

Speech Recognition of Isolated Words in the “Baoulé” Language Via Back-Propagation Neural Networks (BNN)

Francis Adlès KOUASSI

Ecole Supérieure Africaine des TIC, LASTIC, Abidjan, Côte d’Ivoire

Hyacinthe Kouassi KONAN

Ecole Supérieure Africaine des TIC, LASTIC, Abidjan, Côte d’Ivoire

Mamadou COULIBALY

Ecole Supérieure Africaine des TIC, LASTIC, Abidjan, Côte d’Ivoire

Olivier ASSEU

Ecole Supérieure Africaine des TIC, LASTIC, Abidjan, Côte d’Ivoire

ABSTRACT

The main objective of this research is to explore how a back propagation neural network (BNN) can be applied to the speech recognition of isolated words. The simulation results show that a BNN offers an efficient 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 in this article can be extended to other applications, such as the recognition of sonar targets and the classification of underwater acoustic signals.

Keyword: Speech recognition, back propagation neural network, LPC

INTRODUCTION

The demand for real-time processing in a vast majority of applications (eg, signal processing and weapon systems) has led researchers to seek new approaches to address the bottleneck of conventional serial processing.

Artificial neural networks, based on human-like perceptual characteristics, are one such approach that is currently gaining much attention. Due to significant advances in computing and new technologies, neural networks, once thought to be a dead zone, are resurrecting after a four-decade sleepy period.

The main objective of our research in this article is to explore how neural networks can be used to recognize single word speech as an alternative to traditional methodologies.

The main benefit of this study would be its contribution to the understanding of neural network based techniques to solve the common but difficult problem of pattern recognition, especially in automatic speech recognition (ASR). This technique can be extended to various other important practical applications. The article is organized as follows:

Section 2 presents neural networks. The reasons for the use of neural networks in speech recognition are briefly discussed, and the structure and characteristics of a feedback neural network (BNN) are described.

Section 3 reviews the basic principles of speech recognition and describes techniques for estimating the parameters used as characteristics of the speech signal in the recognition step. Section 4 presents the results of a small vocabulary system.

Section 5 summarizes the important findings, examines the limitations of the proposed system, and offers ideas for future research.

NEURAL NETWORKS

Why neural networks?

Recently, studies have revealed the potential of neural networks as useful tools for various applications requiring deep categorization. Several neural network applications have been shown to be successful or partially successful in the fields of character recognition, image compression, medical diagnostics, and financial and economic forecasting [1].

The power of parallel processing in neural networks and their ability to classify data based on selected characteristics provide promising tools for pattern recognition in general and speech recognition in particular.

Traditional sequential processing technology has serious limitations for the implementation of pattern recognition problems. The classical approach to pattern recognition is often expensive, inflexible, and requires data intensive processing [2].

in other words, neural networks accomplish the processing task by training rather than programming in a manner analogous to how the human brain learns. Unlike traditional Von Neumann sequential machines where the formula and rules must be specified explicitly, a neural network can derive its functionality by learning from the examples presented.

Architecture

Neural networks are made up of a large number of neurons or processing elements, **PE** for short. Each PE has a number of inputs with associated connection weights, as shown in Figure 1. These weighted inputs are summed and then mapped via a nonlinear threshold function. The threshold function, also called the activation function or the transfer function, is continuous, differentiable and increasing monotonically [1]. Two widely used threshold functions are the sigmoid

$$f(z) = \frac{1}{1 - e^{-z}} \quad (2.1)$$

and the hyperbolic tangent

$$f(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}} \quad (2.2)$$

The processing elements are layered in a predefined manner. Each PE processes the inputs it receives through its weighted input connections and provides a continuous value to the other

PEs through its outgoing connection. The basic neuron model shown in Figure 1 can be easily realized in hardware. An equivalent electrical analog model could be constructed using a nonlinear feedback operational amplifier to represent the summation operator and the activation function while the connection weights could be achieved by the input resistors to the operational amplifiers [3].

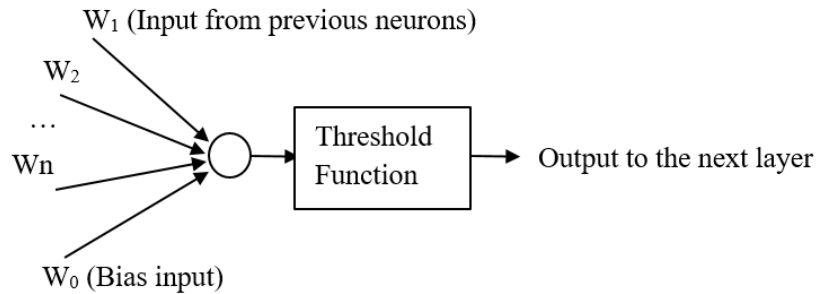


Figure 1. An artificial neuron

Back propagation neural networks

Currently, the most popular and commonly used neural network is the Backpropagation Neural Network (BNN). A typical backpropagation neural network is a multi-layered, direct-feedback network with an input layer, an output layer, and at least one hidden layer. Each layer is fully connected to the next layer, however, there is no interconnection or feedback between PEs in the same layer.

Each PE has a bias entry with an associated non-zero weight. The bias input is analogous to connecting to ground in an electrical circuit in that it provides a constant input value. Figure 2 shows a typical multi-layered backpropagation network with a single hidden layer.

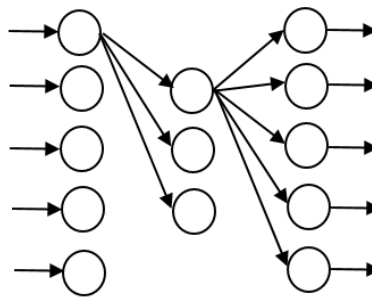


Figure 2. A back propagation neural network

In a neural network composed of N processing elements, the input / output function is defined as [4]:

$$\mathbf{y} = \mathbf{G}(\mathbf{x}, \mathbf{W}) \quad (2.3)$$

where $\mathbf{x} = \{x_j\}$ is the input vector of the network, $\mathbf{y} = \{y_j\}$ is the output vector of the network and $\mathbf{W}$ is the weight matrix. The latter is defined as

$$\mathbf{W} = (\mathbf{w}_1^T, \mathbf{w}_2^T, \dots, \mathbf{w}_N^T)^T \quad (2.4)$$

where the vectors $\mathbf{w}_1^T, \mathbf{w}_2^T, \dots, \mathbf{w}_N^T$ are the individual PE weight vectors, which are given by

$$\mathbf{w}_i^T = (w_{i1}, w_{i2}, \dots, w_{in}) \quad ; \quad i = 1, 2, \dots, N \quad (2.5)$$

These connection weights adapt or change in response to the examples of inputs and outputs desired according to a learning law. Most of the learning laws are formulated to guide the vector of weight $\mathbf{w}$ to a location which gives the desired network performance. There are many learning laws depending on the type of application : **delta rule** or **LMS algorithm**, competitive learning and adaptive resonance theory to name a few [4]. The back propagation neural network implements the delta rule to update the connection weights. The delta rule is a modification of the least squares (LMS) algorithm in which the overall error function E between the current output, y_i , and the desired output, D_i , given by

$$E = \frac{1}{2} \sum_j (d_j - y_j)^2 \quad (2.6)$$

is minimized. The resulting weight update equation can be represented as [Ref. 1]

$$\mathbf{w}_{ijnew}^s = \mathbf{w}_{old}^s + \mu \left(\frac{\partial E}{\partial \mathbf{w}_{ij}} \right) + \nu (\mathbf{w}_{ijnew}^{s-1} - \mathbf{w}_{ijold}^{s-1}) \quad (2.7)$$

where μ is the learning rate, ν is the pulse term and s is the layer number. The learning rate μ is equivalent to the step size parameter in the LMS algorithm. Selecting an appropriate value for μ is usually done by trial and error. Too high a learning rate often results in a divergence of the learning algorithm, while too small a value results in a slow learning process. The value of μ must be between $0 < \mu < 1$ [4]. The pulse term ν has been added in equation (2-7) to act as a low pass filter on the delta weight term; it is used to help "smooth out" changes in weight. The value of ν is generally between zero and one [1]. It has been found that using a momentum term significantly speeds up the learning process. It allows a larger value for μ while avoiding algorithm divergence.

The main mechanism of the backpropagation neural network is to propagate the input through all the layers to the output layer. At the output layer, errors are determined and the associated weights, $\mathbf{w}_{ij}$, are updated by equation (2.7). These errors are then propagated through the network from the output layer to the previous layer (hence the name backpropagation). The error back propagated to PE j in layer s is given by [1]

$$e_j^{[s]} = x_j^{[s]} (1.0 - x_j^{[s]}) \sum_k (e_k^{[s+1]} w_{kj}^{[s+1]}) \quad (2.8)$$

This process continues until the input layer is reached. In summary, artificial neural networks, which use highly parallel architectures with distributed processing among PEs, overcome the limitation of the bottleneck of conventional sequential processing. The backpropagation

algorithm described in this chapter provides a simple mechanism to implement a nonlinear mapping between input and output.

THE CONCEPTS OF SPEECH RECOGNITION

Existing difficulties in isolated-word speech recognition

Although the field of automatic speech recognition has been around for several decades, many ASR difficulties still exist and need to be addressed [5]. Besides syntactic and semantic issues in linguistic theories, segmentation of speech is of great concern. The boundaries between words and phonemes are very difficult to locate, except when the speaker pauses.

There is no separation between words in relation to the written language. Although the limits can be estimated by a sudden and large change in the spectrum of speech, this method is not very reliable due to coarticulation, i.e. changes in articulation and acoustics of a phoneme because of its phonetic content.

The voice signal depends on the context, i.e. the word before and the word after affect how the signal is formed.

The voice signal also depends on the speaking mode. Smoothness, volume, and stuttering also affect the voice signal.

The environment, such as a quiet room, background noise, or interference in the same channel, greatly affects the signal.

Input devices, such as microphone, amplifier and recorder, also modify the voice signal [6]. Due to asymmetries in its production and interpretation, speech is a complex signal for recognition. For effective results, ASR requires an approach closer to human perception, and traditional computer techniques are inefficient for such tasks.

Neural networks, which are modeled on the human brain, appear to provide a useful tool for speech recognition.

Basics of speech recognition

Voice data is often redundant and too long to apply the entire signal to the recognition device. A pre-treatment step is often necessary to achieve a high yield. This step consists in determining the discriminating characteristics specific to the signal [2].

Most speech recognition tools only use those aspects that help in sound discrimination and capture spectral information in a few parameters to identify spoken words or phonemes.

Typical speech signal parameters that are used as characteristics are LPC coefficients, energy, zero crossing rate, and speech / non-speech classification.

Most ASR systems use the same techniques as those used in the traditional field of pattern recognition. The procedure for implementing these systems includes several steps: normalization, parameterization, feature extraction, similarity comparison and decision. Figure 3 shows a block diagram of the general voice recognition procedure [7].

The voice signal is fed into a pre-processor which performs amplitude normalization, parameterization and generates a test model based on these characteristics. The test model is then compared to a set of pre-recorded reference models. The reference model that most closely matches the test model is determined based on predetermined decision rules.

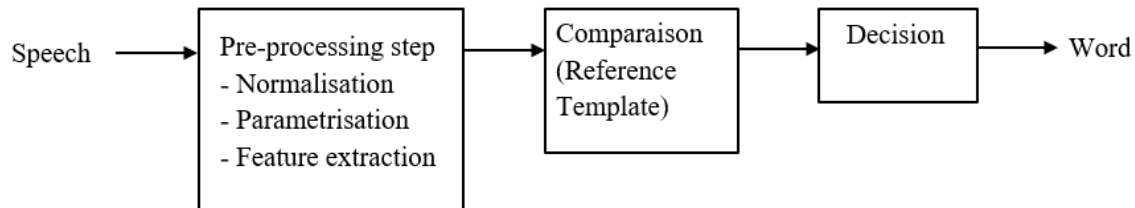


Figure 3. Functional diagram of pattern recognition for ASR

Standardization

The amplitude normalization step attempts to eliminate variability in the input speech signal due to the environment - variations in recording level, distance from microphone, original speech intensity and signal loss in transmission.

Parameterization and feature extraction

The parameterization step consists in converting the voice signal into a set of statistical parameters which maximize the probability of choosing the right output characteristics corresponding to the input signal. Since most useful acoustic speech details are below 4KHz, a bandwidth of 3-4KHz is sufficient to represent the speech signal.

Experience has shown that using additional bandwidth actually degrades ASR performance [5]. In general, eight to fourteen LPC coefficients are considered sufficient to model the spectral behavior of the spectral envelope of short time speech [8]. Other parameterization techniques, such as cepstral analysis, can also be used.

The test pattern is formed by an arrangement of features, such as LPC or Cepstral coefficients, short term energy, short term zero crossing rate, and V / U switch.

Comparison and similarity decision

Even for a single speaker, the same utterances are spoken with different durations and at different speaking rates when repeated; therefore, a normalization of the reference utterance along the time axis is necessary before the comparison step of Figure 3 [5]. Linear temporal distortion, which is achieved by adjusting the frame interval before parameterization or by decimating / interpolating the feature sequence, can be used to partially overcome this problem.

The linear temporal distortion is easy to implement; however, it ignores the non-linearity of the change in speech rate. For example, the energy of long and short versions of the same word is not distributed evenly for each phoneme but is often concentrated on a dominant vowel. Other approaches add silence to the ends of shorter utterances, using the length of the longer utterance as a reference.

The test model is then compared with the reference models based on Euclidean distance or LPC distance measurement. The word corresponding to the reference template which gives the minimum distance is announced as the spoken word. When ASR is performed by neural networks, the comparison and decision steps are performed through learning. The neural network is trained by presenting examples of the reference models. A learning algorithm based on the delta rule described in chapter two is used to minimize an overall error function between the current output and the desired output.

Once the error or delta weight term ($w_{ijnew} - w_{ijold}$) is less than a predefined threshold, a final set of weight values is obtained and used as network characteristics. The unknown signal (or test) is then passed through this network with fixed weight values; the word corresponding to the highest value of the output vector is the recognized word. The recognition rate is then calculated on the basis of the percentage of correctly identified exits.

SOFTWARE SIMULATIONS STUDIES AND RESULTS

Simulation

In this section, the back propagation neural network is applied to the speaker independent voice data.

Data Corpus

The voice data set used in this article, both for training and testing, contains a total of 6,300 sentences spoken by 630 speakers from 8 different main geographies.

The sentences are phonetically diverse to provide good coverage of phonetic utterances and to add diversity to phonetic contexts.

Each sentence is broken down into four different representations: waveform, text, word and phoneme.

The waveform is the voice data in the original digitized format. Text is the spelling transcription of words. Word and phoneme representations use the transcription of time-aligned words and phonemes, respectively. The boundaries were aligned with the phonetic segments using a dynamic string alignment program.

Speaker independent system

In this research, three speech recognition devices, consisting of a vocabulary of 3 words, 5 words and 10 words, were developed using a back-propagating neural network. These recognition tools were trained using voice recordings of various speakers, resulting in a speaker independent system.

Figure 4 shows the block diagram of a typical 10 word recognition device. The voice signal is passed through an endpoint detection block to isolate the words. The input vector is then formed by calculating the necessary characteristics and fed into the back propagation neural network. The output vector is first normalized and then compared to the desired output vector, e.g. (1,0,0,0,0,0,0,0,0,0) for the first word, (0,1, 0, 0,0,0,0,0,0,0) for the second word, and so on. The recognition rate is calculated based on the percentage of correctly matched outputs.

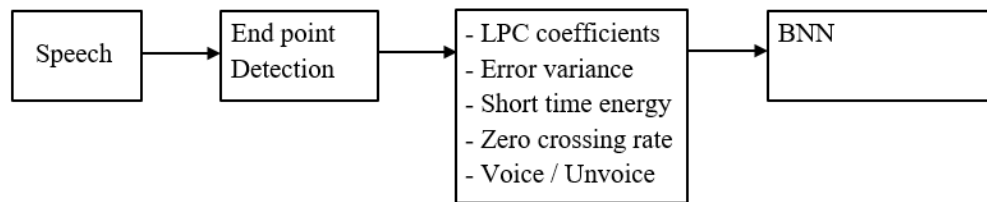


Figure 4. A typical 10 word recognition tool

All pattern recognition tasks, including ASR, use two phases:

- Learning and
- Testing or recognition.

While there are general guidelines for the size of training and testing data, particular issues are often limited by the data available. Larger datasets produce more reliable results with lower variances in performance estimates (error rates).

The training set used for the 10-word recognition module includes 100 words spoken by 10 male speakers.

There are two sets of tests:

- one consists of 100 words spoken by 10 speakers and
- The second comprises 380 words spoken by 38 different speakers.

For the 5 and 3 word recognition module, the number of speakers remains the same, but the total number of words for the training data is reduced to 50 and 30, respectively. The test data consists of 190 words (5x38) for the 5-word recognition module and 114 words (3x38) for the 3-word recognition module.

Preprocessing and feature extraction

The original speech signals from the database are digitized at 16 KHz. Since most of the energy for voiced sounds is concentrated below 4 KHz [3], the speech signals are low pass filtered to a bandwidth of 4 KHz and decimated to remove the redundancy. The input vector to the back-propagation neural network consists of linear predictive coding coefficients, a_n , error variance, E , short time energy function, E , short time average zero crossing rate, Z_n , and voiced/unvoiced discrimination, V/U .

Speech is a non-stationary and time-variant signal. To cope with the temporal variability, short-time processing techniques are used. Short segments of the speech signal are isolated and processed as if they were segments from a signal that is statistically stationary.

There are two methods of segmenting the speech signal. The first one segments the speech signal into fixed length frames. The second divides the speech signal into a fixed number of frames with variable length. The latter was used in this work to deal with the non-uniform word length in the speech recordings. The digitized speech signal is segmented into 10 frames of 10 to 40 milliseconds each depending on the length of a given word in the vocabulary. Each frame is weighted by a hamming window. A brief summary of the processing steps is given in Figure 5 [9].

The digitized speech signal sampled at 16 KHz is first pre-processed. It is passed through a lowpass filter and decimated to 8 KHz; the signal is then segmented into 10 variable length frames. The feature vector is obtained by computing the LPC coefficients, the error variance, the short time energy, the zero crossing rate and the voiced/unvoiced classification for each segment. The Matlab programs for processing these parameters in the feature vector are listed in Appendix A.

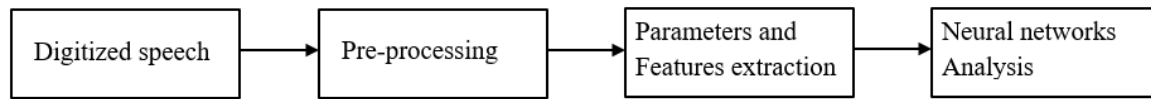


Figure 5. Block diagram of pretreatment.

LPC coefficients and errorvariance

Linear predictive analysis is a technique for estimating the basic speech parameters, e.g., formants, spectrum, and vocal tract area function [3]. A unique set of parameters is determined by minimizing the sum of the squared error between the actual and the predicted speech signal. The LPC parameters provide a good representation of the gross features of the short time spectrum. The poles of the transfer function indicate the major energy concentration in the spectrum, and the error variance indicates the energy level of the excitation signal [5]. The algorithm used to compute the LPC coefficients is the covariance method which requires solving a set of leastsquare based normal equations; Matlab Code for this is included in Appendix A.

Short-timeenergy function

The short-time energy function is used to reflect the amplitude variation between voiced and unvoiced speech segments. This quantity is defined as

$$E_n = \sum_{m=-\infty}^{+\infty} x^2(m)h(n-m) \quad (4.1)$$

where $h(n) = w^2(n)$, and $w(n)$ is the Hamming window (a length of 50 is used in this experiment).

Short-time averagezero-crossingrate

The short-time average zero-crossing rate is a simple measure of frequency content of the speech signal. This quantity is defined as

$$Z_n = \sum_{m=-\infty}^{+\infty} |sgn[x(m)] - sgn[x(m-1)]|w(n-m) \quad (4.2)$$

where

$$sgn[x(n)] = \begin{cases} 1 & x(n) \geq 0 \\ -1 & x(n) < 0 \end{cases} \quad (4.3)$$

and

$$w(n) = \begin{cases} \frac{1}{2N} & 0 \leq n \leq N - 1 \\ 0 & \text{otherwise} \end{cases} \quad (4.4)$$

Voiced / unvoiced discrimination

There is a strong correlation between the energy distribution and the voiced/unvoiced classification of speech. The speech signal corresponding to each word is divided into 10 smaller segments for classification as voiced or unvoiced speech. The differentiation between voiced and unvoiced segments is accomplished by setting a threshold based on the short time energy function. For energy values higher than 0.8 of the maximum level, the segment is treated as voiced and assigned a value of 1. For values of energy lower than 0.2 of the maximum level, the segment is classified as unvoiced and assigned a value of 0. Any mid-level is considered intermediate and assigned a value of 0.5.

Results and analysis

For consistency and ease of comparison for different cases, BNN parameters are kept constant in all experiments reported here. The learning algorithm used is the delta rule. The transfer function is hyperbolic tangent. The values of momentum and learning rate are 0.2 and 0.5, respectively. The training cycles used are 5000 for 3- and 5-word networks, and 10000 for the 10-word network.

The feature vector

The performance of a network varies drastically depending on the features used in the training and testing of the network. Too many features may decrease the efficiency of the network since it takes too long to train the network, while too few features may degrade the network's performance. Two experiments are conducted to emphasize the importance of the choice of the features used in the preprocessing step. The first experiment only uses the LPC coefficients and the error variance. The second one uses all features: LPC coefficients, error variance, short time energy, short time average zero crossing rate, and voiced/unvoiced discrimination.

In the first experiment, a 4th order LPC analysis is performed; for each frame, a set of four LPC coefficients is generated along with the error variance. Thus the input vector to the recognizer consists of a total of fifty parameters per word (five parameters per frame and ten frames per words).

For the second experiment, the short-time energy, the short-time average zerocrossing rate, and the voiced/unvoiced switch are also computed for each frame, making a total of eighty parameters per word. The results are summarized in Table 1. For a small vocabulary set, the recognition rates do not change much between the two experiments. With the 38-speaker testing set, the recognition rate remains 100% for the 3-word recognizer in both experiments. For the 5-word recognizer the rate decreases from 94.7% to 91.6%. For the 10-word recognizer, the result not only deteriorates from 91.3% to 63.7%, but also the training time increases drastically from 10000 to 80000 training cycles.

Table 1. Effect of choice of features on recognition rate

Training vector	Vocabulary size	Error	Recognition rate
50x30	3 words	0	100%
50x50	5 words	16/190	91.6%
50x100	10 words	138/380	63.7%
80x30	3 words	0	100%
80x50	5 words	10/190	94.7%
80x100	10 words	36/380	91.3%

B. LPC order

The order of the LPC coefficients plays an important role in the performance of the network. Four experiments with different LPC orders were conducted on a 10-word vocabulary back propagation network with 10 speaker testing set. A full feature vector is used for all of the following experiments.

Table 2. Effect of lpc order on recognition rate

LPC order	Recognition rate
2	84%
4	94%
8	92%
12	86%

The recognition rates shown in table 2 indicate that the fourth order LPC system captures the essential features of the speech data. The second order system apparently does not have enough parameters to differentiate between the words. However, as the LPC order increases, the complexity of the system also increases, and the information becomes redundant, making the network more cumbersome for training and recognition. This can actually result in decrease recognition rate, as shown.

Vocabulary size

The size of vocabulary used in the training and testing set also affects the recognition rate of the network. As the number of words used in the network increases, the performance decreases rapidly. The results are summarized in Table 3. With a 3-word vocabulary and a 12th order system, the recognition rate is 96.5% for the 380 word testing set. This rate decreases to 93.2% for a 5-word system and deteriorates to 86.6% for a 10-word system.

Table 3: effect of vocabulary size and lpc order on performance of BNN

Vocabulary size	3 words		5 words		10 words	
Testing set size	100wrds	380wrds	100wrds	380wrds	100wrds	380wrds
2nd order LPC	100%	100%	96%	94.2%	84%	87%
4th order LPC	100%	100%	100%	94.7%	94%	91.3%
8th order LPC	100%	99.2%	96%	94.7%	92%	89%
12th order LPC	100%	96.5%	98%	93.2%	86%	86.6%

The number of PEs in hidden layers and the number of hidden layers

The structure of the network also affects the performance of the network. A back-propagation neural network generally has one to two hidden layers. The performance of the network

increases as the number of PEs increases to an optimal number and then starts to decrease. Here the optimal number of PEs in the hidden layer is determined experimentally; a hidden layer with 24 PEs is found to perform well for the speech data. The single hidden-layer network is found to give better results than the network with two hidden-layers. The results are summarized in tables 4 and 5.

Table 4. Effect of hidden layer on the recognition rate

	Testing set size	
	100 wrds	380 wrds
one hidden layer 80-24-10	94%	91.3%
Two hidden layers 80-24-12-10	90%	87.4%

Table 5. Effect on the number of pes in the single hidden-layer network.

PE in hidden layer	Testing set size		Training cycles required
	100 wrds	380 wrds	
12	91%	90%	10000
24	94%	91.3%	10000
32	89%	88.4%	15000
64	89%	88%	25000

Learning rate and Momentum

For a learning rate of 0.5, the effect of the momentum term was experimentally studied and the results are shown in Table 6. A momentum of 0.1 was found to give the best recognition rate and fast training.

Table 6. Effect of momentum on recognition rate

Momentum	Testing set of		Training cycles required
	100 wrds	380 wrds	
0.01	94%	90%	5000
0.1	98%	92.1%	5000
0.2	95%	90.5%	10000
0.3	92%	88.4%	15000
0.4	77%	75.8%	30000

Effect of embedded noise

One special characteristic of the neural network is its non-susceptibility to noise. Its ability to classify in a noisy environment, which is demonstrated here, makes it suitable for ASR. Noise added to the speech signal only affects the performance of the network slightly. Uniform random noise may be added to the weighted sum input signal prior applying the weight matrix. The source and the amount of noise added is determined by the mode (learning or testing) and the appropriate parameter in the "Temperature" row from the "learning schedule" [1]. Table 7 shows the results of a 10-word vocabulary using a 4th order LPC system with 20% added noise.

Table 7. Effect of embedded noise

	Testing set of	
	100 wrds	380 wrds
No noise	94%	90.5%
With 20% added noise	91%	89.2%

In summary, the choice of the feature vector in the design of a BNN is very important. The discriminating characteristics that are particular to the speech signal must be recognized to achieve good results. Also, the information fed to the network should be adequate and optimal, i.e., too little information will lead to a low recognition rate, and too much information will result in extensive training time.

CONCLUSION

A back-propagation neural network, combined with speech signal processing techniques, is used to develop a speech recognition system. Specifically, a BNN was used to design a 10-word speech recognizer. Experiments were conducted on the recognizer. The main observations from these experiments are summarized below:

- BNN is an effective approach for small vocabulary ASR. The recognition rate is 100% in most cases for the 3 and 5-word vocabulary systems, and 94% for the 10-word system.
- 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, which consisted of the LPC coefficients, short time energy, zero-crossing rate and voiced/unvoiced classification, worked well and provided good results for the systems studied.
- The techniques developed in this research on isolated-word speech recognition can be extended to other important practical applications, such as sonar target recognition, missile seeking and tracking functions in modern weapon systems, and classification of underwater acoustic signals. However we cannot make predictions about the likely performance of the methods in these areas until they are actually tested.

In this research on ASR, all experiments used male speakers from one specific geographical region. A larger, more diverse group of speakers should be used for a more general case. In addition, the 10-word recognizer system is small for most real applications. Future research should be directed toward larger vocabulary systems, involving say 50 words or more.

The emphasis of the research was to develop an isolated-word speech recognizer using a BNN. The techniques and schemes used in training and testing the network to improve the recognition rate, as well as to keep the system stable and the convergence rate high, worked well in the experiments reported in the thesis. However, this set of techniques may be only applied to this particular case. There are still no firm rules available to train the networks. Most of the rules of thumb are experiment dependent. More widely applicable learning and testing schemes are needed to further improve the recognition rates as well as increase the vocabulary size.

ANNEXES

Appendix a. Simulation programs

Covarc.m Covariance method to solve the leastsquare normal equation $(X'X)a=(S,O)'$

Usage: [A,S]=covarc(x,p,N); signal x of length N and order p

```
function [A,S]=covarc(x,p,N);
    for i = 1:(N-p)
        for j = 1:(p+1)
            xp(i,j) = x(p+1+i-j);
        end
    end
    Rx = xp'*xp;
    [m,n] = size(Rx);
    Rxp = [Rx(2:m,2:n)];
    A = [inv(Rxp)*(-Rx(2:m,1))];
    S = Rx(1:n)*[1;A];
    S = S/N;
```

stef.m Short time energy function

Usage : y=stef(x);

x is input speech vector and y is corresponding
y corresponding short Time Energy function En.

```
function xos=stef(xi)
    [m,n]=size(xi);
    if n~=1,
        xi=xi'; % make sure input is a row vector
    end;
    N= input (' enter window size (50 to 300 samples) N? ');
    % N=50 is used;
    wt=input (' choose window type 0. Rectangular 1. Hamming? ');
    % wtf; hamming window is used
    if wt==0
        window=ones(1,N);advance=fix(N/2);wtstrng=' Rect. Win';
    else
        window=harnming (N);advance=fix(N/4);wtstrng=' Harem. Win';
    end
    %shortseq = 0;
    if length(xi)<N % case if length of input<length of window
        xi=[xi,zeros(1,N-length(xi))]; % zero pad
    end
    input=xi.^2;
    imp.response=window.A2;
    xo(l)=sum(input(1 :N).*imp response);
    for n=advance:advance:length(xi)-N,
        xo(l+n/advance)=sum(input(n:n+N-1).*imp-response);
    end;
```

```
xos=sum(xo);
subplot(21 1)
clg; plot(xo);
xlabel('time');title(['N=',num2str(N),wtstmg]);
ylabel('En');grid;
```

```
function xos=stazcr(xi)
stzcr.m Short time average zero crossing rate
usage : y=stazcr(x)
x : input speech vector
y : Short-Time Avarage Zero-Crossing Rate Zn
```

```
function xos=stazcr(xi)
[m,n]=size(xi);
if n=-1,
    xi=xi'; % make sure input is a row vector
end;
%N= input (' enter window size (50 to 300 samples) N? ');
N=50; % window length N=50
%wt--input (' choose window type 0. Rectangular 1. Hamming? ');
wt=1; % hamming window is used
if wt==0
    window=ones( 1,N);advance=fix(Nt2);wtstrng=' Rect. Win';
else
    window=haniming (N)';advance=tix(N/4);wtstrng=' Hamnm. Win';
end
if length(xi)'zN %case if length of input'clength of window
    xi=[xi,zeros(1,N-length(xi))]; %zero pad sequence
end
input=abs( sign( [xi(2:length(xi)),0] )-sign(xi))
xo(1)=sum(input(1:N).*window);
for n=advance:advance:length(xi)-N,
    xo( 1+n/advance)=sum(input(n: n+N- 1).*window);
end;
xos=sum(xo);
subplot(21 1)
cig; plot(xo);
xlabe1( 'time '); title(['N=',num2str(N),wtstrng]);
ylabel('En');grid;
```

```
voiceunv.m Voiced and unvoiced discrimination
usage: vuv=voiceunv(data)
function will return 0 if unvoiced
1 if voiced
0.5 if in between
```

```

function vuv=voiceunv(data)
    NSeg=10;
    NSample=fix(length(data)/NSeg);
    for indx=1:NSeg,
        xx = data((NSample/2)*(indx-1)+ 1:(NSample/2)*(indx));
        en(indx) = stef(xx);
    end;
    vuv=sum(en(1:NSeg));

```

arl2lin.m 12th order AR model coefficients (normalized)

Usage: arcoeff=arl2lin(data)

```

function arcoeff=arl2lin(data)
    p=12; % order of AR coefficients
    Nd =2;
    x=decimate(data, Nd);
    NSeg= 10;
    NSample=fix(length(x)/NSeg);
    for indxl=1:NSeg,
        xx = x(NSample*(indxl-1)+ 1:NSample*indxl);
        xx = dtrend(xx);
        xstore(:,indxl) = xx(:) ./ max(abs(xx));
        [a,s]=covarcovr(xx,p,length(xx));
        aa(:,indxl) = a(1:p)
        astore(:,indxl) = a(1:p) ./ max(abs(a));
        ss(indxl) = s;
        en(indxl) = stef(xx);
        zc(indxl)=stazcr(xx);
        vuv(indxl)=voiceunv(xx);
    end;
    vuv =vuv/max(vuv);
    ss=ss/max(abs(ss));
    en=en/max(abs(en));
    zc=zc/max(abs(zc));
    for indx3=1:NSeg
        if vuv(indx3)<.2
            vuv(indx3)=0;
        elseif vuv(indx3)<.8
            vuv(indx3)=.5;
        else
            vuv(indx3)= 1;
        end
    end
    arcol=aa(:);
    scalef=1/max(abs(arcol));
    arcoef=aa/scalef;
    arcoeff=[arcoef;ss;en;zc;vuv];
    arcoefff=arcoeff(:);

```

Références

NeuralWare, Inc., Neural Computing, documentation for the Neural Professional II Plus Neural Network simulation Software, 1991.

Gorman, R.P. and Sejnowski, T.J. "Learned Classification of Sonar Targets Using a Massively Parallel Network," *IEEE Trans. Acous., Speech, and Sig. Processing*, Vol. 36, No. 7, July 1998.

Webster, Willard P., "Artificial Neural Networks and their Application to Weapons," *Naval Engineering Journal*, v. 103, pp. 46-59, May 1991.

Hetcht-Nielsen, R., *Neurocomputing*, Addison-Wesley, 1990.

Rabiner, L.R. and Schafer, R.W., *Digital Processing of Speech Signals*, Prentice Hall, 1978.

Mariani, J., "Recent Advances in Speech Processing," *IEEE Int. Conf. Acous., Speech, and Signal Processing*, Vol. 1, pp 429-440, 1989.

O'Shaughnessy, D., *Speech Communication*, Addison-Wesley Publishing Company, 1987.

NTIS, TIMIT, CD ROM on Line Documentation for the DARPA TIMIT Acoustic Phonetic Continuous Speech Corpus, American Helix, October 1990.

Tom, Daniel M. and Tenorio, Fernando M., "Short Utterance Recognition Using a Network with Minimum Training," *Neural Networks*, Vol. 4, No. 6, pp 711-722, 1991.