Hybrid HMM/DTW based Speech Recognition with Kernel Adaptive Filtering Method

Siva Prasad Nandyala\textsuperscript{1} and T. Kishore Kumar\textsuperscript{2}

\textsuperscript{1} Research Scholar, \textsuperscript{2} Associate Professor, Department of E.C.E, NIT Warangal, Andhra Pradesh, India

\textbf{ABSTRACT}

We have proposed new approach for the speech recognition system by applying kernel adaptive filter for speech enhancement and for the recognition, the hybrid HMM/DTW methods are used in this paper. Noise removal is very important in many applications like telephone conversation, speech recognition, etc. In the recent past, the kernel methods are showing good results for speech processing applications. The feature used in the recognition process is MFCC features. It consists of a HMM system used to train the speech features and for classification purpose used the DTW method. Experimental results show a relative improvement of recognition rate compared to the traditional methods.

\textbf{KEYWORDS}

Speech Enhancement, Hidden Markov Model, Dynamic Time Warping, Kernel Least Mean Square, Least Mean Square, Signal to Noise Ratio

\textbf{1. INTRODUCTION}

In many quality progressive speech recognition systems, Hidden Markov Models (HMMs) are used to model the acoustic behavior of basic speech units like words or phones. The existence of ground noise can cause degradation inside the standard and quality of the speech signal in many applications, as associate example, speech recognition system, conference systems, mobile communications and hearing aids,\cite{1} \cite{12}. The speech enhancement in the above application is a challenging task, the main reason is the noise signals nature and characteristics may change dramatically in time and also on application to application, for this reason, an effective speech enhancement technique to recover the clean speech from its corrupted observation is necessary \cite{11} \cite{13}. It is required to estimate the speech signal from the observed noisy one in the above applications.

Lot of different noise reduction algorithms have been proposed in literature to reduce the noise effect for speech enhancement, if we take for example, spectral subtraction, Wiener filtering for noise reduction, minimum mean square error (MMSE) based speech estimation, Least Mean Square algorithm, Bayesian based estimation and kalman filtering.

\textbf{2. ADAPTIVE FILTERS}

There are many approaches are present among the literature related to speech enhancement. For the last 20 years, adaptive filters are effective and popular approaches for the speech enhancement.
The advantage of adaptive filters is that they find time varied potentials and conjointly track the dynamic variations of the signals.

Before progressing to the LMS and KLMS, associate introduction to adaptive filter is given as follows. Because the name is “adaptive filters” it’s going to be vital to know that means of the terms “adaptive” and “filter” during a very general sense. The adjective adaptation will be understood by considering a system that is making an attempt to regulate itself thus on reply to some development that's going down in its close. In different words the system tries to regulate its parameters with a aim of meeting some well outline goal or target that depends upon the state of system moreover as its close. this is what adaptation suggests that. Furthermore there's a necessity to own a collection of steps or sure procedure by that this method of adaptation is administered. And at last the system that carries out or undergoes the method of adaptation is termed by the additional technical name “filter”.

The adaptive filter has several benefits which incorporates lower process delay and higher trailing of the mechanical phenomenon of nonstationary signals. These area unit necessary characteristics in applications like noise estimation, echo cancellation, echo cancellation delay estimation and channel equalization in mobile telecommunications, wherever low delay and quick wrenching of time-varying processes and time-varying environments area unit necessary objectives [2].

The mathematical definition of adaptive filter is given below as follows

\[ y(k) = \sum_{n=0}^{M-1} W_n(k)x(k-n) = W^T(k)x(k) \]  

\[ \text{eq. (1)} \]

In which k is the time index, y is the filter output, x the filter input, Wn are the filter coefficients.

The below figure 1 shown in the below block diagram, gives the foundation for particular adaptive filter realisation [3], which is named as Least Mean Squares (LMS). The main concept that lies in the block diagram is, a variable filter extracts the estimate of the desired signal.

![Block diagram of adaptive filter](image)

From the block diagram, the following assumptions are made:

1) The input signal \( x(n) \) which is the sum of a desired signal \( d(n) \) and interfering noise \( v(n) \)

\[ x(n) = d(n) + v(n) \]  

\[ \text{eq. (2)} \]
The variable filter has the structure of Finite Impulse Response (FIR). The impulse response is adequate for such structures the with the filter coefficients. For the filter of order p with the coefficients for square measure given as follows

\[ w_n = [w_n(0), w_n(1), ..., w_n(p)] \]  

\[ w_n(0), w_n(1), ..., w_n(p) \]

1) the second one is the cost function or the error signal or which is the difference between the estimated signal and desired signal

\[ e(n) = d(n) - \hat{d}(n) \]

\[ e(n) = d(n) - \hat{d}(n) \]

The desired signal is estimated by the variable filter which convolves the input signal with the impulse response. This is expressed in vector notation as follows

\[ \hat{d}(n) = W_n \ast x(n) \]

\[ \hat{d}(n) = W_n \ast x(n) \]

Where

\[ x(n) = [x(n), x(n-1), ..., x(n-p)] \]

\[ x(n) = [x(n), x(n-1), ..., x(n-p)] \]

In which \( x(n) \) is an input signal vector and the variable filter updates the filter coefficients at every time instant

\[ W_{n+1} = W_n + \nabla W_n \]

\[ W_{n+1} = W_n + \nabla W_n \]

Where, \( \nabla W_n \) is a correction factor for the filter coefficients.

This correction factor is generated by the adaptive algorithm based on the error and input signals. The Least Mean sq. (LMS) algorithmic program was initially developed by Widrow and Hoff in 1959 through their studies of pattern recognition. From there it's become one among the foremost wide used algorithms in adaptative filtering algorithms [4]. The LMS algorithmic is a stochastic gradient-based algorithm in which it utilizes the gradient vector of the filter tap weights to converge on the optimum wiener solution. It is renowned and wide used owing to its procedure simplicity. It is this simplicity that has created it the benchmark against that all alternative adaptation filtering algorithms which are judged.

The LMS algorithmic program could be a linear adaptive filter algorithmic program that normally consists of two basic processes.

1. A filter process: In this the following steps are involved
   a. Calculating the linear filter output in response to an input signal.
   b. The estimation error is generated by comparing this output with a desired response.

2. An adaptive process: This involves the automatic adjustment of filter Parameters in accordance with the estimation error.

The combination of those two processes working together constitutes a feedback loop; first, we've got a transversal filter, around which the LMS algorithm is constructed. This component is
accountable for performing the filtering process. Second, we've got a mechanism for performing the adaptive control process on filter tap weights of the transversal filter. With each iteration of the LMS algorithm, the filter tap weights of the adaptive filter are updated in keeping with the subsequent formula.

\[ w(n+1) = w(n) + 2\mu ex(n) \]  
\[ \text{eq. (8)} \]

Here \( x(n) \) is the input vector of time delayed input values,

\[ x(n) = [x(n) \ x(n-1) \ x(n-2) \ldots x(n-N+1)] \]  
\[ \text{eq. (9)} \]

The vector \( w(n) = [w_0(n), w_1(n), w_2(n), w_3(n) \ldots w_{N-1}(n)] \) represents the coefficients of the adaptive Finite Impulse Response filter tap weight vector at time \( n \). The parameter \( \mu \) is understood because the step size parameter and it is a tiny positive constant. This step size parameter controls the influence of the updating issue.

Selection of an appropriate value for \( \mu \) is imperative to the performance of the LMS rule, if value is simply too less the time, the adaptative filter takes to converge on the optimum solutions are too long; if \( \mu \) is simply too large or more the adaptative filter becomes unstable and its output diverges. is that the simplest to implement and is stable once the step size parameter is chosen suitably.

The main steps of LMS algorithm are given as follows:

The LMS rule equations which are developed primarily based upon the speculation of the wiener solution for the best filter tap weights, \( W_0 \) and it in addition depends on the steepest-descent method.

This can be a formula that updates the filter coefficients using the current tap weight vector and also the current gradient of the cost function with respect to the filter tap weight coefficient vector, \( \xi \)

\[ W(n+1) = W(n) - \mu \nabla \xi(n) \]  
\[ \text{eq. (10)} \]

Where

\[ \xi(n) = E[e^2(n)] \]  
\[ \text{eq. (11)} \]

As the negative gradient vector points within the direction of steepest descent for the \( N \)-dimensional quadratic cost function, every recursion shifts the value of the filter coefficients nearer toward their optimum value, that corresponds to the minimum possible value of the cost function, \( \xi(n) \).

By doing the random process implementation of the steepest descent algorithm, the LMS algorithm can be derived. Here the expectation for the error signal isn’t notable therefore the instantaneous value is employed as associate estimate. The steepest descent algorithmic rule then becomes

\[ W(n+1) = W(n) - \mu \nabla \xi(n) \]  
\[ \text{eq. (12)} \]

Where
The cost function gradient which is \( \nabla \xi(n) \), can be expressed in another alternative form in the following steps.

\[
\nabla \xi(n) = \nabla \{ e^2(n) \} \\
= \frac{\partial e^2(n)}{\partial W} \\
= 2e(n) \frac{\partial e(n)}{\partial W} \\
= -2e(n) x(n)
\]

Substituting this into the steepest descent algorithm of Eq.12, we arrive at the recursion for the LMS adaptive algorithm.

\[
W(n + 1) = W(n) + 2\mu e(n)
\]

The main reason is its computational simplicity behind the popularity of LMS algorithms, making it easier to implement than all other commonly used adaptive algorithms.

3. Kernel Adaptive Filters

In this proposed approach, we have used the kernel methodology which is a smart nonparametric modeling technique within which, it transforms the input data into a high dimensional feature space via a reproducing kernel such that the with the help of the kernel evaluations, scalar product operation within the feature space is computed with efficiency.

When an appropriate linear methods linear strategies are subsequently applied on the remodeled information. As long as algorithm formula is developed in terms of inner product (or equivalent kernel evaluation), there is no need to perform computations within the high dimensional feature space.[5].

The kernel adaptive filtering technique employed in this proposed approach is which depends on the adaptive filtering technique for general nonlinear issues. This itself is a natural generalization of linear adaptive filtering in reproducing kernel Hilbert spaces domain. Kernel adaptive filters are like kernel strategies, closely associated with some artificial neural networks like radial basis function networks and regularization networks [6]. The KLMS algorithm could be a random gradient methodology to unravel statistical procedure issues in RKHS. as a result of the update equation is written in terms of scalar product, KLMS is computed with efficiency within the input space. The nice approximation ability of KLMS stems from the very fact that the remodeled data embody probably infinite totally different features of the first data.

A linear finite impulse response filter is which is assumed within the LMS formula. If the mapping between desired and input is very nonlinear, then poor performance is expected from LMS [7]. therefore the kernel evoked mapping is used to rework the input \( u(i) \) into a high –
dimensional feature house $F$ as $\phi(u(i))$, $\omega^T \phi(u)$ could be a rather more powerful model than $w^T u$ thanks to the distinction in spatiotemporal representation of $u$ and $\phi(u)$. So, finding $\omega$ through random gradient descent might prove as an efficient method of nonlinear filtering as LMS will for linear issues. Denote $\phi(i) = \phi(u(i))$ for simplicity.

The new example sequence yields is modified with the help of LMS formula as follows

$$w(0) = 0 \quad \text{eq. (16)}$$

$$e(i) = d(i) - w(i - 1)^T \phi(i) \quad \text{eq. (17)}$$

$$w(i) = w(i - 1) + \eta e(i) \phi(i) \quad \text{eq. (18)}$$

Where weight update vector $w(i)$ denotes the estimate (at iteration $i$) of the weight vector in $F$. As the dimensionality of $\phi$ is high, the repeated application of the weight-update equation (18) through iterations yields

$$w(i) = w(i - 1) + \eta e(i) \phi(i)$$

$$= [w(i - 2) + \eta e(i - 1) \phi(i - 1)] + \eta e(i) \phi(i)$$

$\ldots \ldots$\n
$$= w(0) + \eta \sum_{j=1}^{i} e(j) \phi(j)$$

$$= \eta \sum_{j=1}^{i} e(j) \phi(j) \quad \text{By assumption } w(0) = 0 \quad \text{eq. (19)}$$

That is, after $i$-step training, the weight estimate is expressed as a linear combination of all the previous and present (transformed) inputs, weighted by the prediction errors (and scaled by $\eta$).

Considering mainly, the output of the system to a new input $u'$ can be individually expressed in terms of inner products between transformed inputs.

$$w(i)^T \phi(u') = \left[ \eta \sum_{j=1}^{i} e(j) \phi(u(j))^T \right] \phi(u')$$

$$= \eta \sum_{j=1}^{i} e(j) \phi(u(j))^T \phi(u') \quad \text{eq.(20)}$$

But by the kernel trick we can efficiently compute the filter output in the input space by kernel evaluations

$$w(i)^T \phi(u') = \eta \sum_{j=1}^{i} e(j) k(u(j), u') \quad \text{eq. (21)}$$

When the above equation is compared with LMS weight update equation 15, it is updated without using the weights.
If \( f_i \) is denoted as the estimate of the input – output nonlinear mapping at time \( i \), we have the following sequential learning rule for the new algorithm

\[
\begin{align*}
f_{i-1} &= \eta \sum_{j=1}^{i-1} e(j)k(u(j),) \\
f_{i-1}(u(i)) &= \eta \sum_{j=1}^{i-1} e(j)k(u(j),u(i)) \\
e(i) &= d(i) - f_{i-1}(u(i)) \\
f_i &= f_{i-1} + \eta e(i)k(u(i),)
\end{align*}
\]

all the above explained equations gives the KLMS method. It's the LMS in RKHS, and filtering is finished by kernel analysis. KLMS allocates a replacement kernel unit for the new training information with input \( u(i) \) because the center and e(i) which is coefficient. These coefficients and also the centers are stored in memory throughout training [8]

4. SPEECH RECOGNITION METHODS

4.1 Dynamic time warping (DTW)

Speech is time-dependent methodology.

Hence, the utterances of constant word can have altogether utterly completely different durations, and utterances of constant word with constant quantity can disagree at intervals the middle, as a results of various components of the words being spoken at different rates. to induce a worldwide distance between two speech patterns (represented as a sequence of vectors) a time alignment have to be compelled to be performed.

Dynamic Time Warping (DTW) that could be a technique that finds the simplest alignment between two statistic if simply the once series may even be “warped” non-linearly by stretching or shrinking it on its time axis. This distortion between two words can then be accustomed notice corresponding regions between the two words or to ascertain the similarity between the two words. Dynamic time warping is typically utilised in speech recognition to ascertain if two waveforms represent constant spoken phrase [10]. In addition to speech recognition, dynamic time warping has together been found useful in many completely different disciplines, in conjunction with data processing, gesture recognition, robotics, producing, and medicine.

This disadvantage is illustrated at intervals in the below figure two, at intervals that a “time-time” matrix is used to visualize the alignment. In general, all the time alignment examples, the reference pattern (template) which may go up the side and also the input pattern may go along all-time low.

Throughout this illustration the input “SsPEEhH” may be a ‘noisy’ version of the model “SPEECH”. The thought is that ‘h’ may be a better match to ‘H’ compared with something at
The input "SsPEEhH" are matched against all templates at intervals the system's repository. The foremost effective matching model is that the one that there's the lowest distance path positioning the input pattern to the model. An easy global distance score for a path is simply the total of native distances that head to create up the path.

We apply certain restriction on the direction of propagation to make the algorithm to reduce excessive computation. The constraints which are given below as follows.

- Matching paths cannot go backwards in time.
- Every frame in the input must be used in a matching path.
- Local distance scores are combined by adding to give a global distance.

This method is thought as Dynamic Programming (DP). Once applied to template-based speech recognition, it's usually spoken as Dynamic Time warping (DTW). DP is certain to find the lowest distance path through the matrix, whereas minimizing the number of computation. The DP method operates in a very time-synchronous manner: every column of the time-time matrix is taken into account in succession (equivalent to process the input frame-by-frame) in order that, for a template of length N, the most variety of ways being considered at any time is N.

Consider if $D(i,j)$ is the global distance up to $(i,j)$ and the local distance at $(i,j)$ is given by $d(i,j)$.

$$D(i, j) = \min[D(i-1, j-1), D(i-1, j), D(i, j-1)] + d(i, j)$$

\text{eq. (26)}

Given that $D(1,1) = d(1,1)$ (which is the initial condition),

We have the idea for a suitable algorithmic rule for computing $D(i,j)$. The ultimate global distance $D(n,N)$ provides overall matching score of the template with the input. The input word is then recognized because the word corresponding the template with all-time low matching score.

4.2 Hidden Markov Model (HMM)

The Hidden Markov Model (HMM) which is a powerful statistical based tool for modeling generative sequences which will be characterised by an underlying method generating an evident
HMMs have found application in several areas fascinated by signal processing, and specifically speech processing, however have additionally been applied successfully to low level NLP (Natural Language processing) tasks like phrase unitization, extracting target info from documents and part-of-speech tagging.

The technique is employed to train a model that is, in our case ought to represent an utterance of a word. This model is employed afterward within the testing of an utterance and conniving the chance of that the model has created the sequence of vectors.

The very important distinction between the Observable Markov Model (OMM) and a Hidden Markov Model (HMM) which is that within the Observable the output state is completely determined at on every occasion instant t. Inside the hidden Markov Model the state at on each occasion t must be inferred from observations. An observation may be a probabilistic function of a state.

The basic Hidden Markov Model is represented by \( \lambda = (\pi, A, B) \).

\( \pi \) = initial state distribution vector.
A = State transition probability matrix.
B = continuous observation probability density function matrix.

The three important fundamental problems in the Hidden Markov Model design which are given as follows:

Problem one – Recognition:
With the given observation sequence \( O = (o_1, o_2, \ldots, o_T) \) and the model \( \lambda = (\pi, A, B) \), how to compute the is the probability of the observation sequence given the model? That is, how the \( P(O|\lambda) \) is computed efficiently?

Problem two - Optimal state sequence:
With the given observation sequence \( O = (o_1, o_2, \ldots, o_T) \) and the model \( \lambda = (\pi, A, B) \), how would be a corresponding state sequence, \( q = (q_1, q_2, \ldots, q_T) \), chosen to be optimal in some sense

Problem three – Adjustment:
How the probability measures will be, \( \lambda = (\pi, A, B) \), adjusted to maximize \( P(O|\lambda) \) with the given a sequence of acoustic vectors \( y = \{y_1, y_2, \ldots, y_T\} \), and for the speech recognition task is to find the optimal words sequence \( W = \{w_1, w_2, \ldots, w_L\} \), in which maximizes the probability \( P(W|y) \), i.e. the probability of the word sequence \( W \) given the acoustic feature vectors \( y \). This probability is normally computed with the help of Bayes Theorem:

\[
P(W|y) = \frac{P(\lambda | W)P(W)}{P(\lambda)}
\]

In the equation 27, probability \( P(W) \) is the word sequence \( W \), which is named language model probability and it is computed using a language model. The probability \( P(y|W) \) is referred to as
acoustic model probability, which is the probability that the acoustic feature vector $\mathbf{y}$ is produced by the word sequence $W$. $P(\mathbf{y}|W)$ is calculated by using acoustic model[14].

### 4.3 Hybrid HMM/DTW Approach

Therefore, we followed a different approach, namely the extension of a state-of-the-art system that achieves extremely good recognition rates with a HMM that is trained. For recognition purpose used the DTW method.

### 5. Experiments And Results

In this section, the performance of the speech recognition of ten words using the hybrid model is explained. The training of spoken words is done with the HMM model and to classify the data, used the DTW method. The noise present in the speech signal is removed by using the KLMS algorithm and the improvement in SNR is 2.1 dB compared to the LMS algorithm. The recognition rate obtained is 94% with the hybrid model compared to 90% using the individual method with noise conditions.

#### Table 1: Comparison of speech recognition rate for isolated words

<table>
<thead>
<tr>
<th>Word</th>
<th>Recognition rate without hybrid model without noise removal</th>
<th>Recognition Rate(For Hybrid HMM/DTW)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Siva</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>Sunny</td>
<td>90</td>
<td>90</td>
</tr>
<tr>
<td>Venkat</td>
<td>90</td>
<td>90</td>
</tr>
<tr>
<td>Santhosh</td>
<td>80</td>
<td>80</td>
</tr>
<tr>
<td>Sanjeev</td>
<td>90</td>
<td>90</td>
</tr>
<tr>
<td>Venu</td>
<td>90</td>
<td>100</td>
</tr>
<tr>
<td>Sai</td>
<td>90</td>
<td>100</td>
</tr>
<tr>
<td>Raju</td>
<td>90</td>
<td>100</td>
</tr>
<tr>
<td>Keerthi</td>
<td>90</td>
<td>100</td>
</tr>
<tr>
<td>Karthik</td>
<td>90</td>
<td>90</td>
</tr>
</tbody>
</table>

The figure 3 gives the simulation results of speech recognition for the given words.
Figure 3: Speech Recognition results using the hybrid methods with kernel adaptive filtering method.

6. CONCLUSIONS AND FUTURE SCOPE

In this work, we used a new approach which is applied for speech recognition using a hybrid HMM/DTW method with kernel adaptive filtering method for speech enhancement. From the experimental results, it is observed that the proposed approach gave better recognition rate for speech recognition than the traditional methods. In future work, Artificial Neural Networks and Support Vector Machines can be used for classification to improve the recognition rate.

REFERENCES