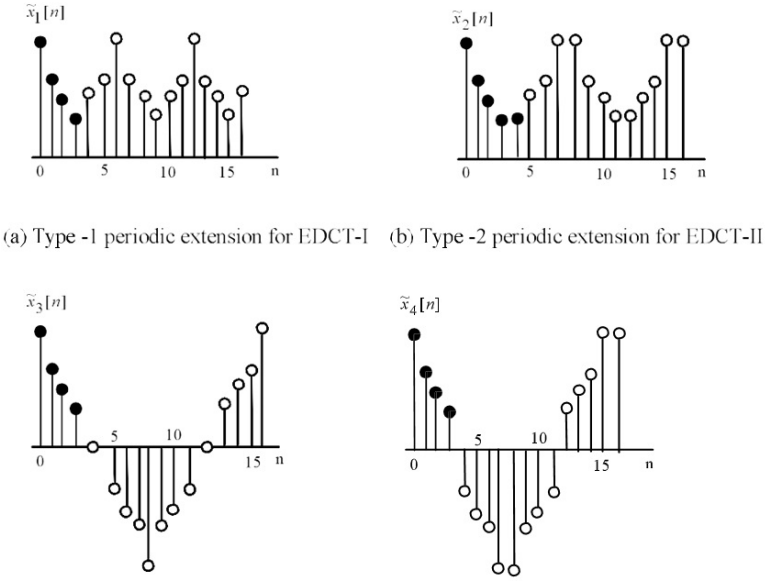

Figure 2.3.4: Four ways to represent for point sequences both periodically and symmetrically

DCTs of types's I-IV imply boundaries that are even/odd around either a data point for both boundaries or halfway between two data points for both boundaries. DCTs of types V-VIII imply boundaries that even/odd around a data point for one boundary and halfway between two data points for better the other boundary. Of theses, DCT-1 and DCT-II are the most commonly used transforms. In general, DCT algorithm are classified into two categories: indirect computation and direct computation. Indirect computation algortihm adopts the fast fourier transform that we used in this thesis. On other hand, direct computation algorithms use techniques such as matrix factorization, divide and conquer method, recursive decomposition and small odd-length DCT modules which are derived from Winograd's small modules of real-valued discrete fourier transform (DFT's). The formulation of DCT can be evaluated as follows:

$$X_K = \sum_{n=0}^{N-1} x(n) \cos\left(\frac{2\pi kn}{N}\right), 0 \leq k \leq N-1 \tag{2.33}$$

$$where \;\; x(n) = \frac{1}{N}\sum_{k=0}^{N-1} x(k) \cos\left(\frac{2\pi kn}{N}\right) \tag{2.34}$$

A common characteristic of most images is that the neighboring pixels are highly correlated and therefore contain highly redundant information. The foremost task is to find an image representation in which the image pixels are decorrelated. Redundancy and irrelevancy reductions are two fundamental approaches used in compressions. Where as redundancy reduction aims at removing redundancy from the signal source (image or video), irrelevancy reduction omits parts of the signal that will not be noticed by the signal receiver.

2.3.1.5 Spectogram

A sound spectrogram (or sonogram) is a visual representation of an acoustic signal. To oversimplify things a fair amount, a Fast Fourier transform is applied to an electronically recorded sound. This analysis essentially separates the frequencies and amplitudes of its component simplex waves. The result can then be displayed visually, with degrees of amplitude (represented light-to-dark, as in white=no energy, black=lots of energy), at various frequencies (usually on the vertical axis) by time (horizontal), shows in figure 2.3.5.

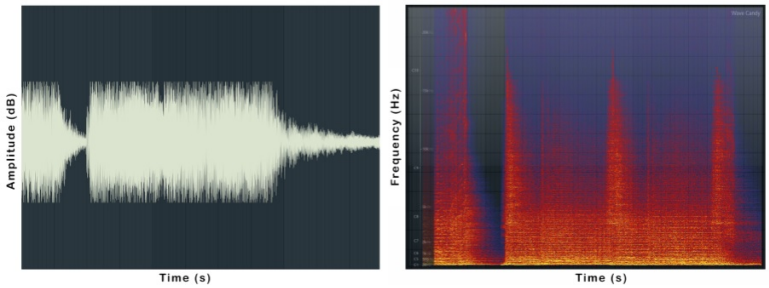

Figure 2.3.5: Random Audio signal (left) and its corresponding spectrogram (right)

The spectrogram in Figure 2.3.4 shows the transition between two loud sections of music, separated by a cut-off that is signified in the spectrogram by the lack of any colour bars in that section. The lower frequencies are more used, and typically louder, tapering off as the frequency gets higher. This is common in a lot of music since percussion and bass lines tend to play throughout and therefore overlap on the lower frequencies of other instruments. This causes the defect shown, wherein lower frequencies which are more common overlap each other and are therefore louder overall. In this case, the music being used as an example is particularly intensive in terms of number of instruments, causing less clearly defined harmonics to show up within the spectrogram itself.

However, before spectrograms can be analysed and compared to each other, they must first be generated from the given audio signal. The modern way of generating spectrograms is using Fast Fourier Transform which relies on samples of digital data being Fourier transformed in order to compute the degree of size of the frequency spectrum in the signal. In a more practical sense, this is done by first splitting the audio signal into chunks of equal size. Each chunk corresponds to a single column in the spectrogram, or more accurately the range of amplitude against frequency for a specific segment in time. The chunks are computed using the squared magnitude of the signal's Short-time Fourier Transform (STFT), and then laid side by side to form the overall three-dimensional representation. In most cases, the chunks tend to overlap to preserve continuity.

Depending on the size of the Fourier analysis window, different levels of frequency/time resolution are achieved. A long window resolves frequency at the expense of time—the result is a narrow band spectrogram, which reveals individual harmonics (component frequencies), but smears together adjacent 'moments'. If a short analysis window is used, adjacent harmonics are smeared together, but with better time resolution. The result is a wide band spectrogram in which individual pitch periods appear as vertical lines (or striations), with formant structure. Generally, wide band spectrograms are used in spectrogram reading because they give us more information about what's going on in the vocal tract, for reasons which should become clear as we go. The equation to calculate a continuous-time STFT is given below for a time variable t, x is the frequencies of signal being transformed, the spectrogram can be calculated by:

$$spectrogram\,(x,t) = |STFT\,(x,t)|^2 \tag{2.35}$$

About time-frequency imaging there are a spectrogram represented by STFT and a scalogram represented by wavelet. The former is restrained by the uncertainty principle of time-frequency because it divides a signal into short pieces. Since former has the same observation time width in all frequency-range, its time-resolution and the frequency-resolution are uniform. The later changes the observation time-width according to frequency. In the case of low frequency, the observation time-width is lengthened and the frequency-resolution becomes small. Conversely, in the case of high frequency, the time-resolution becomes small and the frequency-resolution becomes large. The time resolution and frequency-resolution of wavelet transform are not uniform.

Further, the inverse transform to the time domain signal is possible in STFT or wavelet transform. On the other hand, Wigner distribution that is Fourier transform of time-domain signal correlation is also known as a time frequency analyzing method. And in Wigner distribution, there are a Cohen class based on the spectrogram, and an affine class based on the scalogram. But the inverse transform to the time-domain signal is impossible.

3

Deep Belief Network: Case Study Pattern Recognition

3.1 A Brief History of Pattern Recognition

Deep learning has found applications in various areas of information processing such as audio processing, natural language modeling and processing, object recognition and computer vision because of the high accuracy of its models. Deep Belief Networks (DBNs) learn complex function mapping from input to output directly from raw pixels of data. DBN training, which includes the pre-training and fine-tuning processes, in conventional central processing units (CPU) platforms is computationally expensive because multiple hidden layers with high number of hidden units are required to train the mapping function for high dimensional raw pixel data.

Large amounts of training data are needed to learn the parameters of such a network for preventing overfitting. Execution of Deep Learning algorithms can be made faster by reducing the dimensionality of the data, as illustrated in recent years. One such approach is presented in which illustrates the use of Discrete Cosine Transform (DCT) for image classification. Wavelet transform, rough set theory, and artificial neural networks are combined together to form a hybrid image classification method in. Multiresolution image features have been utilized for object detection in.

The first description modeling a Deep Belief Network (DBN) was published in 1986[10]. At

the time, the model was referred to as "harmonium". It is a type of deep neural network composed of multiple layers, each layer consisting of visible neurons representing the layer input, and hidden neurons representing the layer output. The visible neurons will be owned by the preceding layer, for which these neurons are hidden. The visible neurons are fully interconnected with the hidden ones. The distinctive feature of a DBN is that there are no connections between the visible neurons and no connections between the hidden neurons. The connections are symmetric and are exclusively between the visible neurons and the hidden ones. In the following process, start with a definition of a stochastic neuron, which will be used in the DBNs. Then we will describe the architecture of Restricted Boltzmann Machines, which represent the main building block of DBNs.

Further demonstrated that deep architectures have the merits over shallow architectures in terms of model expressiveness and efficiency. This exponential efficiency which is required to represent energy functions stands out as the major contribution of the deep architecture. Moreover, with the greed layer-wise unsupervised pre-training, the weights of the model will be better initialized in a region in the vicinity of a good local optimum. This strategy helps the optimization and generalization, giving rise to energy function that are high-level abstractions of the lower layers[59].

Learning a single layer model is well-documented in the case where we know both the correct outputs of the layer as well as the input to the layer. However, in the case of deeper networks with one or more hidden layers, by its very definition the input to and output from the hidden layers are unknown. A neural network consists of a number of units which have simple nonlinear transfer functions and approximation capability for a number of kinds of complex problems comparative small number of calculation. Therefore, neural networks are applied to data analysis, data mining and data classification.

A sufficient learning cannot be performed if the size of the network is too small. Adversely, overfitting occurs to the learning data and it loses generalization ability if the size is too large. Therefore, the appropriate structure of the neural network is required to be determined for each target problem for higher performance of a neural network. Traditionally, structure of a neural networks are determined through a trial and error procedure based on the experiences of a designer of the neural networks. However, a huge computation time is required by such determination process.

On other opinion, several structural optimization methods of the neural networks simultaneously with learning are proposed. Hayashida et al.[42] propose a structural optimization method for Recurrent Neural Network (RNN) by introducing two stage taboo search, one of the metaheuristic solutions. Here, they define that the structure of a RNN is determined by the number of inputs, the number of intermediate layer units, the number of feedback layers. Hayashida et al.[50] proposed a structural optimization for a combined neural network model of a Feedforward Neural Network (FNN) and an Auto Encoder (AE) which performs dimension compression to remove extra data and redundant data. Their procedure optimize the number of input data, the number of units of the middle layer of AE, and the number of units of the hidden layer of FNN by using tabu search. Auto encoders can be stacked to form a deep network by feeding the internal representation (output code) of the Auto encoder at the layer below as input to the considered layer. The unsu-

pervised pre-training of the architecture is done one layer at a time. Internally, it has a hidden layer **h** that describes a **code** used to represent the input. The network may be viewed as consisting of two parts: an encoder function $h = f(x)$ and a decoder that produces a reconstruction $y = g(h)$. This architecture is presented in figure 3.1.1. If an auto encoder succeeds in simply learning to set $g(f(x)) = x$ everywhere, then it is not especially useful. Instead, auto encoders are designed to be unable to learn to copy perfectly. Usually they are restricted in ways that allow them to copy only approximately, and to copy only input that resembles the training data. Because the model is forced to prioritize which aspects of the input should be copied, it often learns useful properties of the data.

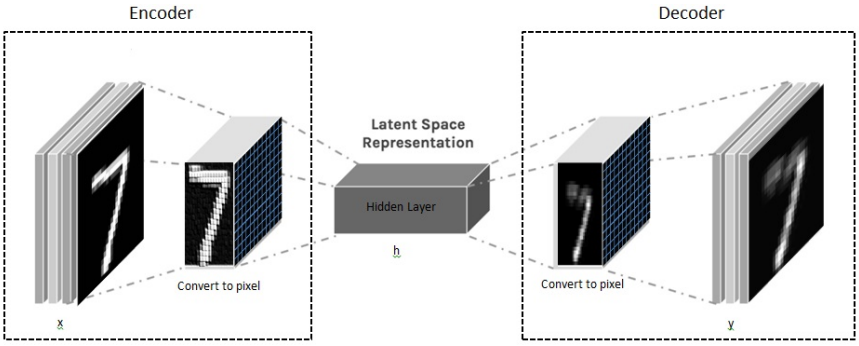

Figure 3.1.1: Architecture of Auto Encoder

The idea of auto encoders has been part of the historical landscape of neural networks for decades[22]. Traditionally, auto encoders were used for dimensionality reduction or feature learning. Recently, theoretical connections between auto encoders and latent variable models have brought auto encoders to the forefront of generative modeling. Auto encoders may be thought of as being a special case of feed-forward networks and may be trained with all the same techniques, typically minibatch gradient descent following gradients computed by back-propagation.

Comparing the structure of either a stacked autoencoder (SA) or a stacked denoising autoencoder (SdA) (they are equal) with the structure of a deep belief network (DBN), one can see that the fundamental differences consist in the output layer and the direction of the connections between layers. In the first architecture the output layer is separate from the input layer but in the second architecture it coincides with it. This is also remarked by the arrows connecting the layers. In a SA or SdA the information flows unidirectionally from the input layer, through the hidden layer, up to the output layer. In a DBN the information flows both ways between the visible (input/output) layer and the hidden layer.

Unlike general feed forward networks, auto encoders may also be trained using re-circulation[46], a learning algorithm based on comparing the activations of the network on the original input to the activations on the reconstructed input. Re-circulation is regarded as more biologically plausible than back-propagation but is rarely used for machine learning applications.

Copying from data input to data output sounds like it's useless, but we are usually not interested in the decoder output. Instead, we hope that training auto encoders to perform input copying tasks will result in **h** taking useful properties. One way to get useful features from auto encoder is to limit the hidden value (**h**) has a dimension smaller than input value (**x**). Automatic encoding maker whose code dimensions are less than the input dimension is called **undercomplete**. Learning the complete representation below forces the auto encoder to capture the most prominent features of training data.
The learning process is described simply as minimizing a loss function:

$$L\left(x, g\left(f\left(x\right)\right)\right) \tag{3.1}$$

where L is a loss function penalizing $g\left(f\left(x\right)\right)$ for being dissimilar from **x**, such as the mean squared error (MSE). When the decoder is linear and L is the mean squared error, an undercomplete auto encoder learns to span the same subspace as PCA. In this case, an auto encoder trained to perform the copying task has learned the principal subspace of the training data as a side effect.

Undercomplete auto encoders with smaller code dimensions than the input dimension, can study the most prominent features of data distribution. It can be observed that these auto encoders fail to learn something useful if the encoder and decoder are given too much capacity. A similar problem occurs if the hidden code is allowed to have the same dimensions as the input, and in the case **overcomplete** where the hidden code has a dimension larger than the input. In this case, even the linear encoder and linear decoder can learn to copy input to the output without learning anything useful about data distribution.

Ideally, a system can successfully train each auto encoder architecture, choose code dimensions and encoder and decoder capacities based on the complexity of the distribution to be modeled. Authorized auto encoders provide the ability to do so. Rather than limiting the capacity of the model by keeping encoders and shallow decoders and small code sizes, regular auto encoders use a loss function that drives the model to have other properties besides the ability to copy input to its output. Other properties include the sparsity of the representation, the small derivative of the representation, and the robustness of the sound or missing input. Auto encoders that are regulated can be nonlinear and overcomplete but still learn something useful about data distribution, even if the capacity of the model is large enough to learn trivial identity functions.

Like many other machine learning algorithms, auto encoders exploit the idea that data concentrates around a low-dimensional manifold or a small set of such manifolds. Some machine learning algorithms exploit this idea only insofar as they learn a function that behaves correctly on the manifold but that may have unusual behavior if given an input that is off the manifold. Auto encoders take this idea further and aim to learn the structure of the manifold.

To understand how auto encoders do this, we must present some important characteristics of manifolds. An important characterization of a manifold is the set of its **tangent planes**. At a point x on a *d*-dimensional manifold, the tangent plane is given by *d* basis vectors that span the local directions of variation allowed on the manifold. All auto encoder training procedures involve a

compromise between two forces:

Step 1 Learning a representation **h** of a training example **x** such that **x** can be approximately recovered from **h** through a decoder. The fact that **x** is drawn from the training data is crucial, because it means the auto encoder need not successfully reconstruct inputs that are not probable under the data generating distribution.

Step 2 Satisfying the constraint or regularization penalty. This can be an architectural constraint that limits the capacity of the auto encoder, or it can be a regularization term added to the reconstruction cost. These techniques generally prefer solutions that are less sensitive to the input.

At present, constrastive divergence is one of the most popular gradient approximations for RBMs. However, there are multiple alterations to the standard CD algorithm, and it is not obvious which is the best one. A very common alternation is Persistent CD, abbreviated as PCD. Some research claims that PCD produces more meaningful feature detectors, and outperforms the other variants of CD algorithms. PCD algorithm uses a different approximation for sampling states than CD. It eliminates Step 3 when compared to standard CD described in the above steps. This step initializes the hidden states based on the input pattern. As a result, the sampling chain is being restarted for every observed pattern.

Instead, PCD initializes the hidden states only once at the beginning of the training. This causes the single chain of sampling throughout the whole training, which helps move faster towards the model distribution p_model rather then p_data. The smaller the learning rate, the better PCD works. It is because the smaller parameter updates are then small enough compared to changes in the sampling chain (mixing rate of Markov chain), and the chain can easier catch up to the changes in the model.

The Restricted Boltzmann Machines themselves are capable of detecting and extracting features from input data. Several layers of RBMs can be stacked one onto each other to form a multilayer network. Each RBM layer uses the hidden neurons from preceding RBM layer as its input (see figure). The deep architecture with multiple layers can then extract deep hierarchical representation of the training data

A set of RBM layers performs the feature detection task, while the classification task is performed by using a multilayer perceptron as the last layer. The last MLP layer is using by hidden neurons of the preceding RBM layer as its input. The result of DBN architecture is a mixture of probabilistic neurons in the feature extraction phase, and deterministic neurons in the classification phase. The DBN architecture consists of RBM and MLP layers. However, RBMs employ unsupervised learning, while MLPs employ supervised learning. In genereal theory DBN has a two-phase training process.The first phase, called pretraining, performs unsupervised training of each RBM layer separately. The second phase uses the gradient descend method for supervised training of the MLP as well as RBM layers.

The pretraining and training general process can be summarized as follows:

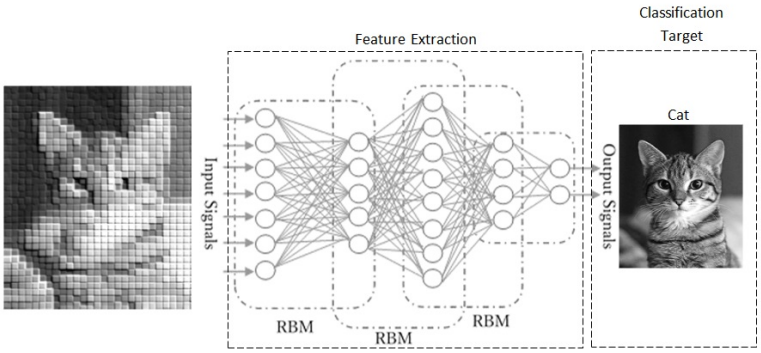

Figure 3.1.2: Architecture of a Deep Belief Network

Step1 The network is initialized with small random weights, biases and other parameters.

Step2 The first RBM layer is initialized with input data representing potentials in its visible neurons. Then the unsupervised training is performed on this layer iterating over the training dataset for predefined number of epochs.

Step3 The next layer obtains its input by sampling the potentials generated in the hidden neurons of the previous layer. Then the unsupervised training is performed on this layer iterating over the training dataset.

Step4 Iterate the previous step for the desired number of layers. In each iteration the samples are propagated upwards deeper into the network. The pretraining phase is finished when the first MLP layer is reached.

Step5 Fine-tuning via supervised gradient descend starts. The training is stopped after reaching a predefined number of epochs, or is finished successfully after reaching the target error rate.

The structurally of neurons of the DBN are interconnected in the same manner as the MLP network. This allows the second phase of training to be performed exactly as if training MLP network. Therefore the whole training procedure is equivalent to initializing the weights and biases of a deep MLP network with the values obtained in the unsupervised probabilistic pretrainig. After the network is trained, the classification of the presented input data is performed exactly like in the case of MLP network.

Because of the performance of a neural network should be measured based not only on the learning accuracy but also the generalization capability, a neural network is evaluated based on the degree of error between both the training data and data for verification of the generalization capability in above mentioned optimization method. In order to evaluate a neural network, it is necessary to divide the known data into training data and for verification of the generalization capability. However, even in a method such as cross validation, the data may be biased and the network cannot be evaluated well. Nishida et al.[42] has improved the method of Hayashida et al. [42], they

use the Self Organization Map (SOM) to convert the data mapped onto a 2 D plane by k-means method and divide them into training data and data for verification of the generalization capability. Though such data generation method, they succeed in reducing the bias of features between divided data, and improving learning accuracy and improving generalization capability.

Deep Neural Network (DNN) consists of a lot of multiple layers, and the data analysis performance such as data prediction, data classification, or data mining is dramatically improved compared with conventional neural networks such as Feedforward Neural Network (FNN). Therefore, a lot of applications of DNN are reported in the various study fields. A neural network composed of many layers is difficult to learn properly by back propagation, however, the learning procedure of DNN is constructed for appropriate learning by applying apply pretraining, drop out and so forth. Various models of DNN such as Convolutional Neural Network (CNN), Deep Belief Network (DBN), are proposed. This paper focuses on DBN which has a structure with multiple layers of Restricted Boltzmann Machine (RBM). DBN has succeeded in acquiring higher data analysis capability by effectively incorporating a feature extraction process which is conventionally performed by trial and error.

In DBN, multiple RBMs were incorporated into the learning process as feature extractors. DBN performs feature extraction with unsupervised learning called Pre-training and supervised learning called fine-tuning are performed based on the extracted features. From the structural characteristics of DBNs, it can be considered that there exists a great relationship between the structure of DBM, the number of hidden layers and units constituting each layer, and the performance in data classification or prediction. Performance improvement is expected by giving an appropriate structure corresponding to input data. This paper proposes a new method for highly accuracy and efficient structure optimization for DBNs. Additionally, this paper compares the proposed method and the conventional methods by the numerical experiments, and verifies the effectiveness of the proposed method.

3.2 Method

This paper proposes a structure optimization method with parameters of each hidden layer and unit numbers of each layer of RBMs constituting DBN. The proposed method includes local search based on tabu search for structural optimization, modularization for improving of RBMs the learning efficiency which is required for structural evaluation of DBN and enormous calculation time. Furthermore, number of hidden layers and the number of units are optimized separately to reduce the search space. The DBN structure optimization method proposed in this paper is shown below:

Step 1 Optimize number of hidden layers

Step 1-1 Let $n \varepsilon [\underline{n}, \overline{n}]$ as the hidden layer dimensional.

Step 1-2 Item A DBN with n hidden layers is evaluated based on the training and generalization capability. Here, the number of units of each layer is 500.

Step 1-3 If $E_n > E_n^*$, then update the best solutin of $E_n^* = E_n$, $n^* = n$.

Step 1-4 $\overline{n} > n$, let $n = n+1$ and return to step 1-2. Otherwise let n^* be number of hidden layer.

Step 2 Optimize number of units of each layer (Rough search)

Step 2-1 Let m_i be number of units of i-th layer, $i = 1, 2, \ldots, n^*$ and let t=0

step 2-2 Divide search range of number units of i-th hidden layer, $\left[\underline{x_i}, \overline{x_i}\right]$ into k_i subrange. Let $d_i^{j_i} \equiv \left[\underline{x_i^{j_i}}, \overline{x_i^{j_i}}\right]$ be j_i th subrange, $j = 1, 2, \ldots, k_i$, $\underline{x}_i^1 = \underline{x}_i \overline{x}_i^{-k_i}$, as the dimensional.

step 2-3 Let $x_1^j, x_2^j, \ldots, x_{n^*}^j$ be the center of gravity of the subrange j, and let $x^j \equiv \left(\widehat{x}_1^j\right), \left(\widehat{x}_2^j\right), \ldots, \left(\widehat{x}_{n^*}^j\right) = \left(\left\lfloor \dot{x}_1^j + 0.5 \right\rfloor, \left(\left\lfloor \dot{x}_2^j + 0.5 \right\rfloor, \ldots, \left(\left\lfloor \dot{x}_{n^*}^j + 0.5 \right\rfloor\right)\right.\right.$ be representative point of subrange j.

step 2-4 Evaluate the structural of the neural network to the point of x^j. Then evaluate E^j as the subspace of j.

step 2-5 Choose a subspace with highest evaluation value. Then selected space value as "θ.

step 2-6 If $x_i^{-\Theta} - \underline{x}_i^{\Theta} \leq k_i, \forall iort = T_l$ then go to step 3. Otherwise, generate next search range at representatives $\left[x_i^{j-1}, x_i^{j+1}\right]$ in the neighboring subspace centered "θand go to step 2-2.

Step 3 Optimize number of units of each layer (Detailed search using Tabu search)

step 3-1 Randomly generate an initial solution

step 3-2 Evaluate each neighbor of the current solution

step 3-3 Add solution using Tabu List

step 3-4 Search the neighboring space with the best solution

step 3-5 find the best solution and terminate it, and goto step 3-2.

To optimize the structure of a DBN, considering the number of hidden layers and the number of units simultaneously is required. However, in this paper, after optimizing the number of hidden layers of DBN first, optimize the number of units of each hidden layer. Even when structural optimization is conducted in such order, verification experiments on the relation between DBN structure and data prediction accuracy are conducted to verify whether same structure are obtained or not, compared to a optimization procedure such that both are taken into consideration simultaneously. In the experiment, the accuracy for unknown data D_3 is calculated by using DBN where the number of units of each n hidden layer is fixed to 500.

Additionally, the accuracy for D_3 is calculated by using DBN where the number of units of each n hidden layer is optimized by the method described in the next section. The 3-category image data is used for the experiment, and the classification accuracy for the data D_3 is set as the verification result. In other words, DBN showed that the superiority and inferiority relationship of performance

based on the number of hidden layers is the same irrespective of whether the number of units of each hidden layer is optimized. In the experiment corresponding to figure 2.2.3, only experiments using 3-category image data are shown. However, similar relationships are observed in other prior experiments. Therefore, the optimal number of hidden layers of DBN is determined that is with the highest performance fixing the number of each layer as 500 first. Subsequently, the numbers of units of all hidden layers are optimized.

As the number of components of the neural network increases, the accuracy for learning data increases. However, the generalization ability for unknown data decreases. On the contrary, if the constituent elements of a neural network are small, the features of the target data cannot be properly learned. Therefore, it is desirable that the structure of a neural network is not only the prediction error with respect to the learning data, but also the prediction error with respect to the data not used for learning. In this paper, the target data is divided into three and evaluate the network structure by the following procedure.

Firstly, divide the target data D into three dataset (D_1, D_2, D_3) for learning, generalization verification, and test. Error back propagation is conducted using data D_1 and the verify generalization capability is evaluated based on output error when input data of D_2 is given to the learned neural network. Let e_{tr} be training error for dataD_1 and e_{ve} be the output error for data D_2 i.e e_{ve} represents the generalization capability. Let T be a number of data, M be the number of units of the output layer, $O_j(t)$ be the output value from the j-th, unit in the output layer of the neural network at the period t and $Y_j(t)$ be the j-th factor of the target value. The error is defined by the mean square error between the target value and the output of neural network output as:

$$e_A = \frac{1}{T}\sum_{j=1}^{M}\sum_{i=1}^{T}\left(O_j(i) - Y_j(i)\right)^2, \ A = tr, ve \tag{3.2}$$

Based on these criteria of error, structure of a neural network A^k is evaluated by:

$$E\left(A^k\right) = \frac{1}{e_{tr} + e_{ve}} \tag{3.3}$$

To optimize the hidden layer of structural optimization in DBN, let $\underline{n}$ and $\overline{n}$ be minimum and maximum number of hidden layers, respectively. In the related literature [81] 500 is employed for the number of units of each hidden layer of DBN. Similarly, this paper employs 500 for the number of units of each layer in Step 1. Set the initial number of hidden layers be $\underline{n}$ and the number of hidden layers is added one by one up to $(n = \underline{n}, \underline{n}+1, \underline{n}+2, \ldots, \overline{n}-1, \overline{n})$ hidden layers such that the number of units of each hidden layer is 500 is evaluated based on the evaluation function. A DBN with highest evaluation value is selected and let n^* the corresponding number of hidden layers.

After the number of hidden layers n^* of DBN is determined, the number of units of each hidden layer should be determined for optimizing of number of units of each layer. Let $(x_1, x_2, \ldots, x_{n^*})$ be a n^* dimensional solution in solution space $X = \prod_{i-1}^{n^*}[\underline{x}_i, \overline{x}_i]$ where $x_i \in [\underline{x}_i, \overline{x}_i]$ is the number of

hidden layer. There exits numerous solutions in the space X. Therefore, the search space is divided and generate small areas to search and realize efficient search by the following procedure. Here, let k_i be the number of division of the dimension of the solution space corresponding to the i-th hidden layer.

Let $d_i^{j_i} \equiv \left[\underline{x}_i^{j_i}, \overline{x}_i^{j_i}\right], j_i = 1, 2, \ldots, k_i, x_i^1 = \underline{x}_i, x_i^{-k_i} = \overline{x}_i$ be the j-th interval of subspace, where $J = (j_1, j_2, \ldots, j_{n^*})$. Select the center of gravity $x_1^j, x_2^j, \ldots, x_{n^*}^j$ as a representatives point of subspace D_j. A subspace with highest evaluation value among the representatives point of $\prod_i k_i$ subspaces is defined as $j^* = (j_1^i, j_2^i, \ldots, j_{n^*}^i)$.

In general, a pair of neural networks which have similar structures have similar performance to each other. In this paper, the optimal network structure is chosen by tabu search which is one of evolutionary computation methods based on the neighbor search, from the solution subspace divided according to the above mentioned procedure.

The DBN consists of connected plurality of RBMs and FNN. The learning process of the parameters such as connection weight and biases are performed in order from the RBMs closer to the input layer. In this paper, in order to avoid the redundancy of the structure evaluation in the structure optimization procedure of the DBN, a RBM utilizes the past learning information of another RBM which has common structure partially, input information to the RBM, number of units of a hidden layer, and the number of hidden layers, by following procedure.

For example, consider DBN1 including η_1, hidden layers which is the learning process is completed and DBN2 including η_1 hidden layers which is not completed. Let $\eta_1 \geq min\left\{\eta_1, \eta_2\right\}$ assume that the number of units of each hidden layer from the first layer to $(\eta - 1)$-th the layer of DBN1 and DBN2 are all the same, and the number of unitsof hidden layer of the η-th layer is different. Learning from the remaining the η-th to theη_2-th layer of DBN2 is performed by using the connection weights and biases of each hidden layer from the first to the $(\eta - 1)$-th layer of DBN1 in the learning process of DBN2. This mechanism improves learning efficiency of DBNs with different network structures.

3.3 Experiment

In fact, deep learning provides an efficient multi-level image representation, which can learn the image semantic structure information progressively from the low-level features to the high-level structures. For instance, given an image, the most sensitive information for the human brain is the shape, colour, and background of the image. However, the data of one image that can be recognized by one computer is just pixel, the content of the image cannot be understood by the computer. Using simple machine learning algorithm may be helpful, but the effect is not obvious. Because simple machine learning algorithm has shallow learning structure, if we wish the computer can respond like a human brain, deep machine learning is the most efficient method.

Just like other machine learning methods, deep learning can be divided into supervised learning and unsupervised learning. DBN is a kind of unsupervised deep learning model. It can be viewed

as the stacking of multiple RBMs on each other, creating "deep"networks which can capture high-level dependency among the input visible units.

A DBN is built based on RBMs. The first RBM is trained on the input data. Then, the second RBM is based on the output of the first one, and so on, until a sufficiently deep architecture is created. RBMs are restricted connectivity and stochastic generative artificial neural networks. From a theoretical viewpoint, RBMs are interesting because they are able to discover complex regularities and find notable features in raw data.

Based on this deep architecture, a novel image retrieval method with four phases is proposed: first, an unsupervised pre-training phase will be executed, which sets the network weights roughly; second, a fine-tuning phase is performed in order to update the network weights to the local optimum through BP algorithm on labelled data; third, classify the samples in the image library by the classifier; and finally, the query image will be input into the model composed of DBN. After obtaining the label of query image, the similar images in the same class are returned.

In **fine-tuning stage**, we the backpropagation (BP) algorithm through the whole deep model to fine-tune the parameters for global optimal. BP algorithm is commonly used to compute the gradients for all the layers of the stacked RBMs in one iteration. Thus, BP can also be used to greatly improve the performance of stacked RBMs. From a high-level perspective, fine-tuning treats all layers of stacked RBMs as a single model,so that we can improve all the weights in the stacked RBMs in one iteration. Considering BP algorithm can be extended to apply for any number of layers. We can actually use this algorithm on stacked RBMs of arbitrary depth.

After the **fine-tuning stage**, one model with optimal parameters (including weights and biases) is obtained. In order to realize image retrieval, we first classify the image data-set with the trained model which consists of DBN. The classification procedure is given as follows.

Step1 First, all images in the data-set are resized to the 30×30 dimension, and then perform image binarization, so that the size is accordant to the input layers.

Step2 Extract features from the raw pixels through the trained DBN model, and get the highest layer representation of these original images. Then, classify the image using taboo search to search the optimum value.

In the **Pre-training Phase** most existing deep models initialize the parameter space in a random manner and approximate a local optimal solution gradually by learning. Unfortunately, a bad initial parameter space may lead to a poor local optimum and thus affect the following learning procedure seriously. To solve this problem, in the pre-training phase of our model, an efficient greedy layerwise training algorithm is utilized to train multiple RBMs from the visible input layer up to the output layer, so that the network weights and the biases can be placed to suitable neighbourhoods in the parameter space.

In DBN, each layer consists of a number of units (usually hundreds or thousands), which calculates the received data independently. There is no connection between each two units in the same

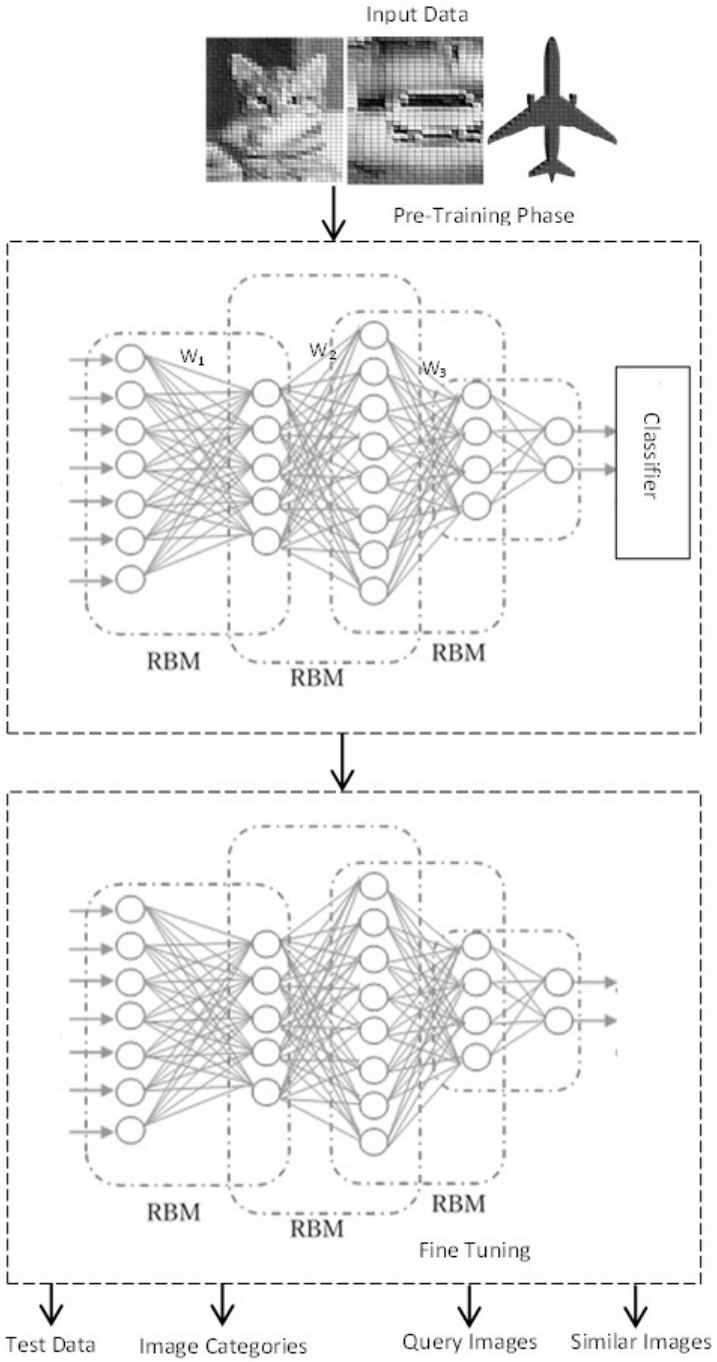

Figure 3.3.1: The procedure of the proposed image retrieval method based on DBN.

layer. The first layer (also called input layer) is in charge of receiving data (such as image data) from the outside world. The first layer and the second layer constitute a typical RBM. In order to achieve

the energy balance of this RBM, unsupervised learning method is used to finetune the network parameters. Then, the output of the second layer is considered as the input of the third layer, and the second and third layers constitute a new RBM.

In the new RBM, in order to achieve energy balance, network parameters are adjusted continually and RBMs are continuing to be stacked layer by layer greedily to find a great parameter space. When the unsupervised learning layer by layer is finished, and then executes supervised learning according to the original input and target output on the entire network.

3.4 Result and Discussion

In this section Caltech101[35] are used as benchmark of image classification in many related literature for dataset in experimental design. The image data of the Caltech 101 are gray scale 30×30 grids images, and each grid is scaled in the range of $[0, 1]$. Images are classified into 4 categories "airplane", "cat", "face", "dolphin". There are 65 images per a category. This paper performs the following 2 kinds of experiments using Caltech101 as follows:

1 3-category classification experiment using 3 categories, "airplane", "cat", and "face".

2 4-category classification experiment using 4 categories.

Here, the number of hidden layers of a DBN without structural optimization is set to 3, and the number of each hidden layer unit is 500, 500, 2000. The number of hidden layer of a FNN without structural optimization is set to 3, and the number of hidden layer units was set to 200, 200 and 800, respectively. For a data classification problem with m categories, the number of units of an output layer is set to m and that data is classified into a certain category corresponding to the unit such that output value is maximum in all output units. The experiments are conducted 10 trials, and the average value of classification accuracy is shown in Table 1 as experimental results.

Table 3.4.1: Image Classification Test: Result (accuracy (%))

No	Method	3-Category	4-Category
1	DBN with structural optimization (Proposed method)	**85.0**	**74.7**
2	DBN without structural optimization	77.1	62.2
3	FNN with structural optimization	75.2	61.8
4	FNN without structural optimization	59.8	40.1

From Table 1, the proposed method has the highest performance, and this experimental result indicates that the proposed method succeed to discover the appropriate structure of DBNs to increase the data classification accuracy. In the case of 3-category classification, structure of all DBNs obtained by the proposed method have 3-layer structure in all 10 trials. The average value of the number of units of the hidden layers are 454.7, 1834.5, and 2935.9 from the closer to the input layer, respectively. In each trial, numbers of units of hidden layers are similar to each other. Also,

in the case of 4-category classification, DBNs with a 5-layer structure are obtained in all 10 trials. The average value of the number of units of the hidden layers are 457.0, 212.9, 2046.9, 1109.5, and 5974.9 from the closer to the input layer, respectively. Same as 3-category classification, almost same structure are obtained in all trials.

In the structure optimization of DBN by the proposed method, the solution space are divided into multiple subspaces first and solution search procedure are performed intensively in the promising regions. Such searching process can realize both diverse and intensive solution search and stably discover appropriate structure. Additionally, it is also successful to improve the computational efficiency by modularization focusing on that the DBN has a structure in which a plurality of RBMs are superimposed. After optimizing the number of hidden layers of DBN first, optimize the number of units of each hidden layer. Even when structural optimization is conducted in such order, verification experiments on the relation between DBN structure and data prediction accuracy are conducted to verify whether same structure are obtained or not, compared to a optimization procedure such that both are taken into consideration simultaneously. In the experiment, the accuracy for unknown data D_3 is calculated by using DBN where the number of units of each n hidden layer is fixed to 500. The 3-category image data is used for the experiment, and the classification accuracy for the data D_3 is set as the verification result. The experimental results are shown in figure 3.4.1.

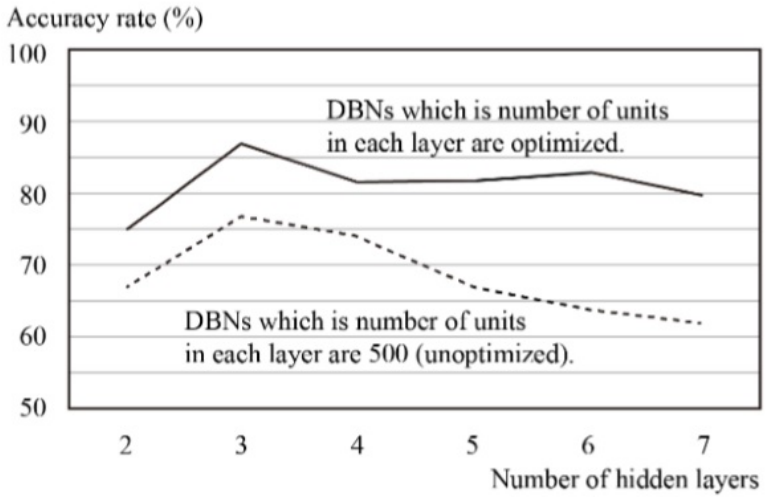

Figure 3.4.1: Relation between optimized and unoptimized number of units of each hidden layer

From figure 3.4.1, the data classification accuracy is highest when the number of hidden layers is 3 in both of two types of variation experiments indicated by a broken line and a solid line. In other words, DBN shows that the superiority and inferiority relationship of performance based on the number of hidden layers is the same irrespective of whether the number of units of each hidden layer is optimized. In the experiment corresponding to figure 3.4.1, only experiments us-

ing 3-category image data are shown. However, similar relationships are observed in other prior experiments. Therefore, the optimal number of hidden layers of DBN is determined that is with the highest performance fixing the number of each layer as 500 first. Subsequently, the numbers of units of all hidden layers are optimized.

Let $d_i^{j_i} \equiv \left[\underline{x}_i^{j_i}, \overline{x}_i^{j_i}\right], j_i = 1, 2, \ldots, k_i, x_i^1 = \underline{x}_i, x_i^{-k_i} = \overline{x}_i$ be the j-th interval of subspace, where $J = (j_1, j_2, \ldots, j_{n^*})$. Select the center of gravity $x_1^j, x_2^j, \ldots, x_{n^*}^j$ as a representatives point of subspace D_j. A subspace with highest evaluation value among the representatives point of $\prod_i k_i$ subspaces is defined as $j^* = (j_1^i, j_2^i, \ldots, j_{n^*}^i)$.

Let a superior rectangular paralleled piped whose vertices are $x_1^{j^*-1}, x_1^{j^*+1}, x_2^{j^*-1} x_2^{j^*+1}, \ldots, x_{n^*}^{j^*-1}$, $x_{n^*}^{j^*+1}$ be a new search space and repeat the above steps until $x_i^{-j_i} - \underline{x}_i^{j_i}, \leq k_i, \forall_i$ is satisfied. In other words, the division of solution space is completed, the optimal solution of the network structure is searched by using taboo search. As an example, the procedure of division of the solution space with $n^* = 2$ is shown in figure 3.4.2.

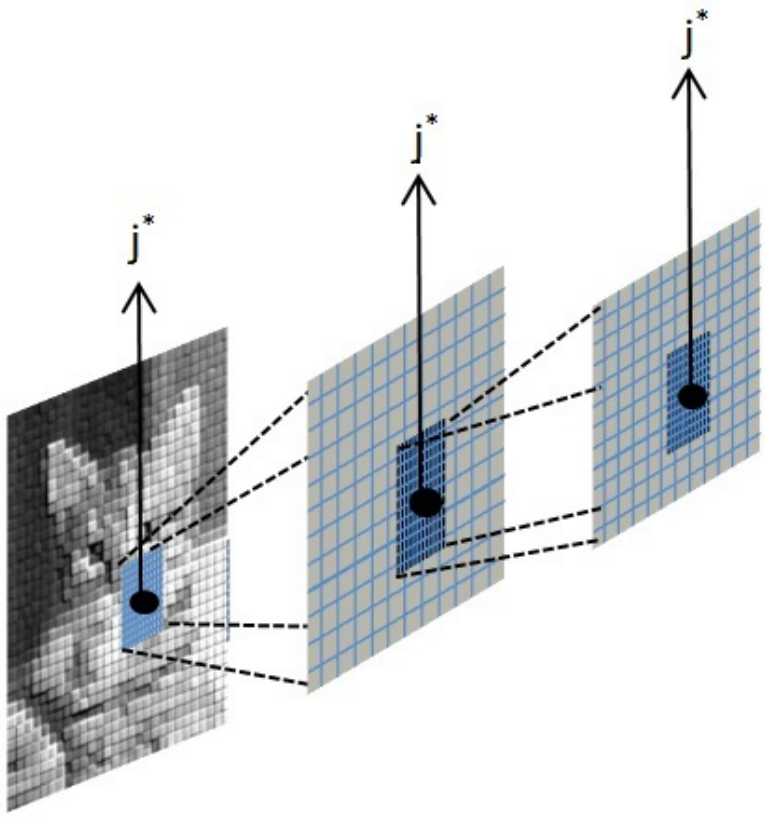

Figure 3.4.2: Division of the Solution Space ($n^* = 2$)

3.5 Conclusion and Future Work

This paper proposes structure optimization method for a DBN (Deep Belief Network) which consists of multiple RMBs (Restricted Boltzmann Machine) and a FNN (Feedforward Neural Network). The features of the proposed method are that it realizes searching both in wide range of solution areas by division of solution space and intensive search by tabu search, introduces modularization of RBMs to improve the calculation efficiency drastically by reducing the calculation amount in solution search. Numerical experiments using multiple categories image data indicates that it succeed in obtaining appropriate structure of DBN with high data classification accuracy by the proposed structural optimization method for DBNs. To develop a network structure optimization method that supports data analysis for high dimensional time series data such as voice data can be one of the future works.

4

Structural Optimization using DBN: Case Study Speech Recognition

4.1 Introduction of Speech Recognition System

Time series is a special kind of data which has ordered sequence. Ordering property is crucial since it effects dependency of data points and meaning of data. Data points forming the time series have some more pre-defined properties. One of the properties is that data points need to be obtained through repeated measurements over time at equally spaced intervals. The time interval of data points should be continuous and each time unit observations should have at most one data point.

Time series are used in various areas, such as statistics, economics, pattern recognition, control engineering, signal processing, astronomy, meteorology, entertainment and so on. Time series analysis has been developing and trending research areas for decades. Despite the progress, there are a lot of open topics about time series.When analyzing the time series previously mentioned unique properties should be taken in consideration. In order to reduce challenge in analyzing data, features may need to be transformed into invariant feature space.

Speech is the time series audio form for communication in human behaviors, which is spoken continuously states. In the speech recognition the continuously spoken states convert an acoustic signal, it captured by a microphone or telephone to a length of words. The characteristics of signal it reflects the different speech sound being spoken. The information from a speech that we are

gathering is represented by a spectrum of amplitude from speech waveform. Based on this speech characteristic allows us to recognize the feature information from the waveform of the speech signal. Recognizing word from the acoustic signal is a though work, many researchers are emerging in the area of speech recognition and signal processing[17].

Speech Recognition as well-known as computer speech recognition is the process learning from the computer to understand our spoken by an algorithm implemented as a computer program. The main goal of the speech recognition area is developing speech recognition technology and systems into machines. The basic communication from a human being is a speech of human speech ability of the machine, the desire to automate simple tasks requiring machine interaction with humans in an automatic speech [63].

Nowadays, the application in tasks that require human-machine interfaces, such as automatic call processing in telephone networks, and query-based information systems find widespread in the statistical modeling of speech, automatic speech recognition systems. That provides updated travel information, stock price quotations, weather reports, data entry, voice dictation, access to information: travel, banking, commands, transcription, disabled people (blind people) supermarket, railway reservations, etc.[49]. Speech recognition technology was increasingly used within telephone networks to automate as well as to enhance the operator services[11].

In the sixth decades of speech, recognition area has attracted many researchers' attention for the reason of curiosity about technology and the mechanism of realization. The speech feature extraction is a key issue for all classification methods to obtain better generalization. The extracted features should minimize the distances between samples with the same speech class and maximize the distances between samples with the different speech classes[80]. If the features are not well defined, the best classifier could have difficulty in reaching the good performance. Most typical features are predefined by hand-engineered ones, including newly proposed nonlinear dynamic features[3].

They have achieved the great success in specific fields where the small speech training data can be available only. However, these features perform inconsistently on different speech recognition task. They are on the lower level to make themselves difficult to extract and organize the discriminative features from the speech signals. It is not clear which speech features are most powerful in distinguishing recognition. They are easily influenced by speakers, speaking styles, sentences, and speaking rates because these factors directly affect the extracted speech features such as pitch and energy contours[71]. Besides, they are not easily tuned for the newly coming speech signals-pitch and energy contours. Besides, they are not easily tuned for the newly coming speech signals.

Recently, the development of machine learning based on speech has been made in a deep neural network similar as a neural network. One of approaching algorithm in deep neural networks is Deep Belief Network (DBN)[45]. For example, speech signal utilizes the higher level features to represent the more abstract concepts. This is the reason that they succeed in breaking most of the world records of the recognition tasks. Among deep learning methods, deep belief network (DBN) is the most representative one. It applies the unsupervised learning algorithms such as

auto-encoders and sparse coding to learn higher level feature representations from the unlabeled data. It has produced the state-of-the-art results on recognition and classification tasks[9]. On the other hand, typical classification methods used for speech recognition include hidden Markov model (HMM)[16], Gaussian Mixture Model (GMM)[78], artificial neural networks such as recurrent neural network (RNN[87], support vector machine (SVM)[90], and the fuzzy cognitive map network[55]. These methods are confronted with the complicated decision boundary of the classification.

In such case, the ensemble learning can be applied that can learn any nonlinear boundary through appropriately combining the simple classifiers. It has potential ability to reduce over fitting problems greatly, to decrease the risk of a single classifier, and to obtain better performance than its single classifiers. The usual ensemble classifiers are boost-based, bagging-based approaches, random subspace, and so forth. Some of them have been applied to perform speech recognition but still fail to reach the performance as expected. For example, it seems that random forest and AdaboostDT have the bad effect for speech classification. The possible reason is that the diversity of the base classifiers is not guaranteed. As to random subspace, the classifiers trained with different features should have certain diversity inherently. However, in the neural networks (NN) are prone to over fitting. Especially, the deep neural networks in some cases where the training data are not abundantly clear. For instance, there are two different features sets, but the classifiers trained by the two features sets may have the similar classification results, leading to no rich diversity between them. To ensure the diversity among base classifiers, the features in random subspace should be further abstracted from different viewpoints using DBN.

Finally, obtaining large amounts of labelled time series training data may be expensive,resourceful and difficult. Whereas,huge amount of unlabelled data can be easily obtained in various areas. Hence, current shallow-structured methods needing large amount of label led training data can not be used for most of the time series data. Deep learning networks using unsupervised learning have gotten highly successful results. However, so as to get more accurate results the architecture of the model should be adjusted or modified respecting the characteristics of time-series.

Based on this deep architecture, a novel image retrieval method with four phases is proposed: first, an unsupervised pre-training phase will be executed, which sets the network weights roughly; second, a fine-tuning phase is performed in order to update the network weights to the local optimum through BP algorithm on labelled data; third, classify the samples in the image library by the classifier; and finally, the query image will be input into the model composed of DBN. After obtaining the label of query image, the similar images in the same class are returned.

In **fine-tuning stage**, we the backpropagation (BP) algorithm through the whole deep model to fine-tune the parameters for global optimal. BP algorithm is commonly used to compute the gradients for all the layers of the stacked RBMs in one iteration. Thus, BP can also be used to greatly improve the performance of stacked RBMs. From a high-level perspective, fine-tuning treats all layers of stacked RBMs as a single model,so that we can improve all the weights in the stacked RBMs in one iteration. Considering BP algorithm can be extended to apply for any number of

layers. We can actually use this algorithm on stacked RBMs of arbitrary depth.

After the **fine-tuning stage**, one model with optimal parameters (including weights and biases) is obtained. In order to realize image retrieval, we first classify the image data-set with the trained model which consists of DBN. The classification procedure is given as follows.

Step1 First, all images in the data-set are resized to the 30×30 dimension, and then perform image binarization, so that the size is accordant to the input layers.

Step2 Extract features from the raw pixels through the trained DBN model, and get the highest layer representation of these original images. Then, classify the image using taboo search to search the optimum value.

In the **Pre-training Phase** most existing deep models initialize the parameter space in a random manner and approximate a local optimal solution gradually by learning. Unfortunately, a bad initial parameter space may lead to a poor local optimum and thus affect the following learning procedure seriously. To solve this problem, in the pre-training phase of our model, an efficient greedy layerwise training algorithm is utilized to train multiple RBMs from the visible input layer up to the output layer, so that the network weights and the biases can be placed to suitable neighbourhoods in the parameter space.

In DBN, each layer consists of a number of units (usually hundreds or thousands), which calculates the received data independently. There is no connection between each two units in the same layer. The first layer (also called input layer) is in charge of receiving data (such as image data) from the outside world. The first layer and the second layer constitute a typical RBM. In order to achieve the energy balance of this RBM, unsupervised learning method is used to finetune the network parameters. Then, the output of the second layer is considered as the input of the third layer, and the second and third layers constitute a new RBM.

In the new RBM, in order to achieve energy balance, network parameters are adjusted continually and RBMs are continuing to be stacked layer by layer greedily to find a great parameter space. When the unsupervised learning layer by layer is finished, and then executes supervised learning according to the original input and target output on the entire network.

The literature review presents an evolutionary computational method for speech recognition, which is composed of the DBN and Tabu Search. Hayashida et al.[42] describes a number of subspace the implementation of tabu search is applied. Each subspace can be directly fed into DBN to generate the high-level features. The rest of this thesis is organized as follows: In Section 4.2, several related works are briefly introduced about speech recognition techniques, DBN and RBM. The evaluated system and some experiments on simple voice dataset are presented in Section 4.3. Then Section 4.4 describes about the results and discussion. Finally, Section 4.5 concludes this thesis.

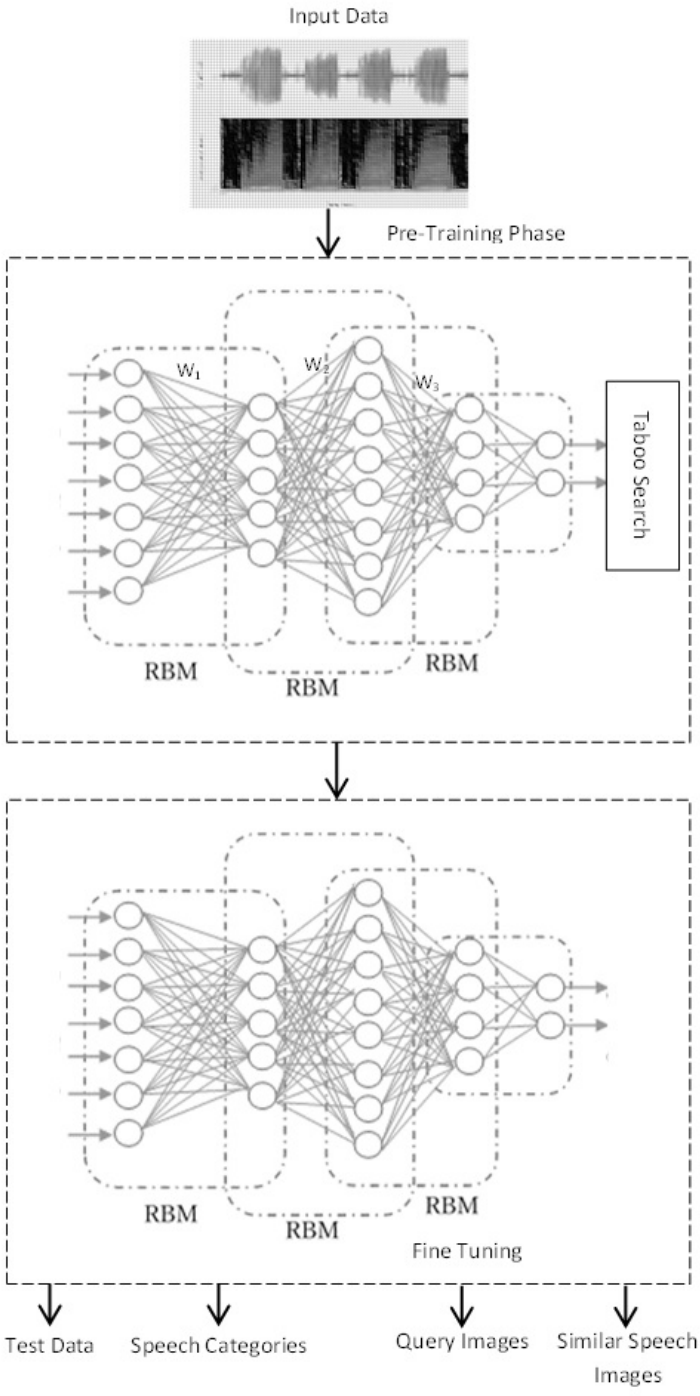

Figure 4.1.1: The procedure of the proposed speech retrieval method based on DBN.

4.2 Method of Feature Extraction in Speech Signal

Here we review the more recent work of speech signal extraction in developing a similar type of auto encoder for extracting bottleneck speech instead of image features. Discovery of efficient binary codes related to such features can also be used in speech information retrieval. Importantly, the potential benefits of using discrete representations of speech constructed by this type of deep auto encoder can be derived from an almost unlimited supply of unlabeled data in future-generation speech recognition and retrieval systems.

The deep auto encoder with three hidden layers is formed by "unrolling" the DBN using its weight matrices. The lower layers of this deep auto encoder use the matrices to encode the input and the upper layers use the matrices in reverse order to decode the input. This deep auto encoder is then fine-tuned using backpropagation of error-derivatives to make its out put as similar as possible to its input.

After learning is complete, any variable-length spectrogram can be encoded and reconstructed as follows. First, N-consecutive overlapping frames of 256 point log power spectra are each normalized to zero mean and unit-variance to provide the input to the deep autoencoder. The first hidden layer then uses the logistic function to compute real valued activations. These real values are fed to the next, coding layer to compute "codes". The real valued activations of hidden units in the coding layer are quantized to be either zero or one with 0.5 as the threshold.

These binary codes are then used to reconstruct the original spectrogram, where individual fixed frame patches are reconstructed first using the two upper layers of network weights. Finally, overlap and add technique is used to reconstruct the full length speech spectrogram from the outputs produced by applying the deep auto encoder to every possible window of N consecutive frames.

The deep auto encoder described above can extract a compact code for a feature vector due to its many layers and the non linearity. But the extracted code would change unpredictably when the input feature vector is transformed. It is desirable to be able to have the code change predictably that reflects the underlying transformation invariant to the perceived content. This is the goal of transforming auto encoder proposed in for image recognition. The building block of the transforming auto encoder is a "capsule", which is an independent sub network that extracts a single parameterized feature representing a single entity, be it visual or audio. A transforming auto encoder receives both an input vector and a target output vector, which is related to the input vector by a simple global transformation; e.g., the translation of a whole image or frequency shift due to vocal tract length differences for speech. An explicit representation of the global transformation is known also.

The bottle neck or coding layer of the transforming auto encoder consists of the outputs of several capsules. During the training phase, the different capsules learn to extract different entities in order to minimize the error between the final output and the target. In addition to the deep auto encoder architectures described in this section, there are many other types of generative architectures in the literature, all characterized by the use of data alone (i.e., free of classification labels) to

automatically derive higher-level features.

In this section, the most widely studied hybrid deep architecture of DNNs, consisting of both pretraining (using generative DBN) and fine-tuning stages in its parameter learning. Part of this review is based on the recent publication of. As the generative component of the DBN, it is a probabilistic model composed of multiple layers of stochastic, latent variables. The unobserved variables can have binary values and are often called hidden units or feature detectors. The top two layers have undirected, symmetric connections between them and form an associative memory. The lower layers receive top-down, directed connections from the layer above. The states of the units in the lowest layer,or the visible units, represent an input data vector.

There is an efficient, layer-by-layer procedure for learning the top-down, generative weights that determine how the variables in one layer depend on the variables in the layer above. After learning, the values of the latent variables in every layer can be inferred by a single,bottom-up pass that starts with an observed data vector in the bottom layer and uses the generative weights in the reverse direction.

DBNs are learned one layer at a time by treating the values of the latent variables in one layer, when they are being inferred from data, as the data for training the next layer. This efficient, greedy learning can be followed by, or combined with, other learning procedures that fine-tune all of the weights to improve the generative or discriminative performance of the full network. This latter learning procedure constitutes the discriminative component of the DBN as the hybrid architecture.

Discriminative fine-tuning can be performed by adding a final layer of variables that represent the desired outputs and backpropagating error derivatives. When networks with many hidden layers are applied to highly structured input data, such as speech and images, backpropagation works much better if the feature detectors in the hidden layers are initialized by learning a DBN to model the structure in the input data as originally proposed in [88]. A DBN can be viewed as a composition of simple learning modules via stacking them. This simple learning module is called RBMs that we introduce next.

An RBM is a special type of Markov random field that has one layer of (typically Bernoulli) stochastic hidden units and one layer of (typically Bernoulli or Gaussian) stochastic visible or observable units. RBMs can be represented as bipartite graphs, where all visible units are connected to all hidden units, and there are no visible–visible or hidden– hidden connections.

An undirected graphical model called a Gaussian-binary RBM is built that has one visible layer of linear variables with Gaussian noise and one hidden layer of 500–3000 binary latent variables. After learning the Gaussian-binary RBM,the activation probabilities of its hidden units are treated as the data for training another binary–binary RBM. These two RBMs can then be composed to form a DBN in which it is easy to infer the states of the second layer of binary hidden units from the input in a single forward pass.

Stacking a number of the RBMs learned layer by layer from bottom up gives rise to a DBN. The stacking procedure is as follows. After learning a Gaussian–Bernoulli RBM (for applications with

continuous features such as speech) or Bernoulli, Bernoulli RBM (for applications with nominal or binary features such as black–white image or coded text), the activation probabilities of its hidden units as the data for training the Bernoulli RBM one layer up. The activation probabilities of the second-layer Bernoulli RBM are then used as the visible data input for the third-layer Bernoulli RBM,and soon.

Some theoretical justifications of this efficient layer-by-layer greedy learning strategy is given in[46], where it is shown that the stacking procedure above improves a variational lower bound on the likelihood of the training data under the composite model. That is, the greedy procedure above achieves approximate maximum likelihood learning. Note that this learning procedure is unsupervised and requires no class label.

When applied to classification tasks, the generative pretraining can be followed by or combined with other, typically discriminative, learning procedures that fine-tune all of the weights jointly to improve the performance of the network. This discriminative fine-tuning is performed by adding a final layer of variables that represent the desired outputs or labels provided in the training data. Then, the backpropagation algorithm can be used to adjust or fine-tune the DBN weights and use the final set of weights in the same way as for the standard feedforward neural network. What goes to the output, label layer of this DBN depends on the application. For speech recognition applications, the output of DBN feature can represent either syllables, phones, sub-phones, phone states, or other speech units used in the HMM-based speech recognition system $(l_1, l_2.l_3, \ldots, l_L)$, see figure 4.2.1.

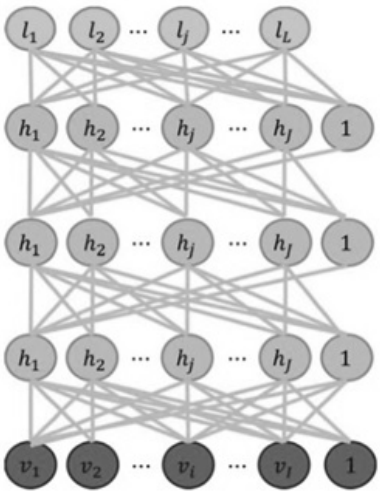

Figure 4.2.1: General Illustration of DBN

Further research has also shown the effectiveness of other pretraining strategies. As an example, greedy layer-by-layer training may be carried out with an additional discriminative term to the generative cost function at each level. And without generative pretraining, purely discriminative train-

ing of DNNs from random initial weights using the traditional stochastic gradient decent method has been shown to work very well when the scales of the initial weights are set carefully and the mini-batch sizes, which trade off noisy gradients with convergence speed, used in stochastic gradient decent are adapted prudently (e.g., with an increasing size over training epochs).

Discriminative "pretraining" is used for positioning a subset of weights in each module in a reasonable space using parallelizable convex optimization, followed by a batch-mode "fine tuning" procedure, which is also parallelizable due to the closed-form constraint between two subsets of weights in each module. Further, purely discriminative training of the full DNN from random initial weights is now known to work much better than had been thought in early days, provided that the scales of the initial weights are set carefully, a large amount of labeled training data is available and mini-batch sizes over training epochs are set appropriately. Nevertheless, generative pretraining still improves test performance, sometimes by a significant amount especially for small tasks. Layer-by-layer generative pretraining was originally done using RBMs, but various types of auto encoder with one hidden layer can also be used.

A DBN discussed above is a static classifier with input vectors having a fixed dimensionality. However, many practical pattern recognition and information-processing problems, including speech recognition, machine translation natural language understanding, video processing and bioinformation processing, require sequence recognition. In sequence recognition, sometimes called classification with structured input/output, the dimensionality of both inputs and outputs are variable.

Also, randomization order in creating mini-batches needs to be judiciously determined. Importantly, it was found effective to learn a DBN by starting with a shallow neural net with a single hidden layer. Once this has been trained discriminatively (using early stops to avoid overfitting), a second hidden layer is inserted between the first hidden layer and the labeled softmax output units and the expanded deeper network is again trained discriminatively. This can be continued until the desired number of hidden layers is reached, after which a full backpropagation "fine tuning" is applied.

Any approaches or methods that take information from training samples to design a classifier can be called learning. Supervised, unsupervised and reinforced learning are the general forms of machine learning. Before explaining the learning types, the data types should be first distinguished clearly. Learning can be seen as a special technique to reduce the error on a training set.

Labelled and unlabelled data are treated differently with respect to the scope of applications. While labelled data are commonly used to predict or estimate the target attributes(or class labels)of new observations (or samples, or data points),unlabelled data are useful for the clustering or investigating associations in data. Attributes of the data excluding the label information are often called features (or attributes) in machine learning.

If class labels of training data are not available, the learning process is called unsupervised learning. Main purpose of unsupervised learning is finding out or underlying similarities, grouping the training samples or detecting the association rules between data points. The grouping procedure is often known as clustering task. RBMs are trained using unsupervised learning. They are not

performing classification themselves, but instead they are able to learn to reconstruct data in an unsupervised fashion.

Input variables define the measurement types used in the learning methods; some of the measurement types are suitable for qualitative input variables whereas some of them are favorable for quantitative inputs. Not only the measurement types but also the model itself is defined for the type of the input attributes. In order to use models that are invented for qualitative variables, grouping or binning the quantitative inputs is mandatory.

Variables, which are also known as features or attributes, are categorized into four main types as nominal, ordinal, interval, and ratio variables. Nominal variables and ordinal variables are called categorical variables. Nominal variables provide descriptive information labels to distinguish one object from another, e.g. red, green, blue. Ordinal variables are similar to the nominal variables, however ordinal variables have a meaningful order and can be arranged in that order, e.g. low, medium, high. Ranges between the ordinal variables may not be equally spaced in ordinal variables.

Interval variables take numerical values which are equally spaced, such as temperatures scale or calendar dates. But the interval variables are not suitable for the proportional calculation. The ratios between the interval variables are not meaningful. Ratio variables are similar to interval variables except that the ratio calculations are meaningful and the origin or zero value represents the absence of measured characteristic.

During the data gathering process some faults can be observed and this failure can negatively affect the learning process. Hence obvious outlier or missing values should be discarded or readjusted before the learning operations to increase performance or reduce the effort. In addition to that, features within different dynamic ranges should be normalized or standardized on demand to equalize the influences in the cost function.

In the Machine learning, we have to deal with high dimensional feature space which means many input attributes for each observation (or sample). This phenomenon can reveal some serious complications and affect negatively the design of the learning task.

Little or nothing in the way of data reduction is provided, which leads to severe requirements for computation time and storage. With sufficient observation samples, some of the dimensionality problems can be overcome, but not all of them. Besides, the number of samples required may be very large. The demand for samples can increase exponentially or power law growth with the high-dimensional feature space. This serious difficulty is often called the curse of dimensionality.

One of the key reasons for the curse of dimensionality is that high-dimensional calculations have more tendency to the potential computational and storage problems than low-dimensional ones. The other one is that it creates noise and hides the real patterns and makes the classification problem more intractable.

The ability of classification of test data, which have not been seen by the model yet, is called generalization. Generalization with small or rate is a main aim of all kind of learning methods. Generalization property of a learning method can be checked by validating the model with dif-

ferent and separate test sets. While trying to reduce training set classification /regression errors, complexity of the classifier should be adjusted by taking generalization issue into consideration.

Over trained complex models may perform perfect classification on the training set during learning process,but not on the new data. This phenomenon is known as overfitting. There is quite important trade off between highly complex structure that tends to overfitting and simple structure producing poor classification result on novel observation samples. Therefore this problem is one of the important research challenge in machine learning and pattern recognition.

As mentioned earlier, one of the major problem is curse of dimensionality in machine learning and pattern recognition. Moreover, limited resource and computational complexity are the major concern in these areas. Hence, dimensionality reduction also known as feature reduction methods are proposed. To design and tune a successful classifier with better generalization ability, reducing the number of dimensions is also critical, especially when limited number of samples(or observation, data points) exist for training process. Besides, noise elimination and outlier detection are other positive capabilities of the feature reduction methods.

The experimental method in this thesis, especially in speech recognition there are so many techniques to store in traditionally of voices signal such as .mp3, .wav, .mid, etc. First step in segmentation technique of continuous speech signal is to digitalize the signal. The short-term energy function for the digitalized signal is calculated. FFT (Fast Fourier Transformation) of the energy function is found. Length of the window4 is approximately adjusted to the length of the signal so that obtained result is closer to the number of phonemes in the taken speech signal. Various windowing techniques are applied to the continuous speech signal and its performance is evaluated.

The resultant FFT is raised to the power of 0.01 so that the magnitude spectrum calculated is brought to the optimized value. Next step is to invert the derived signal. Discrete and Adaptive filters are applied, and then the minimum phase group delay is found for the filtered signal. The results are compared to conclude with the best filter. Hence, the graph is plotted, which is the minimum phase group delay function of the signal. Positive peaks on the graph relate to the phoneme boundary.

Speech is a moving signal. When we speak, our articulatory apparatus (the lips, jaw, tongue, and velum) modulates the air pressure and flow to produce an audible sequence of sounds. Although the spectral content of any particular acoustic signal in speech may include sequences up to several thousand hertz, our articulatory configuration (vocal-tract shape, tongue movement, etc.) often does not undergo dramatic changes more than 10 times per second. The acoustic properties of a waveform corresponding to a phone can vary greatly depending on many factors - phone context, speaker, style of speech, etc.

4.3 Experiments

This section describes the experimental design in speech audio files. The main process in the speech recognition is feature extraction, it would be reduced variability of spoken words signal. Particu-

larly, eliminating various information, such as whether the sound is voiced or unvoiced, it eliminates the effect of periodicity or pitch, the amplitude of excitation signal and also the fundamental frequency etc. The feature extraction techniques for speech recognition describes about reducing dimensionality of input vector while maintaining the discriminating power of signal. Many researchers have some point of important work in speech recognition area [40].

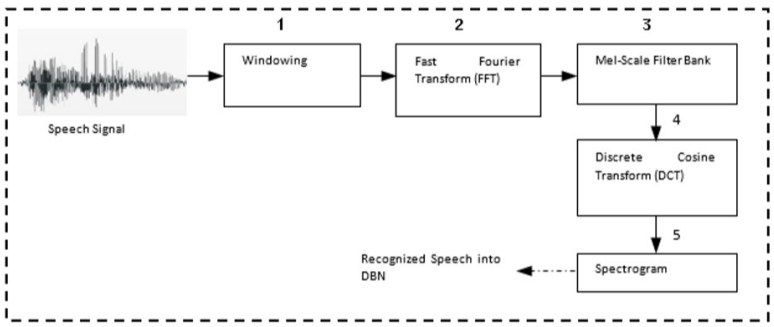

Figure 4.3.1: Proposed Model in Speech Recognition

In the **windowing** process speech is commonly dynamic time series signal in which the composition of properties changes very quickly over time. Before extracting the speech signal from analog to digital, at this stage we do frame blocking, the speech signal is divided into several frames with a general length of 20-30 ms containing N samples of each frame separated by M (M < N) where M is the number of shifts between frames. The first frame contains the first N sample. The second frame begins the sample M after the start of the first frame, so this second frame overlaps the first frame as much as the N-M sample. Frame blocking is necessary because the voice signal changes over a period of time.

Windowing techniques are mainly used in the process of designing digital filters. In order to convert an impulse response of infinite duration to a Finite Impulse Response (FIR) filter design windowing is performed. Symmetrical sequences of Window functions generated for digital filter design. Those window functions are usually an odd length with a single maximum at the center. For spectral analysis5, Windows for DFT/FFT are formed by removing the right-most coefficient of an odd-length, symmetrical window. Truncated sequences are known as periodic. When the truncated sequence is periodically extended, the deleted coefficient is commendably restored (by a virtual copy of the symmetrical left-most coefficient). Window technique consists of a function called window function which is nothing but if some interval is chosen, it returns with finite non-zero value inside that interval and zero value outside that interval. Using formulation in 2.24 and 2.25 in the chapter 2, the result shows in figure 4.2.2 and 4.2.3.

In the windowing process to minimize the discontinuity that occurs in the signal, which is caused

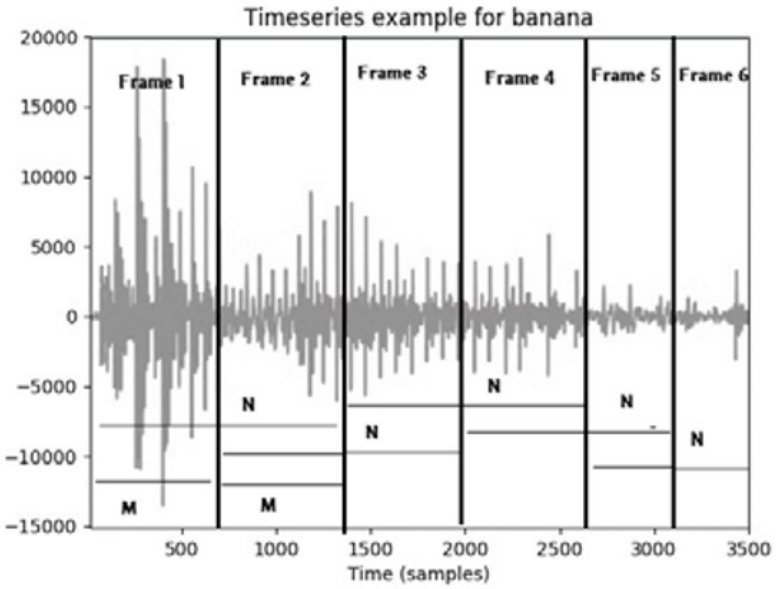

Figure 4.3.2: Frame blocking in windowing function

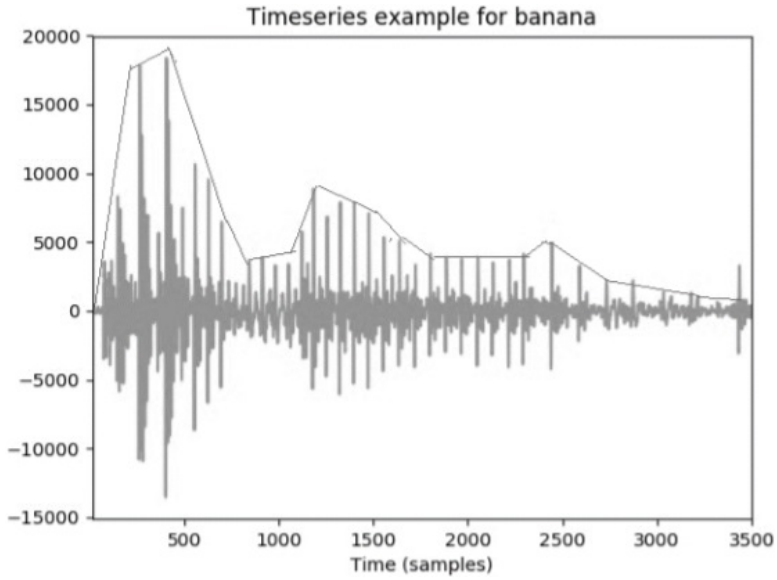

Figure 4.3.3: Hamming Window Process Example from Analog Signal

by spectral leakage when the frame blocking process is done where the new signal, has a different frequency with the original signal. The concept of windowing is to taper the end of the signal to zero at the beginning and end of each frame. By using windowing functions, the ability of an FFT to extract spectral data from signals can further enhance. Windowing functions act on raw data to reduce the effects of the leakage that occurs during an FFT of the data. The windowing process is

multiplying each frame from the type of window used.

When analyzing an **FFT**, the target of interest is the *peaks* that appear. These *peaks* occur at locations where the corresponding frequency is dominant in the audio sample. Some audio samples are cleaner and easier to identify *peaks* than others. Consider an audio sample of an instrument playing a single note and compare this to an audio sample of someone speaking. The voice sample is obviously more complex. So the FFT of the voice will have much more going on with many *peaks*. There will be many more non-zero coefficients. The reason these *peaks* are of interest is that they identify which frequencies are most prevalent in the audio sample. Recall the objective of this thesis is to determine a method of converting someone's vocal frequencies to a targets voice. In order to do this, it is desired to find key characteristics of a persons voice and these large *peaks* help identify which frequencies are fundamental to their voice.

The main point to understand about speech is that the sounds generated by a human are filtered by the shape of the vocal tract including tongue, teeth etc. This shape determines what sound comes out. In any automatic speech recognition system to extract features i.e. identify the components of the audio signal that is good for identifying the linguistic content and discarding all the other stuff which carries information like background noise, emotion etc. If we can determine the shape accurately, this should give us an accurate representation of the phoneme being produced. The shape of the vocal tract manifests itself in the envelope of the short time power spectrum, and the job of **MFCCs** is to accurately represent this envelope.

An audio signal is constantly changing, so to simplify things we assume that on short time scales the audio signal doesn't change much. This is why we frame the signal into 20-40ms frames. If the frame is much shorter we don't have enough samples to get a reliable spectral estimate, if it is longer the signal changes too much throughout the frame. The next step is to calculate the power spectrum of each frame. This is motivated by the human cochlea (an organ in the ear) which vibrates at different spots depending on the frequency of the incoming sounds. Depending on the location in the cochlea that vibrates (which wobbles small hairs), different nerves fire informing the brain that certain frequencies are present.

By designing the analog signal with hamming windows as the explanation in equation 2.25, the spectral analysis can be better for FFT processing. We use hamming windows to detect the peak of signal characteristic. The Hamming window has the lowest first side lobe level of all three types of windows. The slow decay means that leakage two or three bins away from a signal's center frequency are lower for the Hamming window.

For better understanding, the FFT uses due to anaylze the potential characteristic of signal waveform when detects the pattern in voices. Fast Fourier Transform algorithm serves as a signal modifier from time domain to frequency domain. The frequency values obtained from this process will be used in the filtering stage to obtain vector coefficients. Fast Fourier Transform (FFT) is a step to change each frame consisting of N samples from time domain into frequency domain. FFT is done to get the frequency of each frame. The output of this FFT process is a spectrum or periodogram.

One of the most popular FFT algorithms is radix-2. As a comparison with the DFT, for a large number of N samples such as N = 512, using DFT calculations requires a calculation of 114 times more than is required by FFT calculations. The larger the number of N samples, the more complex the calculation is if using DFT. The first step to interpret FFT is to calculate the frequency value of each middle sample of the FFT. If the sample time received by FFT is in real form, then only the output $X(m)$ from m=2 to $m = \frac{N}{2}$ as independent.

In this last stage, the value of mel will be converted back into time domain, the result is called Mel Frequency Ceptral Coefficient. This conversion is done using **Discrete Cosine Transform (DCT)**. The average value in dB that can be used to estimate the energy coming from the filter bank. The DCT coefficient is the amplitude value of the resulting spectrum. At this stage the number of ceptrum taken is as much as 13 pieces per frame.

After all process is completed the analog signal will be converted into **spectrogram**. A sound spectrograph (or sonogram) is a visual representation of an acoustic signal. To simplify things with a reasonable amount, the Fast Fourier transform is applied to electronically recorded sounds. Basically, this analysis separates the frequency and amplitudes of the simplex wave components. The results can be visually displayed from this spectrograph, we can see with the amplitude level (represented light to dark, like white = no energy, black = lots of energy), at various frequencies (usually on the vertical axis) by time (horizontal). The DBN structure optimization method proposed in this thesis is shown below:

Step 1 Optimize number of hidden layers

Step 1-1 Let $n\ \varepsilon\ [\underline{n}, \overline{n}]$ as the hidden layer dimensional.

Step 1-2 Item A DBN with n hidden layers is evaluated based on the training and generalization capability. Here, the number of units of each layer is 500.

Step 1-3 If $E_n > E_n^*$, then update the best solutin of $E_n^* = E_n$, $n^* = n$.

Step 1-4 $\overline{n} > n$, let $n = n + 1$ and return to step 1-2. Otherwise let n^* be number of hidden layer.

Step 2 Optimize number of units of each layer (Rough search)

Step 2-1 Let m_i be number of units of i-th layer, $i = 1, 2, \ldots, n^*$ and let t=0

step 2-2 Divide search range of number units of i-th hidden layer, $\left[\underline{x_i}, \overline{x_i}\right]$ into k_i subrange. Let $d_i^{j_i} \equiv \left[\underline{x_i^{j_i}}, \overline{x_i^{j_i}}\right]$ be j_i th subrange, $j = 1, 2, \ldots, k_i$, $\underline{x}_i^1 = \underline{x}_i \overline{x}_i^{-k_i}$, as the dimensional.

step 2-3 Let $x_1^j, x_2^j, \ldots, x_{n^*}^j$ be the center of gravity of the subrange j, and let $x^j \equiv \left(\widehat{x}_1^j\right), \left(\widehat{x}_2^j\right), \ldots, \left(\widehat{x}_{n^*}^j\right) = \left(\left\lfloor \dot{x}_1^j + 0.5 \right\rfloor, \left(\left\lfloor \dot{x}_2^j + 0.5 \right\rfloor, \ldots, \left(\left\lfloor \dot{x}_{n^*}^j + 0.5 \right\rfloor\right)\right.\right.$ be representative point of subrange j.

step 2-4 Evaluate the structural of the neural network to the point of x^j. Then evaluate E^j as the subspace of j.

step 2-5 Choose a subspace with highest evaluation value. Then selected space value as "θ.

step 2-6 If $x_i^{-\Theta} - \underline{x}_i^{\Theta} \leq k_i, \forall iort = T_l$ then go to step 3. Otherwise, generate next search range at representatives $\left[x_i^{j-1}, x_i^{j+1}\right]$ in the neighboring subspace centered "θand go to step 2-2.

Step 3 Optimize number of units of each layer (Detailed search using Tabu search)

step 3-1 Randomly generate an initial solution

step 3-2 Evaluate each neighbor of the current solution

step 3-3 Add solution using Tabu List

step 3-4 Search the neighboring space with the best solution

step 3-5 Find the best solution and terminate it, and goto step 3-2.

4.4 Result and Discussion

All the code used in this work is written in the Python language, using Theano which is a Python library that is used to define, optimize, and evaluate mathematical expressions involving multi-dimensional arrays. Important features of Theano are the possibility to share parameters between different networks and use the Central Processing Units (CPUs) to speed up simulations. Theano has been employed for large-scale computationally intensive scientific investigations since 2007 [].

The experimental design in this thesis derives from simple dataset was traditionally recorded from Google Code Archive, this data set is consisting of 105 audio files in .wav formatted and each files containing utterance of one fruit name spoken by a single speaker. These audio files are divided into seven class categories of fruit names i.e (apple, banana, kiwi, lime, orange, peach and pineapple) one category consists of 15 audio files. The whole dataset is separated into training (91 samples of audio) and testing (14 samples of audio) shows in Table 4.3.1.

In our experiment the signal is sampled in a range 8000 Hz and quantized with 16 bits. The signal is splits up in short frames of 80 samples corresponding to 10 ms of speech. That's range was choosing by relatively limited flexibility of the throat. When process into deep belief network we pick put the features from frequency domain and taking it with fast Fourier transform multiply by a hamming window to reduce the spectral leakage caused by framing of the signal. So after we gets the binary data through to RBM, the structure of RBM has a weakness such as repeating learning could be consuming the time of structure of evaluation and leads to inefficiency in the structure optimization, so by the modularization structure of optimization is performed efficiently by shortening calculation time. By optimized the structure of DBN, we were able to find appropriate structure. conducted the signal it could be read in our system, defined the signal voice with maximum size of 32 KHz then processed the signal into short-time Fourier transform (STFT) by windowing process using hamming-window (change the figure become time vs freq) as shown as figure 4.3.1

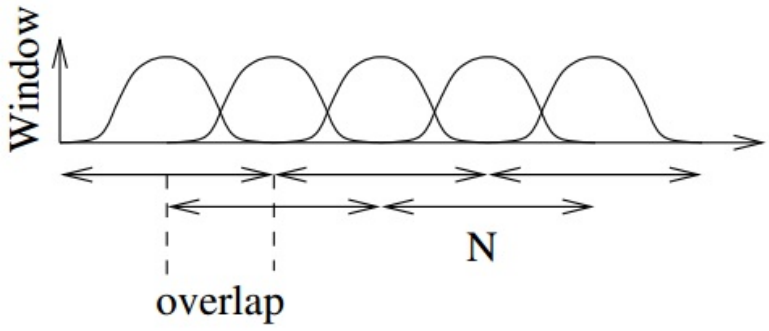

Figure 4.4.1: Windowing process in random signal

to reduce spectral leakage caused by the framing of the signal. After that signal waveform will be extracted into a single data matrix, and a label vector with the correct label for each data file is created.

Once of data has been inputted into a system and converted into a data matrix. The next step extracted the feature selection from the raw data, when it is done we have conducted the signal into Mel Frequency Cepstral Coefficient (MFCC). In this research, we used Short Time Fourier Transform (STFT) to approach the signal peak for processing into signal digital then converted it into the spectrogram.

When analyzing an FFT, the target of interest is the *peaks* that appear. These *peaks* occur at locations where the corresponding frequency is dominant in the audio sample. Some audio samples are cleaner and easier to identify *peaks* than others. Consider an audio sample of an instrument playing a single note and compare this to an audio sample of someone speaking. The voice sample is obviously more complex. So the FFT of the voice will have much more going on with many *peaks*. There will be many more non-zero coefficients. The reason these *peaks* are of interest is that they identify which frequencies are most prevalent in the audio sample. Recall the objective of this thesis is to determine a method of converting someone's vocal frequencies to a targets voice. In order to do this, it is desired to find key characteristics of a persons voice and these large *peaks* help identify which frequencies are fundamental to their voice.

Once we found the peak of the signal as seems like figure 4.3.2, the signal converted to "mfcc" vector and having six features. These mfcc features represented applying Gaussian Mixture Model (GMM). Feature extraction provides a complexity representation of digitalizing from speech. This digitalize perform Figure 4.3.3. indicates that spectral look of voice from sample speaks about ("banana") utterance. This signal has characteristic recorded in 3.5 second and the frequency domain of speech is 16 KHz.

The spectrogram in Figure. 7 has 30×30 dimensional matrix. This dimensional is advantageous for us to process in DBN through to RBM layer as binary data. For the first layer we used binary

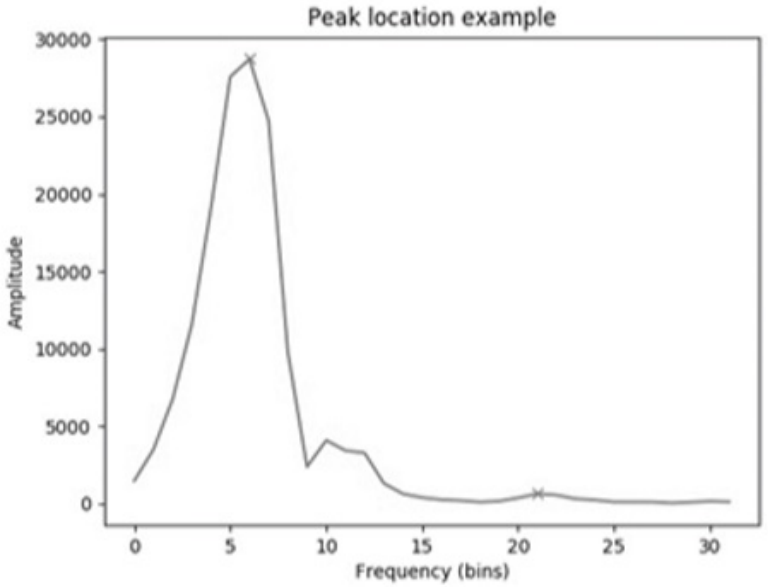

Figure 4.4.2: Spoken word *Peaks* Banana in Time Series

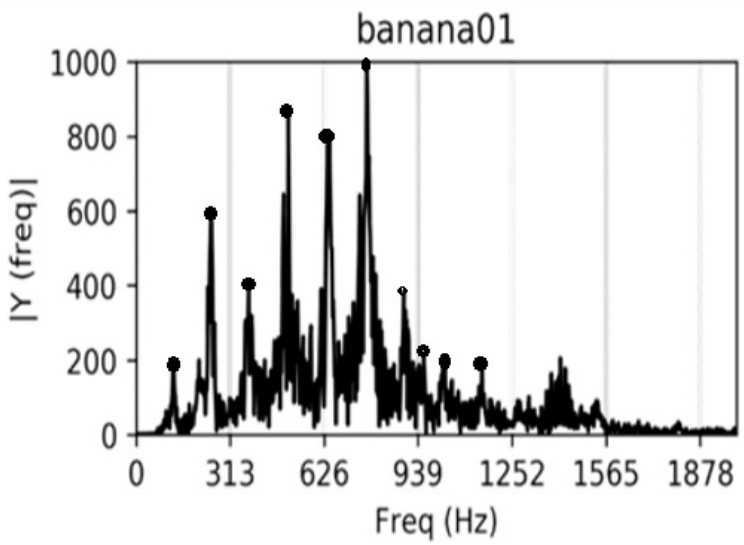

Figure 4.4.3: *Peaks* detection spoken word of banana

data to the RBMs input layer. We normalize the data so it has the zero mean and unit variance. Before processing into RBM the MFCC features attempt to eliminate the information of speech data when it is not having relevancies for recognition purpose. MFCC itself offered the alternative model as individual component to be independent so they are much easier to model using a mixture of diagonal covariance Gaussians.

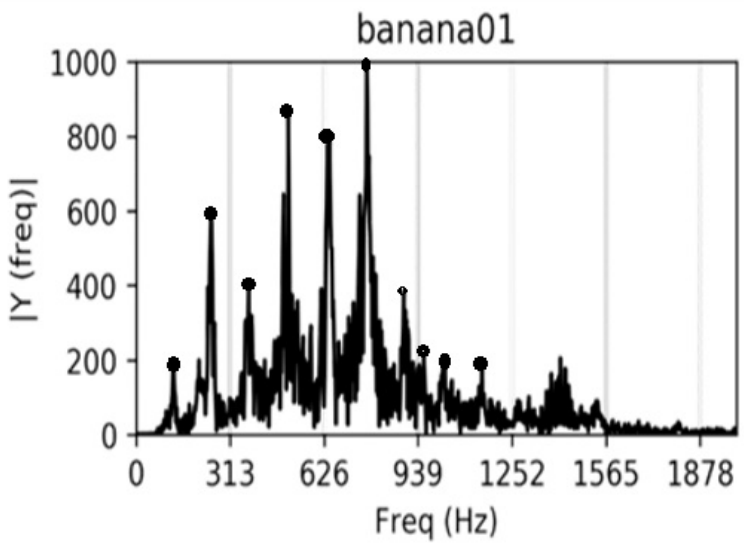

Figure 4.4.4: Spectrogram of Banana

We consider using taboo search in partially of solution space and divided it into multiple subspaces first then the promising regions performed the search procedure. We conducted the experiment with 10 trial of each experiment with different modularization. In each trial, number of units hidden layers are similar to each other. Also we conduct of experiment with the number of hidden layer using different parameter setting, it's like 500-1000 and 2000 in single run of epochs.

Table 4.4.1: Test Accuracy

Test Data	DBN (%)	DBN with Optimization (%)
Speech	99.38	100

In this experiment we believe, using more number of unit in DBN, the feature is not good enough to generate the classifier. Due to the simple speech that we have conducted the performance is quite well enough. We tried to figure it out about the effect of varying size number of unit in initial space search solution. The main trend that is adding more initial number of unit gives the better performance. Although, this research does not try the bigger size of dataset. In other hand, we tried an experimental design in HMM theorem for bench-marking algorithm, we got the accuracy of classification 80 % also with the same data set.

We randomly chose the data for training, validation and testing sets with ratio of 2:1:1 and did the observation of MFCC as well as for feature learned to test our system performances. After that, the voice signal analyzed with processing in windowing and fixed frame rate. Then processed to Fourier transform based on the log filter bank and the energy was disturbed in a Mel-scale, from

Table 4.4.2: Dataset

Dataset	Apple	Banana	Kiwi	Lime	Orange	Peach	Pineapple
Number of Train	14	14	14	14	14	14	14
Total Train	91						
Validation	105						

this process the signal transformed into Discrete Cosine Transform (DCT) derived into MFCC features. Then, data were normalized so that each coefficient had zero mean and unit variance across the training into RBM unit layers.

Dataset in this speech recognition experiment contains 367.5 seconds of speech. This number comes from in each single speaker the datasets consists of 3,5 second length of time then we time this with 105 total of dataset. The acoustic model training set was also used to train only single language for these experiments. Once the data has been inputted and turned into an input matrix, the next step is to extract feature from the raw data, then extracts the raw data into matrix the information of data input describe the sound over both frequency and time.

After that's, we prepare the speech analyses using hamming window with fixed frames rate. In the Mel Frequency Cepstral Coefficient we normally use Cepstral mean normalization over each utterance. These are generated by applying a truncated discrete cosine transformation (DCT) to a log spectral estimate computed by smoothing a Fast Fourier Transforms (FFT) with around 20 frequency bins distributed across the speech spectrum to find peaks in frequency. Typically, we use 13 mfcc coefficients in our experiment.

4.5 Conclusion and Future Work

This study review of the work in DBN and DBN improvement with RBM modularization as better as predicted the simplest speech audio signal files in better way accuracy. The modularization of hidden layer using taboo search is almost the same performance as DBN as without modularization. Even we set the minimum parameter setting of hidden layer size in 500, the performance is optimized well. Otherwise the speed of running the model much be increased when we were running the big data of speech. The larger the number of solution dimensions the execution time will be shortened and made the effective of modularization. Then, the larger number of input dimensions the smaller of calculations amount in solution search. However, when comparing the structure optimized DBN without using DBN structurally optimized using modularization the performance is almost same.

In the future work, we will conduct some kind of the experiments on different voices dataset for benchmarking. Also, we will consider about different processing procedure in signal audio to improve the accuracy. Also we considered about speech in Parkinson diseases would be interested area.

5 Conclusion

Artificial neural networks represent a universal model, which can be leveraged to solve a great variety of tasks. The latest research showed a significant progress of this field in the past few decades. Deep architectures of neural networks started to gain attention after proving success in image and speech recognition in the years after 2000. After the parallel hardware became available for reasonable prices, it fueled the research of efficient optimization of deep neural networks by leveraging parallel architectures.

This thesis presents a models of deep neural networks with taboo search optimization as the evolutionary method. I explained the theory behind each model and described how it is trained to recognize the features in input patterns. The models were also implemented on image and speech dataset and processing using theano on python.

The structure optimization method in classification model for a DBN (Deep Belief Network) which consists of multiple RBMs (Restricted Boltzmann Machine) and a FNN (Feed-forward Neural Network). The features of the proposed method are that it realizes searching both in wide range of solution areas by division of solution space and intensive search by tabu search, introduces modularization of RBMs to improve the calculation efficiency drastically by reducing the calculation amount in solution search.

The modularization of hidden layer using taboo search is almost the same performance as DBN as without modularization. Otherwise the speed of running the model much be increased when we were running the big data of speech. The larger the number of solution dimensions the execution time will be shortened and made the effective of modularization. Then, the larger number of in-

put dimensions the smaller of calculations amount in solution search. However, when comparing the structure optimized DBN without using DBN structurally optimized using modularization the performance is almost same in the speech or classifcation data in image.

References

[1] Ahmed Abbasi, Hsinchun Chen, and Arab Salem. Sentiment analysis in multiple languages: Feature selection for opinion classification in web forums. *ACM Transactions on Information Systems (TOIS)*, 26(3):12, 2008.

[2] Ossama Abdel-Hamid, Abdel-rahman Mohamed, Hui Jiang, and Gerald Penn. Applying convolutional neural networks concepts to hybrid nn-hmm model for speech recognition. In *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, pages 4277–4280. IEEE, 2012.

[3] Ossama Abdel-Hamid, Abdel-rahman Mohamed, Hui Jiang, Li Deng, Gerald Penn, and Dong Yu. Convolutional neural networks for speech recognition. *IEEE/ACM Transactions on audio, speech, and language processing*, 22(10):1533–1545, 2014.

[4] Christos-Nikolaos E Anagnostopoulos. License plate recognition: A brief tutorial. *IEEE Intelligent transportation systems magazine*, 6(1):59–67, 2014.

[5] Itamar Arel, Derek C Rose, Thomas P Karnowski, et al. Deep machine learning-a new frontier in artificial intelligence research. *IEEE computational intelligence magazine*, 5(4):13–18, 2010.

[6] M. A. Atencia, G. Joya, and F. Sandoval. A formal model for definition and simulation of generic neural networks. *Neural Process. Lett.*, 11(2):87–105, April 2000. ISSN 1370-4621. doi: 10.1023/A:1009678528953. URL http://dx.doi.org/10.1023/A:1009678528953.

[7] Metin Mutlu Aydin, Mehmet Sinan Yildirim, Orhan Karpuz, and Kiarash Ghasemlou. Modeling of driver lane choice behavior with artificial neural networks (ann) and linear regression (lr) analysis on deformed roads. *Computer Science & Engineering*, 4(1):47, 2014.

[8] Bhanu Prakash Battula and R Satya Prasad. An overview of recent machine learning strategies in data mining. *Ionosphere*, 351(34):0, 2013.

[9] Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. Greedy layer-wise training of deep networks. In *Proceedings of the 19th International Conference on Neural Information Processing Systems*, NIPS'06, pages 153–160, Cambridge, MA, USA, 2006. MIT Press. URL http://dl.acm.org/citation.cfm?id=2976456.2976476.

[10] Yoshua Bengio et al. Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2(1):1–127, 2009.

[11] Sanjivani S Bhabad and Gajanan K Kharate. An overview of technical progress in speech recognition. *International Journal of advanced research in computer science and software Engineering*, 3(3), 2013.

[12] N Hima Bindu and T Chakravarthi. Booster of an fs algorithm on high dimensional data. 2018.

[13] WW Bledsoe, JS Bomba, I Browning, RJ Evey, RA Kirsch, RL Mattson, M Minsky, U Neisser, and OG Selfridge. Discussion of problems in pattern recognition. In *Papers presented at the December 1-3, 1959, eastern joint IRE-AIEE-ACM computer conference*, pages 233–237. ACM, 1959.

[14] Christopher JC Burges. A tutorial on support vector machines for pattern recognition. *Data mining and knowledge discovery*, 2(2):121–167, 1998.

[15] Jaime G Carbonell, Ryszard S Michalski, and Tom M Mitchell. Machine learning: a historical and methodological analysis. *AI Magazine*, 4(3):69, 1983.

[16] William Chan, Navdeep Jaitly, Quoc Le, and Oriol Vinyals. Listen, attend and spell: A neural network for large vocabulary conversational speech recognition. In *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*, pages 4960–4964. IEEE, 2016.

[17] Yanping Chen, Eamonn Keogh, Bing Hu, Nurjahan Begum, Anthony Bagnall, Abdullah Mueen, and Gustavo Batista. The ucr time series classification archive, 2015.

[18] Mohamed Amine Chéragui. Theoretical overview of machine translation. *Proceedings ICWIT*, page 160, 2012.

[19] George E Dahl, Dong Yu, Li Deng, and Alex Acero. Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. *IEEE Transactions on audio, speech, and language processing*, 20(1):30–42, 2012.

[20] Bishnu Prasad Das and Ranjan Parekh. Recognition of isolated words using features based on lpc, mfcc, zcr and ste, with neural network classifiers. *International Journal of Modern Engineering Research (IJMER)*, 2(3):854–858, 2012.

[21] Namrata Dave. Feature extraction methods lpc, plp and mfcc in speech recognition. *International journal for advance research in engineering and technology*, 1(6):1–4, 2013.

[22] Peter Dayan, Geoffrey E Hinton, Radford M Neal, and Richard S Zemel. The helmholtz machine. *Neural computation*, 7(5):889–904, 1995.

[23] Felice Dell'Orletta. Ensemble system for part-of-speech tagging. *Proceedings of EVALITA*, 9:1–8, 2009.

[24] Li Deng. Deep learning: from speech recognition to language and multimodal processing. *APSIPA Transactions on Signal and Information Processing*, 5:e1, 2016. doi: 10.1017/ATSIP.2015.22.

[25] Li Deng, Ossama Abdel-Hamid, and Dong Yu. A deep convolutional neural network using heterogeneous pooling for trading acoustic invariance with phonetic confusion. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 6669–6673. IEEE, 2013.

[26] Li Deng, Geoffrey Hinton, and Brian Kingsbury. New types of deep neural network learning for speech recognition and related applications: An overview. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 8599–8603. IEEE, 2013.

[27] Li Deng, Dong Yu, et al. Deep learning: methods and applications. *Foundations and Trends® in Signal Processing*, 7(3–4):197–387, 2014.

[28] Tianchuan Du and V Shanker. Deep learning for natural language processing. *Eecis. Udel. Edu*, pages 1–7, 2009.

[29] Dagao Duan, Qian Mo, Yueliang Wan, and Zhongming Han. A detail preserving filter for impulse noise removal. In *Computer Application and System Modeling (ICCASM), 2010 International Conference on*, volume 2, pages V2–265. IEEE, 2010.

[30] Stéphane Dupont and Leila Cheboub. Fast speaker adaptation of artificial neural networks for automatic speech recognition. In *Acoustics, Speech, and Signal Processing, 2000. ICASSP'00. Proceedings. 2000 IEEE International Conference on*, volume 3, pages 1795–1798. IEEE, 2000.

[31] B El Kessab, C Daoui, B Bouikhalene, M Fakir, and K Moro. Extraction method of handwritten digit recognition tested on the mnist database. *International Journal of Advanced Science & Technology*, 50:99–110, 2013.

[32] OS Eluyode and Dipo Theophilus Akomolafe. Comparative study of biological and artificial neural networks. *European Journal of Applied Engineering and Scientific Research*, 2(1):36–46, 2013.

[33] Randall J Erb. Introduction to backpropagation neural network computation. *Pharmaceutical research*, 10(2):165–170, 1993.

[34] Mehdi Fatemi and Simon Haykin. Cognitive control: Theory and application. *IEEE Access*, 2:698–710, 2014.

[35] Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *Comput. Vis. Image Underst.*, 106(1):59–70, April 2007. ISSN 1077-3142. doi: 10.1016/j.cviu.2005.09.012. URL http://dx.doi.org/10.1016/j.cviu.2005.09.012.

[36] Santosh K Gaikwad, Bharti W Gawali, and Pravin Yannawar. A review on speech recognition technique. *International Journal of Computer Applications*, 10(3):16–24, 2010.

[37] Mario Garrido. A new representation of fft algorithms using triangular matrices. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 63(10):1737–1745, 2016.

[38] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576*, 2015.

[39] Ladan Golipour and Douglas O'Shaughnessy. Context-independent phoneme recognition using a k-nearest neighbour classification approach. In *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*, pages 1341–1344. IEEE, 2009.

[40] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *Acoustics, speech and signal processing (icassp), 2013 ieee international conference on*, pages 6645–6649. IEEE, 2013.

[41] Jeff Hawkins, Subutai Ahmad, and Donna Dubinsky. Hierarchical temporal memory including htm cortical learning algorithms. *Techical report, Numenta, Inc, Palto Alto http://www. numenta. com/htmoverview/education/HTM_CorticalLearningAlgorithms. pdf*, 2010.

[42] Tomohiro Hayashida, Ichiro Nishizaki, and Tsubasa Matsumoto. Structural optimization of neural network for data prediction using dimensional compression and tabu search. In *Computational Intelligence & Applications (IWCIA), 2013 IEEE Sixth International Workshop on*, pages 85–88. IEEE, 2013.

[43] Georg Heigold, Erik McDermott, Vincent Vanhoucke, Andrew Senior, and Michiel Bacchiani. Asynchronous stochastic optimization for sequence training of deep neural networks. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pages 5587–5591. IEEE, 2014.

[44] Georgef Hepner, Thomas Logan, Niles Ritter, and Nevin Bryant. Artificial neural network classification using a minimal training set- comparison to conventional supervised classification. *Photogrammetric Engineering and Remote Sensing*, 56(4):469–473, 1990.

[45] Geoffrey E Hinton. Preface to the special issue on connectionist symbol processing. *Artificial Intelligence*, 46(1-2):1–4, 1990.

[46] Geoffrey E Hinton and James L McClelland. Learning representations by recirculation. In *Neural information processing systems*, pages 358–366, 1988.

[47] Haoyuan Hong, Chong Xu, Inge Revhaug, and Dieu Tien Bui. Spatial prediction of landslide hazard at the yihuang area (china): a comparative study on the predictive ability of backpropagation multi-layer perceptron neural networks and radial basic function neural networks. In *Cartography-Maps Connecting the World*, pages 175–188. Springer, 2015.

[48] Eva Hörster and Rainer Lienhart. Deep networks for image retrieval on large-scale databases. In *Proceedings of the 16th ACM international conference on Multimedia*, pages 643–646. ACM, 2008.

[49] Xuedong Huang, Alex Acero, Hsiao-Wuen Hon, and Raj Reddy. *Spoken language processing: A guide to theory, algorithm, and system development*, volume 1. Prentice hall PTR Upper Saddle River, 2001.

[50] Y Ishikawa, T Hayashida, I Nishizaki, and S Sekizaki. Improvement of structure optimization method of deep belief network. In *2017 IEEE SMC Hiroshima Chapter Young Researchers Workshop*, pages 56–60, 2017.

[51] Dan Jurafsky and James H Martin. *Speech and language processing*, volume 3. Pearson London, 2014.

[52] Aman Mohammad Kalteh, Peder Hjorth, and Ronny Berndtsson. Review of the self-organizing map (som) approach in water resources: Analysis, modelling and application. *Environmental Modelling & Software*, 23(7):835–845, 2008.

[53] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1725–1732, 2014.

[54] Stefan Knerr, Léon Personnaz, and Gérard Dreyfus. Single-layer learning revisited: a stepwise procedure for building and training a neural network. In *Neurocomputing*, pages 41–50. Springer, 1990.

[55] Megha Kumar, Kanika Bhutani, Swati Aggarwal, et al. Hybrid model for medical diagnosis using neutrosophic cognitive maps with genetic algorithms. In *Fuzzy Systems (FUZZ-IEEE), 2015 IEEE International Conference on*, pages 1–7. IEEE, 2015.

[56] Kevin J Lang, Alex H Waibel, and Geoffrey E Hinton. A time-delay neural network architecture for isolated word recognition. *Neural networks*, 3(1):23–43, 1990.

[57] Quoc V Le, Jiquan Ngiam, Adam Coates, Abhik Lahiri, Bobby Prochnow, and Andrew Y Ng. On optimization methods for deep learning. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, pages 265–272. Omnipress, 2011.

[58] Nicolas Le Roux and Yoshua Bengio. Representational power of restricted boltzmann machines and deep belief networks. *Neural computation*, 20(6):1631–1649, 2008.

[59] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, 2015.

[60] Honglak Lee, Roger Grosse, Rajesh Ranganath, and Andrew Y Ng. Unsupervised learning of hierarchical representations with convolutional deep belief networks. *Communications of the ACM*, 54(10):95–103, 2011.

[61] Feng Li. The information content of forward-looking statements in corporate filings—a naïve bayesian machine learning approach. *Journal of Accounting Research*, 48(5):1049–1102, 2010.

[62] Le-Wei Li, Ya-Nan Li, Tat Soon Yeo, Juan R Mosig, and Olivier JF Martin. A broadband and high-gain metamaterial microstrip antenna. *Applied Physics Letters*, 96(16):164101, 2010.

[63] Shu-Chiang Lin, Murman Dwi Prasetio, Satria Fadil Persada, and Reny Nadlifatin. A naïve bayes based machine learning approach and application tools comparison based on telephone conversations. In *Proceedings of the Institute of Industrial Engineers Asian Conference 2013*, pages 1017–1023. Springer, 2013.

[64] Xia Lin, Dagobert Soergel, and Gary Marchionini. A self-organizing semantic map for information retrieval. In *Proceedings of the 14th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 262–269. ACM, 1991.

[65] Zhen-Hua Ling, Li Deng, and Dong Yu. Modeling spectral envelopes using restricted boltzmann machines for statistical parametric speech synthesis. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 7825–7829. IEEE, 2013.

[66] Guangkai Ma, Yaozong Gao, Guorong Wu, Ligang Wu, and Dinggang Shen. Nonlocal atlas-guided multi-channel forest learning for human brain labeling. *Medical physics*, 43(2):1003–1019, 2016.

[67] Maria das Graças Bruno Marietto, Rafael Varago de Aguiar, Gislene de Oliveira Barbosa, Wagner Tanaka Botelho, Edson Pimentel, Robson dos Santos França, and Vera Lúcia da Silva. Artificial intelligence markup language: a brief tutorial. *arXiv preprint arXiv:1307.3091*, 2013.

[68] James Martens and Ilya Sutskever. Learning recurrent neural networks with hessian-free optimization. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 1033–1040. Citeseer, 2011.

[69] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

[70] Grégoire Mesnil, Xiaodong He, Li Deng, and Yoshua Bengio. Investigation of recurrent-neural-network architectures and learning methods for spoken language understanding. In *Interspeech*, pages 3771–3775, 2013.

[71] Yajie Miao, Mohammad Gowayyed, and Florian Metze. Eesen: End-to-end speech recognition using deep rnn models and wfst-based decoding. In *Automatic Speech Recognition and Understanding (ASRU), 2015 IEEE Workshop on*, pages 167–174. IEEE, 2015.

[72] Tomáš Mikolov, Anoop Deoras, Daniel Povey, Lukáš Burget, and Jan Černockỳ. Strategies for training large scale neural network language models. In *Automatic Speech Recognition and Understanding (ASRU), 2011 IEEE Workshop on*, pages 196–201. IEEE, 2011.

[73] Abdel-rahman Mohamed, George E Dahl, Geoffrey Hinton, et al. Acoustic modeling using deep belief networks. *IEEE Trans. Audio, Speech & Language Processing*, 20(1):14–22, 2012.

[74] Nelson Morgan. Deep and wide: Multiple layers in automatic speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(1):7–13, 2012.

[75] K-R Muller, Sebastian Mika, Gunnar Ratsch, Koji Tsuda, and Bernhard Scholkopf. An introduction to kernel-based learning algorithms. *IEEE transactions on neural networks*, 12 (2):181–201, 2001.

[76] Ö Özdamar and T Kalayci. Detection of spikes with artificial neural networks using raw eeg. *Computers and Biomedical Research*, 31(2):122–142, 1998.

[77] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28*, ICML'13, pages III–1310–III–1318. JMLR.org, 2013. URL http://dl.acm.org/citation.cfm?id=3042817.3043083.

[78] Pavitra Patel, Anand Chaudhari, Ruchita Kale, and M Pund. Emotion recognition from speech with gaussian mixture models and via boosted gmm. *International Journal of Research In Science & Engineering*, 3, 2017.

[79] Joel Pinto, Sivaram Garimella, Mathew Magimai-Doss, Hynek Hermansky, and Hervé Bourlard. Analysis of mlp-based hierarchical phoneme posterior probability estimator. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(2):225–241, 2011.

[80] David C Plaut and Geoffrey E Hinton. Learning sets of filters using back-propagation. *Computer Speech & Language*, 2(1):35–61, 1987.

[81] Murman Prasetio, Tomohiro Hayashida, Ichiro Nishizaki, and Shinya Sekizaki. Enhancing single speaker recognition using deep belief network. *Transactions on Machine Learning and Artificial Intelligence*, 6(4):01, 2018. ISSN 2054-7309.

[82] Lawrence R Rabiner and Biing-Hwang Juang. *Fundamentals of speech recognition*, volume 14. PTR Prentice Hall Englewood Cliffs, 1993.

[83] Anatol Rapoport. Mathematical biophysics, cybernetics and significs. *Synthese*, 8(1):182–193, 1949.

[84] Anthony J Robinson. An application of recurrent nets to phone probability estimation. *IEEE transactions on Neural Networks*, 5(2):298–305, 1994.

[85] Raúl Rojas. *Neural networks: a systematic introduction*. Springer Science & Business Media, 2013.

[86] Lior Rokach and Oded Z Maimon. *Data mining with decision trees: theory and applications*, volume 69. World scientific, 2008.

[87] Haşim Sak, Andrew Senior, Kanishka Rao, and Françoise Beaufays. Fast and accurate recurrent neural network acoustic models for speech recognition. *arXiv preprint arXiv:1507.06947*, 2015.

[88] Ruslan Salakhutdinov. Learning deep generative models. *Annual Review of Statistics and Its Application*, 2:361–385, 2015.

[89] Claude Sammut and Geoffrey I Webb. *Encyclopedia of machine learning*. Springer Science & Business Media, 2011.

[90] Mohammad Shahbakhi, Danial Taheri Far, Ehsan Tahami, et al. Speech analysis for diagnosis of parkinson's disease using genetic algorithm and support vector machine. *Journal of Biomedical Science and Engineering*, 7(4):147–156, 2014.

[91] Matt Shannon, Heiga Zen, and William Byrne. Autoregressive models for statistical parametric speech synthesis. *IEEE transactions on audio, speech, and language processing*, 21(3): 587–597, 2013.

[92] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[93] Sabato Marco Siniscalchi, Torbjørn Svendsen, and Chin-Hui Lee. A bottom-up modular search approach to large vocabulary continuous speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 21(4):786–797, 2013.

[94] Richard Socher, Christopher D Manning, and Andrew Y Ng. Learning continuous phrase representations and syntactic parsing with recursive neural networks. In *Proceedings of the NIPS-2010 Deep Learning and Unsupervised Feature Learning Workshop*, volume 2010, pages 1–9, 2010.

[95] Dennis Spohr, Laura Hollink, and Philipp Cimiano. A machine learning approach to multilingual and cross-lingual ontology matching. In *International Semantic Web Conference*, pages 665–680. Springer, 2011.

[96] Jiping Sun and Li Deng. An overlapping-feature-based phonological model incorporating linguistic constraints: Applications to speech recognition. *The Journal of the Acoustical Society of America*, 111(2):1086–1101, 2002.

[97] Joshua Susskind, Volodymyr Mnih, Geoffrey Hinton, et al. On deep generative models with applications to recognition. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 2857–2864. IEEE, 2011.

[98] Ilya Sutskever and Geoffrey E Hinton. Deep, narrow sigmoid belief networks are universal approximators. *Neural computation*, 20(11):2629–2636, 2008.

[99] Pawel Swietojanski, Arnab Ghoshal, and Steve Renals. Convolutional neural networks for distant speech recognition. *IEEE Signal Processing Letters*, 21(9):1120–1124, 2014.

[100] Kozo Takayama, Mikito Fujikawa, and Tsuneji Nagai. Artificial neural network as a novel method to optimize pharmaceutical formulations. *Pharmaceutical research*, 16(1):1–6, 1999.

[101] Jiexiong Tang, Chenwei Deng, and Guang-Bin Huang. Extreme learning machine for multilayer perceptron. *IEEE transactions on neural networks and learning systems*, 27(4):809–821, 2016.

[102] Ryan W Thomas, Daniel H Friend, Luiz A Dasilva, and Allen B Mackenzie. Cognitive networks: adaptation and learning to achieve end-to-end performance objectives. *IEEE Communications Magazine*, 44(12):51–57, 2006.

[103] Joseph P Turian, Luke Shea, and I Dan Melamed. Evaluation of machine translation and its evaluation. Technical report, NEW YORK UNIV NY, 2006.

[104] Leonard Uhr. Intelligence in computers: the psychology of perception in people and in machines. *Behavioral Science*, 5(2):177, 1960.

[105] HJL Van Can, HAB Te Braake, C Hellinga, AJ Krijgsman, HB Verbruggen, K Ch AM Luyben, and JJ Heijnen. Design and real time testing of a neural model predictive controller for a nonlinear system. *Chemical Engineering Science*, 50(15):2419–2430, 1995.

[106] M Gr Voskoglou. Stochastic and fuzzy models in mathematics education, artificial intelligence and management. *Lambert Academic Publishing, Saarbrucken, Germany*, 2011.

[107] Mark P Wachowiak, Adel Said Elmaghraby, Renata Smolikova, and Jacek M Zurada. Classification and estimation of ultrasound speckle noise with neural networks. In *Bio-Informatics and Biomedical Engineering, 2000. Proceedings. IEEE International Symposium on*, pages 245–252. IEEE, 2000.

[108] Shenhao Wang and Jinhua Zhao. Framing discrete choice model as deep neural network with utility interpretation. *arXiv preprint arXiv:1810.10465*, 2018.

[109] Bernard Widrow and Michael A Lehr. 30 years of adaptive neural networks: perceptron, madaline, and backpropagation. *Proceedings of the IEEE*, 78(9):1415–1442, 1990.

[110] Michael Wohlmayr, Michael Stark, and Franz Pernkopf. A probabilistic interaction model for multipitch tracking with factorial hidden markov models. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(4):799–810, 2011.

[111] Bishan Yang, Jian-Tao Sun, Tengjiao Wang, and Zheng Chen. Effective multi-label active learning for text classification. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 917–926. ACM, 2009.

[112] D Yang, S Furui, et al. Combining a two-step crf model and a joint source channel model for machine transliteration. In *Proceedings of Association for Computational Linguistics (ACL)*, pages 275–280, 2010.

[113] Dong Yu and Li Deng. Deep-structured hidden conditional random fields for phonetic recognition. In *Eleventh Annual Conference of the International Speech Communication Association*, 2010.

[114] Dong Yu, Li Deng, Yifan Gong, and Alex Acero. A novel framework and training algorithm for variable-parameter hidden markov models. *IEEE transactions on audio, speech, and language processing*, 17(7):1348–1360, 2009.

[115] Xiao-Lei Zhang and Ji Wu. Deep belief networks based voice activity detection. *IEEE Transactions on Audio, Speech, and Language Processing*, 21(4):697–710, 2013.

[116] Jacek M Zurada, Aleksander Malinowski, and Ian Cloete. Sensitivity analysis for minimization of input data dimension for feedforward neural network. In *Circuits and Systems, 1994. ISCAS'94., 1994 IEEE International Symposium on*, volume 6, pages 447–450. IEEE, 1994.

Index