2. Related Works:

2.1 Frame Selection

2.1.1 Bayesian explanation for frame selection in speech recognition

2.1.1.1. Equal Prior distribution

2.1.1.2. Fixed number of selected frames

2.1.1.3 Fixed selected frames

2.1.1.4 Selected frames dependent on hypothesis

2.1.2 Frame Selection in phoneme classification

2.1.2.1 Position Based Fixed frame selection

2.1.2.2 ML based fixed frame selection

2.1.2.3 Combination Based Fixed frame Selection

2.2 Approaches for Feature Extraction:

2.2.1 Need for Feature Extraction

2.2.2 Modelling Techniques:

2.2.3 Robust Techniques for Feature Extraction:

2.2.3.1 MFCC

2.2.3.2 LPC

2.4 De…

2.3 Deep Neural Networks for Speech Recognition:

2.2.1 A New Machine Learning Paradigm

2.2.2 Supervised and Unsupervised Learning

2.2.3 From Shallow to Deep Architectures

2.2.4 Case of Success

2.2.4.1 Deep Convolutional Networks

` 2.2.4.2 Deep Recurrent Networks

**2. Related works:**

The following section covers some pre-existing State of the Art methods involving Deep Neural Network for solving problems in Speech Recognition Model Construction. Speech Recognition is a Broad area in the field of Computer science, particularly in the domain of Artificial Intelligence. Many researchers have worked in different speech recognition techniques for almost four decades. The earliest attempts were made in the 50’s. In 1952, at Bell Laboratories, researchers built a system for isolated digit recognition for a single speaker. In 1956, at RCA Laboratories, researchers developed a system designed to recognise 10 distinct syllables of a single speaker. In 1959, at University College in England, Students demonstrated a system to recognise four vowels and nine consonants. The same year, at MIT’s Lincoln Laboratories professors built a system to recognise 10 vowels in a speaker independent manner. All of these systems used spectral information to extract voice features. After the following years, some significant researches emerged in some 80’s. In the 80’s, the topic was connected word recognition. Speech recognition research was characterized by a shift in technology from template-based approaches to statistical modeling methods, especially Hidden Markov Models (HMM). Thanks to the widespread publication of the theory and methods of this technique in the mid 80’s, the approach of employing HMMs has now become widely applied in virtually every speech recognition laboratory of the world. Another idea that appeared in the arena was the use of neural nets in speech recognition problems. The impetus given by DARPA to solve the large vocabulary, continuous speech recognition problem for defense applications was decisive in terms of increasing the research in the area. Today’s research focuses on a broader definition of speech recognition. It is not only concerned with recognizing the word content but also prosody and personal signature [4]. It also recognizes that other languages are used together with speech, taking a multimodal approach that also tries to extract information from gestures and facial expressions.

Despite all of the advances in the speech recognition area, the problem is far from being completely solved. A number of excellent commercial products, which are getting closer and closer to the final goal, are currently sold in the commercial market. Products that recognize the voice of a person within the scope of a credit card phone system, command recognizers that permit voice control of different types of machines, “electronic typewriters” that can recognize continuous voice and manage several tens of thousands word vocabularies, and so on. However, although these applications may seem impressive, they are still computationally intensive, and in order to make their usage widespread more efficient algorithms must be developed. Summing up, there is still room for a lot of improvement and of course, research.

An overview of some of the popular methods for speech recognition is presented in this section following the schematic diagram that outlines the constituent blocks as in

Figure 2-1. The functionality of the individual blocks is also described in order to precisely state the contributions that stem from the work outlined in this thesis.

Speech Recogniser

Feature Extractor

Recogniser

Text

Speech

Figure 2-1 Basic Building blocks of a Speech Recogniser.

**2.1 Frame selection:**

Most speech recognition systems use a fixed frame rate, typically 100Hz, to decompose speech into a series of frames mainly for its convenience. In such a scheme all speech frames are assigned the same importance from a pattern classification viewpoint. But some limitations appear with this arbitrary fixed frame rate. For instance, a noisy frame may dominate the recognition process; the same importance assigned to each extracted frame is inconsistent with human perception [18]; pitch asynchronous representation [19] [20] caused by the fixed frame rate leads to pitch mismatch due to the presence of pitch-related harmonics in the power spectrum; the well-known implicit weakness of HMM in duration modelling aggravates this concern. Furthermore, in the case of continuous speech recognition, observations at the beginning and the end of a phoneme segment are highly influenced by contextual information. Hence, the distributions of these observations that are dominated by co-articulation are broad and their likelihoods might not be informative. Indeed, [21] discovered that the frames at boundaries may carry more speaker-related information than speech-related information; hence using frames for speech recognition at the boundaries may involve a risk. Moreover, frames at the boundaries usually carry information from both sides, which damages the HMM assumption that all feature vectors on the same state are identically distributed. On the contrary, observations in the steady state 1, although more reliable, tend to be similar and redundant in the decision process. While the recognition decision is made by comparing the sums of the likelihoods of all hypotheses, keeping the most discriminative frames and throwing away redundant ones would not greatly affect the maximization operation. Because of those limitations, researchers are looking for a better frame representation for an utterance and a way to reconstruct the spectra more efficiently. These efforts include variable frame rate [22], segment normalization [23], speaking rate estimation [24], etc.

Among these efforts, one of the significant studies was done by Hillenbrand and his colleagues [25]. The purpose of the study was to replicate and extend the classic study of vowel acoustics by Peterson and Barney [26], including formant contours for F1-F4 and vowel durations in context- fixed (in a /hVd/ structure) vowel discrimination. The study showed that the confusions among adjacent vowel categories in the F1-F2 space are much greater than Peterson-Barney’s study suggested, and many of the vowels are in different locations. The authors suggested that “...the vowels of American English are more appropriately viewed not as points in phonetic space but rather as trajectories through phonetic space” [25]. In the experiments, they showed that using F1-F3 as features and a quadratic discrimination analysis technique [27], the 10-vowel classification accuracy (after omitting two centre diphthongs /e/ and /o/) was 81.0% for only using one frame at steady state, 91.6% for two frames taken at 20% and 80% of vowel duration and 91.8% for three frames taken at 20%, 50% and 80%. The authors claimed that “this would seem to suggest that only a very coarse representation of the spectral change pattern is needed for classification” [25]. We observe that this simple frame selection method could discard those context-influenced frames at the phoneme boundaries and simultaneously reduce the redundancy possibly occurring in the middle part. On the other hand, this method could be a way to overcome the shortcomings of the fixed frame rate. However, this study was limited to a context-fixed experiment and has not been extensively investigated in more complex phonetic environments than the /hVd/ utterances examined. Furthermore, the concept is not directly applicable to continuous speech recognition as it needs to know phoneme boundaries.

The frame selection technique proposed in this chapter is an extension of the simple frame selection done by Hillenbrand, and an alternative solution for overcoming the problems resulting from the fixed frame rate. Frame selection is a technique to select a subset of speech frames as a representative for the whole speech signal to distinguish the characteristics of speech units. The original set of speech frames is usually generated using a high fixed frame rate, on which a frame selection technique is employed to pick reliable and informative frames. In this chapter, we develop a series of simple-to-complex schemes to demonstrate the advantage of frame selection and to investigate the scope within which frame selection is applicable

**2.1.1 Bayesian explanation for frame selection in speech recognition:**

Given acoustic data and word acoustic HMMs Φw. , the task of speech recognition is to look for a word or state sequence which maximizes the posterior probability. According to the Bayesian formula, this turns to look for the maximum of [30].

The application of frame selection is to select a subset of frames from the full set to replace the full frame set. I approximate the likelihood as expectation of. The item indicates the likelihood of selection subset given . Further, instead of taking the expectation for all possible. I simplify the expectation operation to the maximum using maximum function.

The first item on the right side represents the log-likelihood of to the acoustic model of W; it can be estimated by a standard likelihood calculation. The second item is the language model related factor; in a phoneme classification task, for example, this item is equal to all possible sequences. The third item stands for the probability that is generated. The last item represents the probability that the subset is selected from the full set . This item depends on how the subset is generated. The following image illustrate the Bayesian surface interference of sound signal for frame length of 0, 20, 40 and 60.

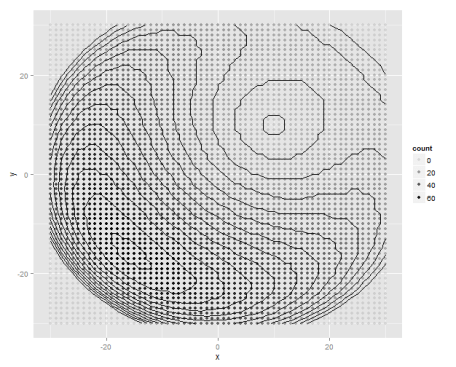


Figure 2-2. Bayesian model for multiple frame

**2.1.2 Equal prior distribution:**

In some applications, such as phoneme classification, the prior distribution of each phoneme is presumed equal. Thus the second item log can be ignored.

**2.1.3 Fixed number of selected frame:**

For some frame selection methods, the number of selected frames is determined a priori, independent of the length of the phoneme segment. In this case is equal for all possible. For instance, if we decide to select 1 frame from a phoneme segment with the length T, then for all. Therefore the equation in 2.1.2 is shortened as.

2.1.4 Fixed Selected Frames

If the selected frames are fixed before recognition, the selected frames are independent of a word sequence, then the items and are identical for all possible hypotheses [30]. As a result, the recognized word sequence is searched by:

**2.1.5 Selected frames dependent on hypothesis:**

In some cases, the selected frame set is dependent on a word sentence hypothesis. In this situation, has to be denoted as W. To abbreviate our notations, we use W to denote W as the frame set selected by hypothesis. Then the recognized word sequence is found as follows:

The third item on the right side is the log probability that is generated. Note that unlike Equation 1.6, cannot be omitted as it is not equal for all hypotheses. In our implementation, a sink model constructed by HMM models for all hypotheses is used to estimate the observation probability of W , as an HMM state is modelled by weighted Gaussian mixtures:

**2.2 Frame Selection in phoneme classification**

Fixed frame selection is a class of methods in which a small and fixed number of frames is selected per phoneme. The number of selected frames is independent of the number of frames a phoneme segment holds [30]. Unlike the method used by Hillenbrand [25], which took frames only at 20%, 50% or 80%, or their combination, the proposed fixed frame selection is to compress the number of selected frames to an extreme case and to investigate at which position(s) frames are the most representative.

In the first study, I attempt to select the most discriminative frame regardless of the duration of a test segment; then we increase the number of desired frames to two, and see how the performance of phoneme classification is enhanced and what their corresponding positions are. Finally, we select one frame to represent each state in a multi-state HMM. In this case the number of selected frames for a test phoneme segment depends on the topology of the HMM. By incrementally increasing the number of selected frames, we are capable of studying factors that influence the accuracy of frame selection and investigate the characteristics of the selected frames. In the selection procedure, two frame properties will be used. One is the likelihood of the frame against all possible classes; the other is the position the frame holds in the phoneme segment. We will show that although both properties are important for accurately spotting the representative frames, the positions of the selected frames are more crucial; but combining these two properties is better than using either property alone

**2.2.1 Position Based fixed frame selection:**

A first family of approaches for selecting frames, called Position based fixed frame selection, is to simply pick frames at fixed relative positions of the duration in a phoneme segment without taking their likelihoods into account [28], For example, we could arbitrarily select a frame at 50% of the duration as the representative one if the desired number of selected frames is one. We may also pick frames at 20%, 50% and 80% of the duration for a three-state HMM, each frame representing one state. Position based fixed frame selection is an extension of the method used by Hillenbrand. However, in Hillenbrand’s study [25], the context was fixed as all vowels were embedded into a /hVd/ structure; the fixed relative positions he used (20%, 50% and 80%) are probably suitable neither for all phonemes, nor in continuous circumstances. Therefore we need to look for more discriminative positions in our case. We will look for frames positioned at percentages of the segment duration. In practice each segment is normalized to 11 frames, yielding normalized positions 0% (first frame), 10%, 20% ... 100% (last frame). For segments with more than 11 frame originally, this normalization is a down-sampling process; for segments shorter than 11 frames, some frames are duplicated. Practically, because the frame shift we use for frame selection is 2msec, only 3.95% of phoneme segments are shorter than 22msec, thus need to duplicate some frames. This normalization is very similar to the approach adopted in [29], where the authors proposed segment normalization to replace the fixed frame rate using down-sampling for longer segments but using the Missing Data Technique (MDT) to reconstruct a phoneme segment shorter than a desired duration. The reason that we use the duplication strategy instead of the MDT is that in our study we are more concerned about the speech event happening at a certain position, while in [29] the authors focused on accurate speech recovery. Assume the fixed relative positions E are pre-defined, is calculated as where indicate the is rounded to the nearest integer.

Note that if no knowledge of prior distribution is known, this approach meets the requirement of the number of selected frames is fixed and the selected frames are independent of the phoneme hypotheses, thus, a speech segment is classified to a phoneme \* which obtains the maximum likelihood:

Where c is a possible phoneme class

**FPOS 1: One frame per phoneme**

In FPOS 1, we look for only one frame at a fixed position to represent the characteristics of the phoneme segment. It is commonly believed that a frame in the steady state is the least influenced by its context, thus is more representative than frames close to boundaries. If the frame at is picked, the log likelihood of the whole segment for a hypothesis class is then defined as, where the maximum operation is used to look for the state which gives the largest likelihood among all possible states for a phoneme class. If we denote the likelihood of the segment to a specific class as , then a phoneme segment is identified to the class whose likelihood is maximal among all possible classes:

**FPOS 2: Two frames per phoneme**

With FPOS 2, two frames are selected at two pre-defined positions. Selecting two frames does not mean to simply add one candidate frame to the frame selected by the FPOS 1 method: although the frame selected from FPOS 1 is the most discriminative one, it does not guarantee that the best performance could be reached by incorporating another distinguished but probably correlated frame. FPOS 2 is a way to investigate at which positions frames could be combined best.

**FPOS N: One frame per HMM state**

A well-known assumption of the HMM is that feature vectors in the same state follow the same distribution. Therefore in frame selection, it is natural to investigate that to what extent the combination of one frame picked from one state could represent the whole segment, as theoretically in the traditional HMM the likelihood of the whole test segment is merely the linear sum of the likelihoods of observations in different states (if the much less influential factor – state transition probability is not considered). Before the investigation, we need to know the best relative position of a state from the training set. To do this, all combinations of three fixed relative positions are tested in the training set and the frame combination which obtains the best performance is retrieved as the best positions. Assuming that the best positions, for all states are found in the training, the score for a phoneme model is nothing more than the sum of the likelihoods of the frames at t :

**2.3.3. ML based fixed frame selection**

Another approach to select frames is based on the likelihoods of frames. Likelihood describes how similar a test sample to a model is, or how close in the feature space a test sample to a model is. The motivation for using likelihood scores as a criterion to select frames is coming from the decision process of an ML framework in speech recognition: a decision is made by comparing the sum of likelihoods of all frames in a test segment to all potential acoustic class models; the one which is the most likely to generate the observation is identified as the test class. The distances, or likelihoods of frames in one speech segment to a hypothesis can be different: some of them may be closer than others; we conjecture that the frames which are the closest to the hypothesis could be a good indicator for the segment itself. In this approach, we have already defined the desired number of selected frames, but the selected frames are dependent on the hypotheses[30].

**2.3.4 Combination based fixed frame Selection**

There are limitations contained in the position based frame selection and the ML based frame selection implicitly. For example, the ML based frame selection does not enforce the selected positions to span over the full segment. When a small part of a phoneme segment is very close to an incorrect phoneme model, all selected frames could originate from the small part, resulting in a mis-classification. For the position based frame selection, frames at some fixed positions can be noisy or less informative by chance for unknown reasons, and thus result in low likelihoods for the correct phoneme class. We find that in fact the advantages and weaknesses of these two methods are complementary, hence it is possible to combine them: the main idea of the combination is that frames lying at the appropriate positions and at the same time having high likelihoods would have higher opportunity to be selected than other frames. In other words, two factors are evaluated: selecting too closely spaced frames and selecting frames with low likelihoods would be penalized and thus prohibited. Therefore we propose two methods to combine two properties, namely likelihood prior combination and position prior combination. In the combination based fixed frame selection, we only investigate the situation of one frame for one HMM state. In this case, the decision rule degenerates to:

The last item represents to what extent the selected frames are representativfor the the full set

Figure 2-3. The linear penalty for the Combination based method

2.2 Approaches for Feature Extractor:

Feature extraction involves reducing the amount of resources required to describe a large set of data. When performing analysis of complex data one of the major problems stems from the number of variables involved. Analysis with a large number of variables generally requires a large amount of memory and computation power, also it may cause a [classification](https://en.wikipedia.org/wiki/Statistical_classification) algorithm to overfit to training samples and generalize poorly to new samples. Feature extraction is a general term for methods of constructing combinations of the variables to get around these problems while still describing the data with sufficient accuracy.

The best results are achieved when an expert constructs a set of application-dependent features a process called feature engineering. Nevertheless, if no such expert knowledge is available, general dimensionality reduction techniques may help [5]. These include:

* Independent component analysis
* Isomap
* Kernel PCA
* Latent semantic analysis
* Partial least squares
* Principal component analysis
* Multifactor dimensionality reduction
* Nonlinear dimensionality reduction
* Multilinear Principal Component Analysis
* Multilinear subspace learning
* Semidefinite embedding
* Autoencoder
* Deep feature synthesis

2.1.1 Need for Feature Extraction:

As Discussed above in the introduction part, there are numerous techniques for feature extraction in the speech signal. The objective of the block is to use a priori knowledge to transform an input in the signal space to an output in a feature space to achieve some desired criteria. Because a priori knowledge is used, the feature extraction (FE) block is usually a static module that once designed will not appreciably change. The criteria to be used depend on the problem to be solved. For example, if a noisy signal is received, the objective is to produce a signal with less noise; if image segmentation is required, the objective is to produce the original image pulse border maps and texture zones; if lots of clusters in a high dimensional space must be classified, the objective is to transform that space such that classifying becomes easier, etc.

The FE block used in speech recognition should aim towards reducing the complexity of the problem before later stages start to work with the data. Furthermore, existing relevant relationships between sequences of points in the input space have to be preserved in the sequence of points in the output space. The rate at which points in the signal space are processed by the FE block does not have to be the same rate at which points in the feature space are produced. This implies that time in the output feature space could occur at a different rate than time in the input signal space. *A priori* knowledge concerning which are the relevant features that should be used in a speech recognition problem comes from very different sources. Results from biological facts, such as the EIH model (which is based on the inner workings of the human hearing system), descriptive methods (like banks of pass band filters), data reduction techniques (such as PCA), speech coding techniques (such as LPC Cepstrum), and neural networks (such as SOM), have been combined and utilized in the design of speech recognition feature extractors. An important result obtained by Jankowski et al, which summarizes all of the above mentioned *a priori* knowledge, suggests that under relatively low noise conditions all systems behave in similar ways when tested with the same classifier. This explains why the speech recognition community has adopted the LPC Cepstrum, which is very efficient in terms of computational requirements, as the method of choice.

2.2.2 Modelling Techniques:

The objective of modelling technique is to generate models using speaker specific feature vector. This modelling technique is divided into two classifications such as speech identification and recognition. The speech identification technique automatically identifies speech which is in the trained database. The speech recognition is also divided into two parts that means speaker dependent and speaker independent. In the speaker independent mode of the speech recognition, the computer should ignore the speaker specific characteristics of the speech signal and extract the intended message. On the other hand, in case of speaker dependent recognition machine should extract speaker characteristics in the acoustic signal [35]. The main aim of speech identification is comparing a speech signal from an unknown speech to a database of known speech. The system can recognize the speech, which has been trained with a number of speakers [36]. The modelling approach for speech recognition process is described as below.

**The Acoustic Phonetic Approach:**

This method is definitely viable and has been studied in great depth for more than 60 years. This approach was based upon theory of acoustic phonetics and postulates [37]. The earliest approaches to speech recognition were based on acoustic phonetic approach. It is assumed in the acoustic-phonetic approach that rules governing the variability are straightforward and can be readily learned by a machine [38]. Formal evaluations conducted by the national institute of science and technology (NIST) in 1996 which demonstrated that automatic language identification (LID) used the phonetic feature of speech signal and discriminate among set of languages [39]. For speech recognition system the phone based approach demonstrates good performance [21, 22]. Phone recognition, Gaussian mixture modeling, and support vector machine classification are two techniques have been applied to the language identification. The acoustic phonetic approach has not been widely used in most commercial applications .

***i. Pitch Synchronous Overlap and Add Technique***

The PSOLA (Pitch Synchronous Overlap and Add) technique is the dynamic technique for pitch estimation. In the basic TD-PSOLA (Time Domain PSOLA) system, prosodic modifications are made directly on the speech waveform. The same approach can also be applied on the error detection in signal resulting from the LPC analysis. The Pitch marks are calculated and denoted on the error signal, which is then divided into a stream of short-term signals, synchronized to the local pitch cycle. The second stage is to warp the reference sentence in order to align it with the target sentence. Once this is done, the sequence of short-term signals of the reference speaker is converted into a modified stream of synthesized short-term signals synchronized on a new set of time instants. This new set of time instants is determined in order to comply with the desired modifications. A new error signal is then obtained by overlapping and adding the new stream of synthesized short-term signals. The last step is to synthesize the synchronized signal to the new pitch marks.

***ii) Spectrum Parameter Modification Algorithm***

In the formant analysis the moving frequency around the circle is result inloss of voice quality individually. Along with bandwidth and gain changes, can be achieved through direct changes to the filter. This technique is also known as speech morphing technique, which changes in spectrum parameter and fundamental frequency [42]. In this algorithm the first stage is to find the pitch marks of each speaker’s utterance and to create a correspondence table to match the source and the target. The values are set for the amount of modification desired is targeted in second step. The third step is to make the necessary spectrum modifications.

***iii) Cross-Correlation Technique***

Cross-correlation is calculated between two consecutive pitch cycles. The cross-correlation values between pitch cycles are higher (close to 1) in voiced speech than in unvoiced speech.

***iv) Filter Bank Analysis Technique***

In signal processing, a filter bank is an array of pass filters that separates the input signal into multiple components, each one carrying a single frequency sub of the original signal. One application of a filter bank is a graphic equalizer, which can attenuate the components differently and recombine them into a modified version of the original signal. The process of decomposition performed by the filter bank is called analysis (meaning analysis of the signal in terms of its components in each sub-band); the output of analysis is referred to as a sub band signal with as many sub bands as there are filters in the filter bank. The speech synthesis means the reconstruction of completer signal resulted from filtering process. The vocodor uses a filter bank to determine the amplitude information of the subbands for a modulator signal (such as a voice). It is used to control the amplitude of the sub-bands of a carrier signal (such as the output of a guitar or synthesizer), thus it is commanding the dynamic characteristics of the modulator on the carrier.

2.2.3 Robust Techniques for Feature Extraction:

Eventhough there are several technique for feature extraction from the speech signal, they all satisfies some problems and fails in some particular aspect. Comparing from several papers [9] [28] [30], I came up with discussing two main feature extraction techniques for speech recognition. And I also modelled those techniques in python. The two main techniques are

1. MFCC (Mel Frequency cpetral coefficient)
2. LPC (Linear Predictive coding)

In the following section we will discuss in the pre mentioned techniques in detail.

2.2.3.1 MFCC

Mel Frequency Cepstral Coefficients (MFCC) technique is the robust and dynamic technique for speech feature extraction. The Mel-frequency Cepstrum Coefficient (MFCC) technique is often used to extract important feature of sound file. The MFCC are based on the known variation of the human ear’s critical bandwidth frequencies with filters spaced linearly at low frequencies [11]. In this technique the logarithm of high frequencies used for capture the important characteristics of speech. From the literature it is observed that human perception of the frequency contents of sounds for speech signals does not follow a linear scale. Thus for each tone with an actual frequency, f, measured in Hz, a subjective pitch is measured on a scale called the Mel scale. The Mel-frequency scale is linear frequency spacing below 1000 Hz and a logarithmic spacing above 1000 Hz. As a reference point, the pitch of a 1 kHz tone, 40 dB above the perceptual hearing threshold, is defined as 1000 Mels.

The following formula is used to compute the Mels for a particular frequency:

A step by step implementation of the MFCC is shown in Figure 3.2 [12]. In the Preemphasis each of the speech samples is sampled to 16000 Hz for analysis purpose. The sample speech signal was pre-emphasized with filter. In the pre-emphasized the signal is blocked onto the frame of N sample, with adjacent frame being separated by M. Finally, the log Mel spectrum was converted into time. The output is called Mel Frequency Cestrum Coefficients (MFCC).The MFCC is real numbers and it can be converted into time domain using the Discrete Cosine Transform (DCT). The MFCC is used to discriminate the repetitions and prolongations in natural speech [13]. The researcher used MFCC with 12, 13, 26 and 39 variations as original feature and derivative of it.

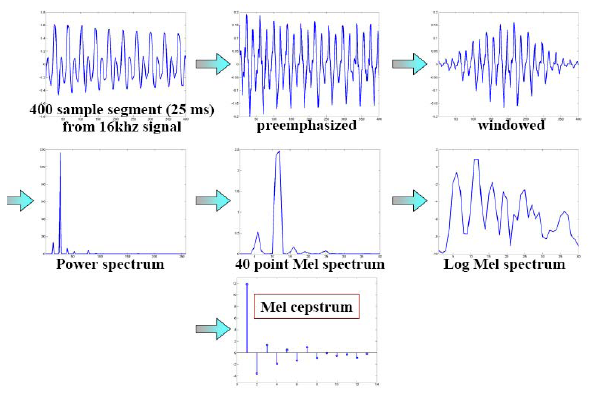


Figure 2-4. Step by Step implementation of MFCC

2.2.3.1 Linear Discriminant Analysis:

Linear Discriminant Analysis (LDA) is commonly used technique for data classification and dimensionality reduction. It easily handles the case where within-class frequencies are unequal and their performances have been examined on randomly generated test data. This method maximizes the ratio of between-class variance to the within class variance in any particular data set thereby guaranteeing maximal reparability. The use of Linear Discriminant Analysis for data classification is applied to classification problem of speech recognition. LDA algorithm provides better classification compared to principal components analysis [43]. The figure 3.3 describes the LDA training after projection.

2.3 Deep Neural Network.

This section is intended to provide an overview about how deep architectures have emerged as a new paradigm in recent years and why they have become a breakthrough within the machine learning field, transferring, at the same time, their influence to other interesting research fields as it can be the speech processing area. Furthermore, some basic concepts about learning algorithms will be summarized, such as the differences between what is called supervised and unsupervised learning algorithms, things that make shallow and deep architectures different, and, finally, some ideas about how to train deep architectures unlike the shallow ones.

2.2.1. A New Machine Learning Paradigm

Broadly speaking, the main aim of artificial intelligence could be stated as \make computers be able to model our world" [12]. That implies to process a large quantity of information to answer questions and generalize to new contexts. Here, it is where machine learning algorithms play an important role. During lots of years, much progress has been made in this field, but the challenge of the artificial intelligence field still remains: computers cannot understand well enough real scenes or describe them in natural language [12]. One of the reasons which make human brain and learning algorithms performance so different is the way both extract useful information from the data. It is believed that human brain acts as a feature extractor that, gradually, gets information from the data at different levels of abstraction. In other words, it tries to decompose the problem to solve into some sub-problems with less complexity, obtaining, at the same time, different levels of representation. This behaviour is what machine learning structures and algorithms have tried to imitate. Firstly, the system captures low-level features that are invariant to small variations (for example, geometric variations if we talk about a computer vision task), transforms them to increase their invariance, and, finally, extracts useful information, which means frequent patterns that could be generalized to other data.

All this process of extracting useful information from raw input data makes necessary to have a structure with the ability of transforming the input in a non-linear way. In these terms, learning algorithms should be able to apply mathematical transformations or functions that are highly non-linear and varying. Most of structures that have been used during years within the machine learning field have not enough capability to apply those complex functions that are necessary to solve certain problems. Classic architectures have usually one hidden layer that implies just one non-linear transformation of the data. The idea of deep architectures comes from this point: if more transformations are required, more hidden layers will be stacked. But this has many consequences for learning algorithms, such as the more complex the function to model is, the more local minimum can be found. Thus, training algorithms that had been used for many years, did not work with these new architectures, known as \deep schemes" due to their higher number of layers respect to the previous ones. Backpropagation and gradient descent algorithms did not yield good results in experiments that used these new architectures, since weights and other parameters converged into really small values, close to zero. There was an exception: Convolutional Neural Networks. This deep architecture yielded the first successful results when Yann Lecun used his structure known as LeNet" for classification tasks [44]. In this architecture, the amount of free parameters that have to be learnt is reduced thanks to many of those parameters are shared between different parts of the network. However, these networks work in a supervised way, so they need a huge amount of labelled data to train, which can be considered a drawback for some problems where it is not easy to get these labels.

2.2.2. Supervised and Unsupervised Learning

A learning algorithm can be defined as a form to calculate a prediction (output) from input data. This prediction is, indeed, a function that matches inputs with outputs.

The concept of learning itself is related to generalization. In a classification task, this can be defined as the ability to categorize correctly new examples that differ from those used for training. Thereby, datasets used in machine learning tasks should be split into, at least, two parts:

* Training data: This set covers the samples used to estimate the parameters of the objective function, with the objective of minimizing the error, for instance, the misclassified examples in a classification task.
* Test data: It is composed of examples that let prove the system. These examples are not included in the training dataset and, thus, they do not interfere in the parameters selection.

Regarding learning algorithms, they can be divided into two main groups: 1) supervised or 2) unsupervised algorithms, depending on the information that the system uses for training (labelled or unlabelled data). There are also semi-supervised algorithms and what is known as reinforcement learning, but we will focus on the first two cases.

1. Supervised learning

If we talk about a classification task, which is a machine learning problem where the output of the system should be the class or category of a certain input, supervised learning includes algorithms that use the information of the class to train. This modality requires which is called labelled data. In this manner, the cost function to minimize can be an error function that measures the difference between the predicted class and the actual class of a certain input. Among this kind of algorithms, it can be highlighted Logistic Regression, Multilayer Perceptron and Deep Convolutional Network, due to their relation with this work.

2. Unsupervised learning

This set of algorithms is also known as algorithms based on clustering. The most known example is the K-means algorithm that tries to create groups among the data based on some measure of similarity between them. These algorithms can be understood as a way of modelling the distribution of the input data too. From this point of view, unlike supervised learning methods, they cannot measure an error between the predicted class and the actual class since they do not have the information about the actual class. Instead, they can try to minimize a \reconstruction error", and this may be the cost function that the algorithm will consider to train.

Within this set, Auto-Encoders and Restricted Boltzmann Machines (RBMs) are included. Apart from these two groups, due to its relation with this work, we can mention a semi- supervised learning algorithm: Deep Belief Networks. This architecture uses RBMs as building blocks, so they are trained in an unsupervised way. But its training has also a fine-tuning" phase, that constitutes a supervised learning algorithm.

**From Shallow to Deep Architectures:**

As it was commented before, from a theoretical point of view, shallow architectures present limitations that deep schemes can potentially solve.

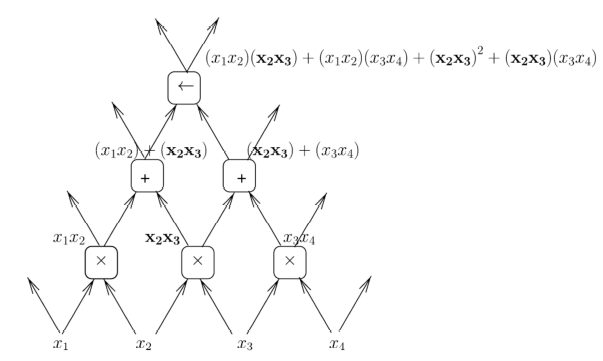


Figure 2-5 . Expression Computed by Deep Architecture with sums and products

The main motivation to explore deep architectures lies on the idea that some functions cannot be efficiently represented, which means with a reasonable number of parameters to tune by learning, by architectures that are too shallow. When this happens, it is said that a function is compact [12]. Thus, a function that can be represented in a compact form by certain architecture may carry an exponential increase in the number of parameters to tune if other architecture with one layer less is used instead. All this implies that, if the amount of data available to train is not enough with respect the number of parameters to be tuned, the representation that will be achieved by a shallow network can be not as good as the representation that a deep architecture could accomplish. Figure 2-5 extracted from [12] shows an example of an expression where the term x2x3 occurs more than once, and how a deep architecture can avoid repeating that computation many times.

With the structure shown in figure 2-5, the number of parameters or connections to tune by learning would be 12, while with the shallow structure represented by figure 2.6 the amount of parameters increases up to 20. Although all the advantages that deep architectures seem to present versus the shallow structures, the big amount of variations within input data from a certain problem, for example data from a vision or audio context, is still a problem for machine learning algorithms. Also, it should be taken into account that not every problem in the world is so difficult that a deep architecture is required to solve it, so that the simplest structure that fits the problem should be used in those cases.

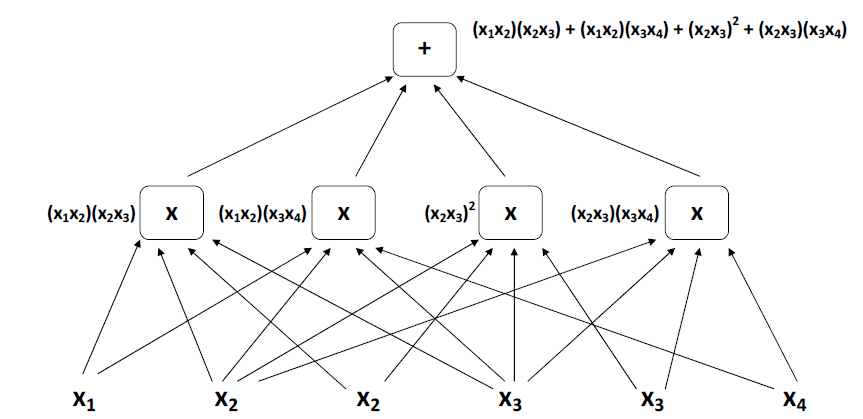


Figure 2-6 . Same expression from figure 2 computed by a shallow architecture

Regarding the issue of how to build a deep architecture, some ideas about it will be presented below, according to the software and architectures [LISA] used in the experimental part of this work. In that sense, shallow structures can be seen as building blocks for deep ones. For example, if a classification task based on supervised learning is being implemented, the structure shown in figure 2.7 could be used. That architecture is composed of the next elements:

* An input layer, which just represents the input used to feed the network.
* Hidden layers, that apply non-linear transformations to the input data, such as, for example, a sigmoid or tanh functions defined as follows:

It can be added as much hidden layers as it was necessary for the problem to solve, building a structure where the input of one hidden layer is the output of the previous one.

* A logistic regression classifier, which is the output layer and performs the classification maximizing a cost function, as, for example the one defined as follows:

Where the selected class for a sertain input would be:

The parameters to be tuned, in that case, would be the weight matrix W, and the bias vector b, that can be learnt by using, for instance, a gradient descent algorithm. Following the same idea, but using a semi-supervised learning algorithm, Deep Belief Network (DBN) constitutes an architecture where Restricted Boltzmann Machines (RBMs) are used as building blocks for pre-training [14]. Finally, a logistic regression classifier can be added as output layer to perform the same classification task as the one mentioned before.

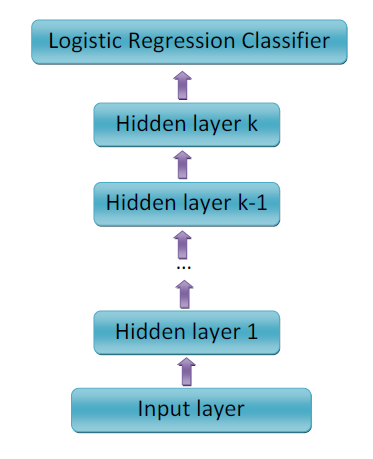


Figure 2-7. Example of deep architecture for classification

2.2.3 Case of Success

The previous sections had deeply illustrated the general related works and techniques for different part of the speech recognition model. There are numerous way to implement frame selection method as described in section 2.1, two most reliable technique to convert speech signal to feature input vector which is described in section \_\_ . These are the primary part for the speech recognition system, but the most vital part in the speech recognition model. Three different way to implement the speech recognition model are been illustrated in section 1.2. Among the three types of implementation, the Neural network model of building the speech recognition system provide most reliable and promising results, but this model need some serious improvement for attaining state of the art performance. This was the primary reason for developing Deep Neural Network model for Speech recognition system. The overview and basic concepts of deep neural network is been described in 2.3 section. But those are the general concept for modelling a deep neural network for Speech recognition. This section describe different types of deep neural network for attaining state of the art performance and accuracy.

2.2.3.1 Recurrent Neural Network:

A recurrent neural network is one in which each layer represents another step in time (or another step in some sequence), and that each time step gets one input and predicts one output. However, the network is constrained to use the same “transition function” for each time step, thus learning to predict the output sequence from the input sequence for sequences of any length. A recurrent neural network (RNN) is a class of artificial neural network where connections between units form a directed cycle. This creates an internal state of the network which allows it to exhibit dynamic temporal behavior. Unlike feedforward neural networks, RNNs can use their internal memory to process arbitrary sequences of inputs. This makes them applicable to tasks such as unsegmented connected handwriting recognitionor speech recognition.

For a standard recurrent neural network, we iterate the following equations in order to do prediction:

*hi=σ(Whhhi−1 + Whxxi + bh)*

The hidden layer at step is given by *i*; similarly, *i* is the input layer at timestep , *i* is output layer at timestep , and the are the weight matrices (with biases ).

Note that this formulation of recurrent networks (RNNs) is equivalent to having a one-hidden-layer feed-forward network at each timestep (with layers ). One can also consider *i-1* to be part of the input layer at each timestep.

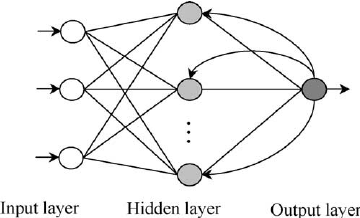


Figure 2-8 . Simple Recurrent Neural Network

**Long Short-Term Memory**

Although standard RNNs are powerful in theory, they can be very difficult to train.

Techniques such as Hessian-free optimization have been applied to them in order to improve training capacity. However, in addition to modifying the training algorithm, we can modify the network architecture to make it easier to train. One of the reasons training networks is difficult is that the errors computed in

backpropagation are multiplied by each other once per timestep. If the errors are small, the error quickly dies out, becoming very small; if the errors are large, they quickly become very large due to repeated multiplication. An alternative architecture built with *Long Short-Term Memory* (LSTM) cells attempts to negate this issue.

A single LSTM unit is shown below.

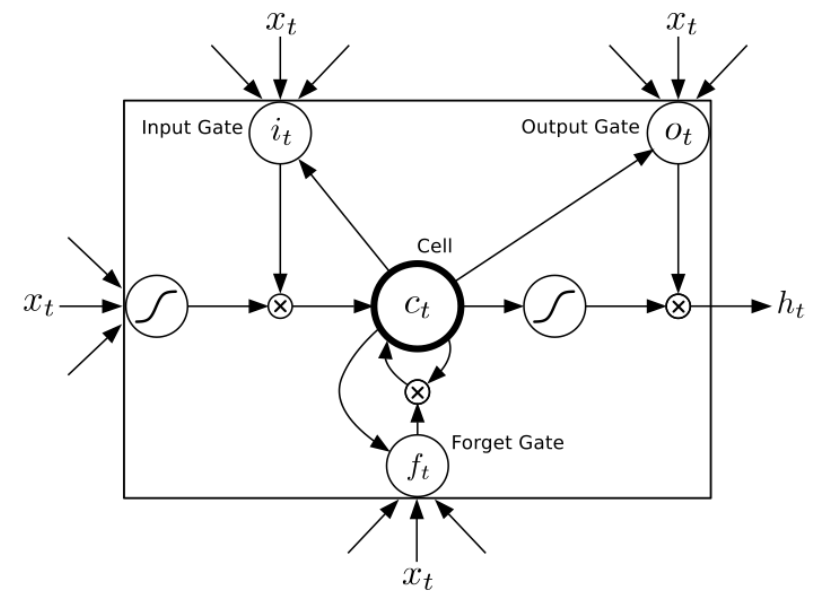


Figure 2-9. Single LSTM unit

The inputs t dictate the behaviour of our LSTM cell. Note that each of the cells (circles) shown above may actually be vectors, and can store a large set of values.

The intuition behind this memory unit is that the *cell* stores a value over time. It feeds itself and can either remember or forget its value, depending on the activation of the *forget gate t* . The cell can optionally output its value, depending on the activation of the *output gate* . Finally, the cell acquires new values if the *input gate* allows it to. Note that in the diagram above, the symbols indicate these complex activation functions that allow these behaviours.

Next, we present the equations that implement the LSTM unit. First of all, the cell must forget its value if the forget gate is low and acquire a new value if the input gate is high. The value it acquires is dictated by the previous hidden layer and the current input. Thus, is determined via the following equation:

Note that when, the cell completely forgets its old value, and stores a new one. When, the new value is given by the inputs and the previous hidden layer; if the input gate is off, though, the cell is either unchanged (for) or simply set to zero (for ). In this manner, the cell implements the main memory storage function of the unit. In order to let and range from zero to one, we let their activation functions be the standard sigmoid, a differentiable approximation to the step function. Both of these are fairly standard, and depend on the input and previous values of the cell and hidden layers:

The only caveat to these is that we enforce a constraint on the weight matrices from the cell to the gate. Recall that the gate is actually a vector of cells. We enforce the c constraint that the th gate element depends only on the *m*th cell element, so that each element of the LSTM unit acts independently. We encode this constraint by enforcing that is a diagonal matrix. In order to get the data *out* of the cell, we have a custom output gate. The output gate is computed just like the other two gates:

The output gate controls whether the hidden state comes from the cell:

**Bidirectional RNNs**

In a standard RNN, the output at a given time depends exclusively on the inputs through (via the hidden layers through). However, while this makes sense in some contexts, many sequences have information relevant to output both before timestep and after timestep . In speech recognition, specifically, the sound before and after a given point gives information about the sound at a particular point in the sequence. In order to utilize this information, we need a modified architecture. There are several approaches possible approaches:

**Windowed Feed-Forward Network**: Instead of using an RNN, simply use a window around the output and use a standard feed forward network. This has a benefit of being easier to train; however, it limits applicability because we must have a window of *exactly* that size, and because we do not use information far away from the output (the size of the window is limiting).

**RNN with delay**: Instead of predicting timestep after seeing inputs 0 through , predict timestep after seeing inputs 0 through , where is some fixed delay. This is fairly close to a standard RNN, but also lets you look a few steps in the future for contextual information.

**Bidirectional RNN**: Add another set of hidden layers to your recurrent network going *backwards* in time. These two hidden layers are entirely separate and do not interact with each other, except for the fact that they are both used to compute the output. Given your weights, you need to run propagation forward in time (from time 0 to the end) to compute the forward hidden layers, and run it backward in time (from the end to time 0) to compute the backward hidden layers; finally, using the values at both of the hidden layers for a given timestep , compute the output at every timestep .

The paper that introduced bidirectional RNNs (<http://www.cin.ufpe.br/~fnj/RNA/BRNN.pdf>) (by Schuster and Paliwal) has two graphics that are very helpful for understanding them and the differences from these other approaches. First of all, we can visualize what part of a sequence each type of network can utilize in order to predict a value at time :

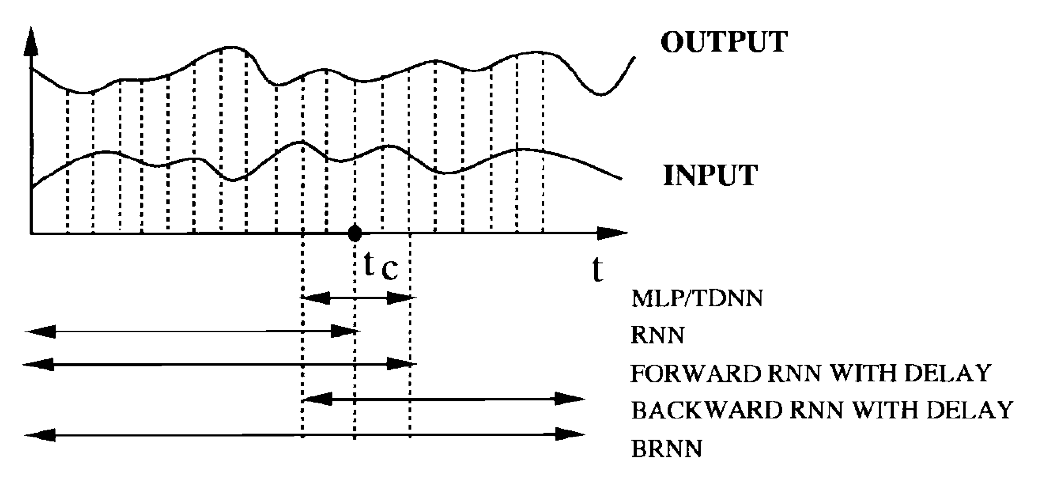


Figure 2-10. Bidirectional RNN

The windowed approach is labeled **MLP** (for multilayer perceptron). Standard RNNs are labeled **RNN**, and utilize information right up to. Delayed RNNs (going forward and backward) can use all their history, with an extra window around. Finally, bidirectional RNNs (**BRNN**s) can use the entire sequence for their prediction. Graves et. al. propose using LSTM units in a bidirectional RNN for speech recognition, so we focus on that approach. It can be trained similar to a standard RNN; however, it looks slightly different when expanded in time (shown in the graphic below, also from Schuster and Paliwal).

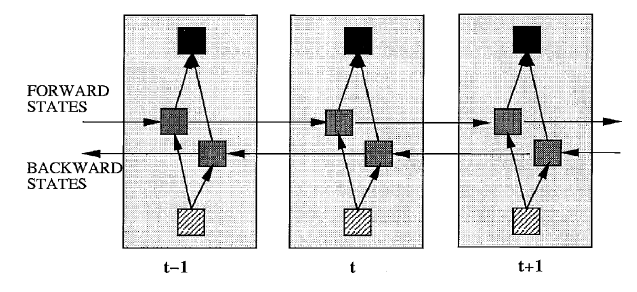


Figure 2-11. Bidirectional RNN expanded in time

Here we see the BRNN expanded in time, showing only the timesteps around timestep . We see that in the middle we have two hidden states (gray), one propagating forwards and one propagating backwards in time. The input (striped) feeds to both of these, and both of them feed to the output of the RNN (black).

2.2.3.2. Convolutional Neural Network

A typical CNN has two major parts: a convolutional modules followed by several fully connected layers. The convolutional module uses two fundamental types of layers: the convolutional layer followed by a pooling layer. A convolutional layer performs convolution operations to generate output values from local regions (often called receptive fields due to the use with images) of feature maps of the previous layer, and all nodes/neurons in one feature map share the same filter. The convolution operation can be expressed as

where ) and are two feature maps in two consecutive layers. The convolutional operation (denoted as ∗) is performed with the filter ) and the feature map ). The bias (l) is then added and finally the activation function typically using a sigmoid or a rectified linear units(ReLU), is applied to generate the the convolutional layer outputs. When multiple feature maps are present in the previous layer, all the results of convolutional operations are accumulated first before adding the bias. The pooling layers perform down-sampling on the feature maps of the previous layer and generate new feature maps with a reduced resolution. Several pooling strategies have been investigated [9], including max-pooling, stochasticpooling [18], etc, and all of them have competitive performance. In this work, max-pooling is used in all CNN models. The most popular configuration for CNNs used in speech recognition is the setup in [9], which has two convolutional layers with 256 feature maps in each, and it uses filters with pooling in the first convolutional layer, 3 × 4 filters in the second convolutional layer without pooling. Finally, there are four fully connected layers each of 2048 hidden nodes in standard multi-layer perceptron (MLP) structure. This setup is also used as the baseline CNN in this paper

**Very Deep CNNS:**

Our previous work in [14] introduced very deep CNNs for ASR of conversational telephone speech, and verified the promising potential of this kind of model. Based on this initial work, the models are here further developed with a focus on improving performance for noisy speech. Before the detailed architecture is described, some fundamental principles of VDCNNs for speech recognition are presented and explained:

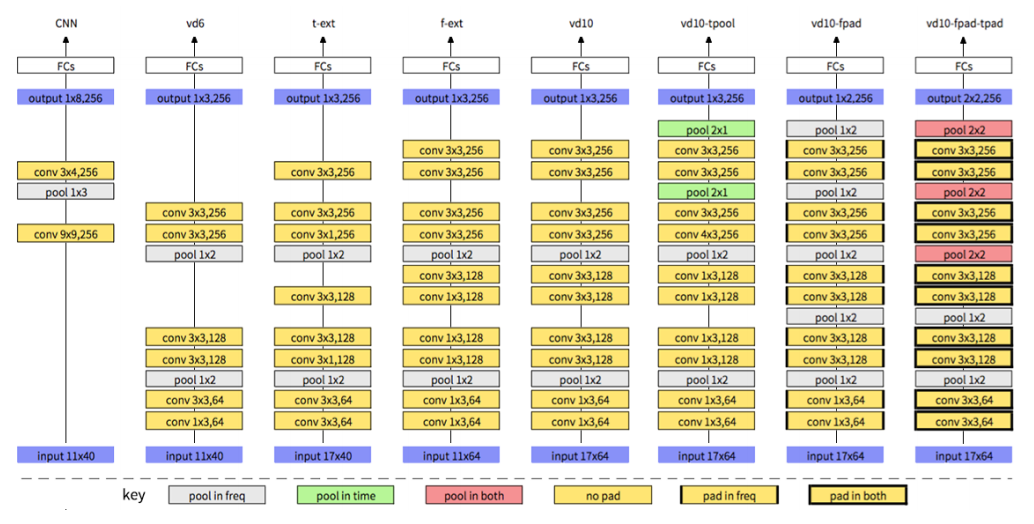
****

Figure 2-12 . Very Deep CNNs Architecture

• Rather than using the traditional 9×9 or 3×4 filters and 1×3 pooling [7, 8, 9], VDCNNs for speech recognition use filters of 3 × 3 (sometimes 1 × 3 and 3 × 1), and the pooling size is constrained to 1 × 2 or 2 × 2. This is also similar as the work in image classification[11]. The stride of the convolution is set to 1 and only nonoverlapping pooling is used in this work. These are the smallest reasonable sizes for the filters and pooling respectively [11], which enables the models to be built with as many convolutional layers as possible.

• Compared to computer vision tasks, the size of neural network inputs in speech recognition is relatively small since the context window and basic feature (e.g. FBANK) dimension are much smaller in most of speech recognition systems1[1]. Accordingly, in addition to the adjustment of the size of filters and pooling, the size of inputs need to be enlarged appropriately for speech recognition to allow more convolution and pooling operations. All the proposed very deep CNNs in this work only one input feature map, i.e. the static feature map, is used unless otherwise noted.

• For the very deep CNNs, a pooling layer is added after at least two convolutional layers. The feature map size before the first fully-connected layer is set to a relatively small value in our proposed VDCNNs, e.g. 1×3 or 1 × 2. In addition the number of feature maps is increased gradually and doubled after some pooling layers. For all the model configurations in this paper, the number of feature maps is increased with subsequent layers and in sequence uses 64, 128 and then 256 feature maps.

• As well convolutional layers and associated pooling layers, 4 fully-connected (FC) layers are added, with an output layer using a softmax activation function. The ReLU is used for the hidden nodes in all proposed VDCNNs to overcome the gradient-vanishing problem

Following these fundamental principles, the very deep CNNs are developed. One model, named vd6, is introduced as the first very deep CNN model, whose configuration can be found in Figure 1. The model vd6 shares the same context window size and feature dimension, i.e. 11 × 40, with the traditional CNN shown as the first column of Figure 1. Following these fundamental principles, 5 convolutions can be performed in time, and 6 convolutions and 2 poolings can be performed in frequency, which results in a VDCNN with 6 convolutional layers and 2 pooling layers. Further modi- fications are based on vd6, and all the discussed structures in this work are illustrated in Figure 1: different types of pooling layers are indicated with different colours, and convolutional layers with various padding strategies are marked with corresponding border styles.

**2.3.3 Deep Belief Networks:**

Deep Belief Networks (DBNs) are neural networks consisting of a stack of restricted Boltzmann machine (RBM) layers that are trained one at a time, in an unsupervised fashion to induce increasingly abstract representations of the inputs in subsequent layers.

**2.3.3.1 Restricted Boltzmann Machines (RBMs ) and Training:**

As shown in Fig. 2 (a), Each RBM has an input layer (visible layer) and a hidden layer of stochastic binary units. Visible and hidden layers are connected with a weight matrix and no connections exist between units in the same layer. Signal propagation can occur in two ways: recognition, where visible activations propagate to the hidden units; and reconstruction, where hidden activations propagate to visible units. The same weight matrix (transposed) is used for both recognition and reconstruction. By minimizing the difference between the original input and its reconstruction (i.e. reconstruction error) through a procedure called contrastive divergence (CD), the weights can be trained to generate the input patterns presented to the RBM with high probability. The RBM pretraining procedure of a DBN can be used to initialize the weights of a deep neural network, which can then be discriminatively fine-tuned by back-propagating error derivatives. The “recognition” weights of the DBN become the weights of a standard neural network. In cases where the RBM models the joint distribution of visible data and class labels, a hybrid training procedure can be used to fine-tune the generatively trained parameters.

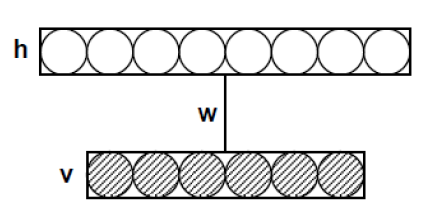


Figure 2-12 . Restricted Boltzmann Machines

**2.3.3.2. DBN Structure.**

Fig. 2 (b) shows the structure of a DBN. A DBN consists of a stack of RBMs, trained one at a time. Each layer of hidden units learns to represent features that capture higher order correlations in the original input data. In DBNs, subsequent layers usually decrease in size in order to force the network to learn increasingly compact representations of its inputs. The training procedure is sometimes augmented to optimize additional terms, such as the L1 and L2 norms of the weight matrices, or sparsity constraints on the unit activations. Weights are initialized from a normal distribution with zero mean and small standard deviation. Weight updates are applied after the presentation of a number of samples in a minibatch. After a number of training cycles through the full training dataset, the stack of RBMs is unfolded, such that first recognitions are computed through all subsequent layers, and next reconstructions through all layers in reverse order. The recognition and reconstruction weights are uncoupled, and can then be fine-tuned with gradient descent, either to become better at reconstructing the inputs, or — in combination with other supervised or reinforcement learning methods — to form features relevant to the task at hand.

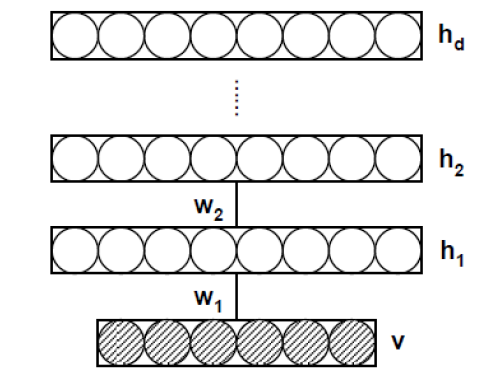


Figure 2-13 . Deep Belief Network

**3.3 Applying DBNs for Speech Recognition**

To apply DBNs with fixed input and output dimensionality to phone recognition, a context window of n successive frames of feature vectors is used to set the states of the visible units of the lower layer of the DBN which produces a probability distribution over the possible labels of the central frame. To generate speech sequences, a sequence of probability distributions over the possible labels for each frame are fed into a standard Viterbi decoder.

[2] - C. R. Jankowski Jr., H. H. Vo, and R. P. Lippmann, “A Comparison of Signal Processing Front Ends for Automatic Word Recognition,” IEEE Transactions on Speech and Audio processing, vol. 3, no. 4, July 1995.

[3] - K. Torkkola and M. Kokkonen, “Using the Topology-Preserving Properties of

SOMs in Speech Recognition,” Proceedings of the IEEE ICASSP, 1991.

[4] - S. Furui, “Digital Speech Processing, Synthesis and Recognition,” Marcel Dekker

Inc., 1989.

[5] - Alpaydin, Ethem (2010). *Introduction to Machine Learning*. London: The MIT Press. p. 110. ISBN 978-0-262-01243-0. Retrieved 4 February 2017.