SPEECH RECOGNITION USING SUB-WORD NEURAL TREE NETWORK MODELS AND MULTIPLE CLASSIFIER FUSION[1]

Manish Sharma and Richard Mammone

CAIP Center, Rutgers University, P O Box 1390, Piscataway, NJ 08855-1390
msharma@caip.rutgers.edu

ABSTRACT

A new neural tree network (NTN) -based speech recognition system is presented. NTN is a hierarchial classifier that combines the properties of decision trees and feed-forward neural networks. In the sub-word unit-based system, the NTNs model the sub-word speech segments, while the Viterbi algorithm is used for temporal alignment. Durational probability is associated with each sub-word NTN. An iterative algorithm is proposed for training the sub-word NTNs. The sub-word NTN models, as well as the sub-word segment boundaries within a vocabulary word, are re-estimated. Thus, the proposed system is a homogeneous neural network -based, sub-word unit-based, speech recognition system. Furthermore, embedded within this word model paradigm, multiple NTNs are trained for each sub-word segment and their output decisions are combined or fused to yield improved performance. The proposed discriminatory training-based system did not perform favourably as compared to a Hidden Markov model-based system. The paradigm presented in this paper can be argued to represent a class of discriminatory training-based, homogeneous (versus hybrid), sub-word unit-based, speech recognition systems. Hence, the results reported here can be generalized to other similar systems.

1. INTRODUCTION

Aritificial Neural networks (ANNs) have been shown to be very successful in classification of static speech patterns. However, in general, they lack the capability of dealing with dynamic pattern recognition tasks, such as speech recognition. Hidden markov models (HMMs), on the other hand, are stochastic signal generative model and have been found to be very successful for temporal pattern recognition problems. Nonetheless, HMMs also suffer from some inherent limitations which opens up opportunities for alternate paradigms. Hidden Markov modeling assumes an underlying probability density function (generally Gaussian) of the observation vectors. Neural Networks (NNs), on the other hand, can model arbitary probability distributions of the observation vectors. Another limitation of HMMs is that the exponential duration density is characteristic of each state within a Markov chain. If an attempt is made to model sub-word speech units (such as, phonemes) with each state of a HMM, then this limitation may be notable. Generally, the individual HMMs are trained separately by maximum likelihood (ML) estimation. Although, recently discriminant training algorithms have been proposed. Discriminant training can be very useful when the training data is limited. The training algorithms of neural networks are inherently discriminative. Therefore, if the HMM states are replaced by neural networks, the data vectors within each state are being discriminantly trained against the data vectors from other states. Motivated by the above observations, several hybrid Neural Network (NN) -Hidden Markov model (HMM) systems have been recently proposed (for eg., [1, 2, 3]). In a *"embedded"*-type hybrid NN-HMM system [1], the probabilitic output of the NN is used as an emission probability for the underlying HMM; whereas, in a *"post-processor"*-type hybrid NN-HMM system [2, 3], the NN is used to re-score the class hypotheses output by the underlying HMM.

The homogeneous, sub-word neural tree network (NTN) -based word model paradigm presented in this paper has several distinct differences with the *"embedded"*-type hybrid NN-HMM systems. During the iterative training of the word models, the sub-word segment boundaries are also re-estimated. This is in contrast with most other hybrid NN-HMM systems, which either use a database with hand-labelled phonetic segments, or the estimates of the segment boundaries derived from a previously trained HMM system. Secondly, instead of using the transitional probability of the underlying HMM, a durational probability function is associated with each sub-word NTN.

A multiple classifier system can be a powerful solution for robust pattern classification, because it allows for simultaneous use of arbitary feature descriptors and classification procedures [4, 5, 6]. For a given problem, each of the classifiers could attain various degrees of success, but none of them will give perfect performance. Ideally, a combination function can be designed which would take advantage of the strengths of the individual classifiers, avoid their weaknesses and improve classification accuracy. Designing multiple codebooks in a discrete HMM may be considered as an attempt in this direction. In this paper, we present experiments wherein each sub-word segment is modeled using multiple neural tree networks. Decision outputs of these multiple NTNs can be combined properly to yield improved recognition performance.

[1]This research was supported through a grant (F30602-91-C-0120) from the U.S. Air Force at Rome Laboratories.

0-7803-2431-5/95 $4.00 © 1995 IEEE

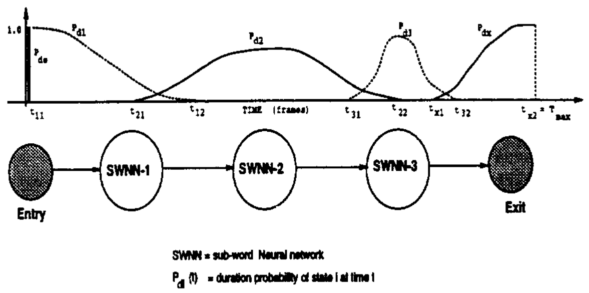

Figure 1: Typical word model with NNs modeling the sub-word segments. The durational probability functions associated with these sub-word segments are also shown.

2. NEURAL TREE NETWORK (NTN)

A neural tree network [7] is a hierarchial classifier that combines the properties of decision trees and feed-forward neural networks. The NTN uses a tree-structure of discriminating elements, with the discriminants being implemented using artificial neurons. The neuron uses all feature elements for the decision, hence the NTN is not constrained to perpendicular discriminant boundaries as are the standard decision trees. The architecture of the NTN is determined during training, thus it is self-organizing. This is in contrast to multi-layer perceptron neural networks, where the architecture must be specified prior to training. NTNs also offer an attractive tradeoff of classification speed versus hardware implementation as compared to MLPs [7]. NTNs have shown to perform favourably to other discriminatory training-based classifiers (including, neural networks and decision trees) in several pattern recognition tasks [7, 8].

2.1. Posterior probability estimation in NTN

A modified neural tree network (MNTN) was introduced in [8]. Forward pruning technique was used while growing the tree. Since, the leaves of the pruned tree are likely to have data from several classes, a *"confidence"* measure is associated with each class. This *"confidence"* measure is equivalent to estimating the posterior probability using Parzens-window method in the region defined by a leaf. In more recently proposed Continuous density NTN (CDNTN) [9], local parametric models (generally, mixture of gaussian) are created for each class at every leaf of a NTN.
Both MNTN and CDNTN have been used in the experiments presented here.

3. WORD MODELING USING SUB-WORD NEURAL TREE NETWORK

In the proposed system each vocabulary word is modeled as a sequence of sub-word neural tree networks (SWNTNs). A typical word model is shown in figure 1. A duration probability function is associated with each sub-word neural network. The number of sub-word neural networks in a word model is generally set to be equal to the number of phonemes present in the *phonetic spelling* of the word.

```
initialization:
    set max-iteration, word "confusability" threshold and
      classification error (CE) threshold for convergence
    for all words, Model-freeze = false
while iteration < max-iteration
    for all words
      if not Model-freeze
        Make word train-file
              (using corresponding "confusable words")
         Train sub-word neural networks
         Compute sub-word duration probabilities
         Resegment sub-word boundaries
              (forced alignment using Viterbi)
      endif
    endfor
    for all words
      Find classfication error (CE)
       if CE < threshold
         Save sub-word segmentation boundaries
         Model-freeze
       else
         Find "confusable" words from the above step
       endif
    endfor
endwhile
```

Figure 2: Iterative training algorithm

3.1. Word model Scoring

Consider the task of scoring the model λ_v, given a sequence of observation vectors $\mathbf{O} = \{\mathbf{x}_1, \cdots, \mathbf{x}_t, \cdots, \mathbf{x}_T\}$. Assume that the output of the sub-word neural networks are normalized so as to yield probabilitic values [8, 9]. The probabilistic output b_{vi} of the SWNTN q_{vi} can then be interpreted as the *a posterior* probability $P(\mathbf{x}_t/q_{vi}, \lambda_v)$ of the observation vector $\mathbf{x}_t$, at time t, given the model λ_v and SWNTN q_{vi}. The durational probability d_{vi} can be argued to represent the probability $P(q_{vi}/\lambda_v)$ of being in state q_{vi}, at time t. The observation probability $P(\mathbf{x}_t/\lambda_v)$ of vector $\mathbf{x}_t$ can then be written as:

$$P(\mathbf{x}_t/\lambda_v) = \sum_i P(\mathbf{x}_t/q_{vi}, \lambda_v)P(q_{vi}/\lambda_v) = \sum_i (b_{vi}d_{vi}) \tag{1}$$

The above equation can be approximated by considering only the maximum value within the summation,

$$P(\mathbf{x}_t/\lambda_v) = \max_i(b_{vi}d_{vi}) \tag{2}$$

If it can be assumed that the successive observations are independent, then the joint probability of the sequence of events can be written as the product of the probability of the individual events.

$$P(\mathbf{O}/\lambda_v) = P(\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_T/\lambda_v) = \prod_{j=1}^{T} P(\mathbf{x}_j/\lambda_v) \tag{3}$$

Equation 3 gives the probability score of an observation sequence O, given the word model λ_v. Equations 2 and 3 can be easily implemented using dynamic programming techniques (Viterbi algorithm).

3.2. Word model Training

The training of each word model is accomplished by using a new iterative algorithm and consists of two steps: segmentation and re-estimation. The segmentation is carried out by forced alignment using the standard Viterbi algorithm. Once the segmentation is produced, the weights of the neural networks associated with each segment can be trained. Given the sub-word segmentation at any iteration, the durational probabilities can also be generated. These steps are executed iteratively until convergence is reached. At the end of each iteration, the classification capabilities of the current model of each word is computed. The information obtained is used in two ways. The classification error (CE) can be used as a convergence criterion. If the CE drops below a pre-determined threshold, the model parameters of that word are *"frozen"*. The other advantage is that the other vocabulary word models which are found to be *"confusable"* with a word, can be used for the discriminant training of the model. The pseudo-code of the training algorithm is illustrated in figure 2.

4. MULTIPLE CLASSIFIER FUSION

In several pattern recognition tasks, a proper combination of various complementary classifiers have shown to improve the performance over individual classifiers [4, 5, 6]. Use of multiple classifier allows for simultaneous use of arbitary feature descriptors and classification procedures. The various feature descriptors may be derived from different analysis procedures, and hence may be in different forms (for example, continuous valued, or binary valued, or the range of their values may differ notably). In such a scenario, it appears sensible to train a classifier for each feature set and combine their outputs for the final decision. Also, for a given problem, each of the various classification procedures could attain various degrees of success, but none of them will give perfect performance. Ideally, a combination function can be designed which would take advantage of the strengths of the individual classifiers, avoid their weaknesses and improve classification accuracy. Even when the feature set used for pattern description and the classification procedure used for recognition, is same in an ensemble of classifiers, the concepts of multiple classifier fusion can still be employed. The diversity in such an ensemble of classifiers can be planned through various means, for example by training each of them on different subsets of the given training data set, or different discriminant data sets (for discriminative training-based classifiers) .

4.1. Decision combination in multiple classifiers

Several methods exist to combine the output of multiple classifiers. The output information of a classification procedure can be divided into three levels, and the decision combination can be done at any of these levels.

1. Class-label level : decision combination rules include, majority voting and logical AND operation (see for eg., [4]).
2. Class-rank level : the objective of the decision combination rule is to derive a consensus ranking, such that the true class is ranked as close to the top as possible. Two commonly used approaches are called class set reduction and class set reordering (see for eg., [5]).
3. Class-score level : Most techniques of decision combination at this level fall under two main categories: Linear opinion pool and Logarithmic opinion pool (see for eg., [10]).
   LINEAR OPINION POOL: can be mathematically formulated as :
   $$P(x) = \sum_{k=1}^{K} \alpha_k P_k(x) \qquad (4)$$
   where, generally, the weights α_k are chosen to be non-negative and to satisfy the constraint $\sum_k \alpha_k = 1$. The weight α_k reflects, in some manner, the relative expertise of the kth classifier. There is no optimal method to choose the weights. However, there are a few suggested approaches, including equal weights ($\alpha_k = \frac{1}{k}$) and weights proportional to classifier ranking ($\alpha_k = \frac{r}{\sum_{r=1}^{K} r}$).
   LOGARITHMIC OPINION POOL: can be mathematically stated as:
   $$P(x) = \prod_{k=1}^{k=K} P_k(x)^{\alpha_k} \qquad (5)$$
   where α_k's are the weights and are often selected to satisfy the unit-sum contraint. The logarithmic opinion pool assumes to an extent that the individual classifiers in the ensemble of K classifiers perform independent of each other (which may be the case, if they are trained on independent feature sets, or independent training sets). Zeros in the logarithmic opinion pool are vetoes, i.e., to say that if any expert assigns $P_k(x) = 0$ the combined decision is also zero. Similar to linear opinion pool, the weights α_k's reflect in some way the relative expertise of the corresponding classifier, and therefore similar weight selection methods could be used. However, the weight selection for the logarithmic opinion pool is less intuitive because of the product form of the rule.

Multiple neural tree networks can be trained to model each sub-word segment in the word model paradigm outlined in section 3. The diversity within the neural network ensemble can be obtained by training them on different discriminant data sets.

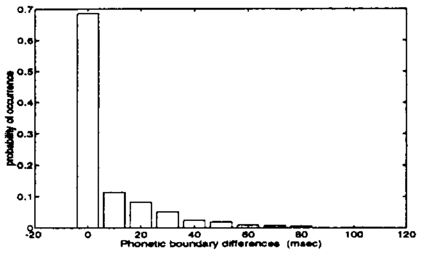

Figure 2: Histogram showing the consistency between the obtained and the supplied phonetic boundaries in TIMIT

5. EXPERIMENTS

Isolated word recognition experiments were conducted to: (1) check the precision of the phonetic segmentation that was obtained at the end of the iterative training procedure, (2) evaluate the recognition performance accuracy of the proposed word model paradigm, and, (3) investigate the multiple classifier concept. For the results reported here, the spectral features used were FFT-derived, raised sine window bandpass liftered, 12^{th}-order cepstral coefficients. The number of sub-word NTNs in a word model were set to be equal to the phonemes present in that word.

Experiments on TIMIT : The male speakers from the north midland dialect region were chosen for the experiments. With the help of the given time-aligned word transcriptions, five words, namely, *dark*, *suit*, *greasy*, *water* and *year*, were extracted from the "SA1"-type sentence of these speakers. Speaker-independent, isolated word recognition experiments were then performed on these extracted words. Since the time-aligned phonetic transcriptions are also supplied alongwith the database, the phonetic segmentation obtained at the end of the iterative training procedure (outlined in figure 2) was compared against it. The histogram in figure 4.1 shows the consistency between the phonetic boundaries obtained by the algorithm and that given alongwith TIMIT database. The histogram shows that approximately 90% of the phonetic boundaries obtained after the convergence of the training algorithm, are within 20 miliseconds of the true boundary locations.

Experiments on TI-46 word : Isolated word recognition experiments were done on the ten digits and the E-rhyme set $\{b, c, d, e, g, p, t, v, z\}$ vocabularies extracted from the TI-46 word speech corpus. The standard speech recognition techniques of dynamic time warping (DTW) and continuous density Hidden Markov models (CD-HMM) [1] were also evaluated for the same task for comparison purposes. The number of states in a HMM word model were set to be equal to the number of phonemes in that word. There were about 48 training tokens and 128 testing tokens for each digit. The table below organizes the performances obtained.

	using CDNTN	using MNTN	DTW	HMM
Digits	88.7 %	82.1 %	71.4 %	97.03 %
E-set	42.9 %	51.4 %	62.8 %	50.22* %

(* optimising the number of states, the best performance of 64.2 % can be obtained for 3 (emitting) states).

To investigate the multiple classifier concept, two neural tree networks (shown as classifier A and B in the table below) were trained for each sub-word segment. The diversity in this network ensemble was obtained by training them with different discriminant data sets. The results obtained on the ten digit vocabulary are tabulated below.

classifier A	classifier B	fusion logic	classifier fusion
87.4 %	83.9 %	$p = p_A \cdot p_B$	92.7 %
87.4 %	83.9 %	$p = p_A^{0.5} \cdot p_B^{0.5}$	85.9 %
87.4 %	83.9 %	$p = p_A^{0.8} \cdot p_B^{0.2}$	93.7 %

[1] using, Cambridge University's HTK software

6. CONCLUSION

A neural tree network -based speech recognition system was presented. The NTNs were used to model sub-word speech segments, while the Viterbi algorithm was used for temporal alignment. A durational probability function was associated with each sub-word speech segment. An iterative training algorithm was outlined. The speech recognition performance was further improved by utilizing the concepts of multiple classifier fusion.

The word model presented in the paper was a homogeneous NN-based system, as opposed to hybrid NN-HMM approaches reported elsewhere. Most hybrid NN-HMM systems discussed in the literature had outperformed baseline HMM systems. However, the proposed word modeling paradigm did not show that trend. The NTN has been reported before to perform favourably as compared to other discriminatory training-based classifiers. Therefore, it is conjectured that the results of our study are representative of the proposed class of homogeneous NN-based speech recognition systems.

7. REFERENCES

[1] S. Renals, N. Morgan, H. Bourlard, M. Cohen, and H. Franco. Connectionist probability estimators in HMM speech recognition. *IEEE Trans. on Speech and Audio proc.*, (vol.2, no. 1, part 2):161–174, January 1994.

[2] R. Schwartz G. Zavaliagkos, Y. Zhao and J. Makhoul. A hybrid Segmental Neural net/Hidden Markov model system for continuous speech recognition. *IEEE Trans. on Speech and Audio proc.*, (vol.2, no. 1, part 2):151–160, January 1994.

[3] S. Katagiri and C-H Lee. A new hybrid algorithm for speech recognition based on HMM segmentation and Learning Vector Quantization. *IEEE Trans. on Speech and Audio proc.*, (vol.1, no. 4):421–430, October 1993.

[4] L. Xu, A. Krzyzak, and C. Suen. Methods of combining multiple classifiers and their applications to handwriting recognition. *IEEE Trans. on systems, man and cybernetics*, 22(3):418–435, 1992.

[5] T.K. Ho, J.J. Hull, and S.N. Srihari. Decision combination in multiple classifier systems. *IEEE Trans. on pattern analysis and machine intelligence*, 16(1):66–75, January 1994.

[6] H. Drucker, R. Schapire, and P. Simard. Boosting performance in neural networks. *Intl. Journal of pattern recog. and artificial intelligence*, 7(4):705–719, 1993.

[7] A. Sankar and R.J. Mammone. Growing and Pruning Neural Tree Networks. *IEEE Trans. on Computers*, (C-42):221–229, March 1993.

[8] K.R. Farrell, R.J. Mammone, and K.T. Assaleh. Speaker recognition using neural networks and conventional classifiers. *IEEE Trans. on Speech and Audio Proc.*, 2(1):194–205, January 1994.

[9] S. Kosonocky and R. Mammone. A continuous density Neural Tree Network word spotting system. In *Proceedings of ICASSP*, 1995.

[10] J.A. Benediktsson and P.H. Swain. Consensus theoretic classification methods. *IEEE Trans. on systems, man and cybernetics*, 22(4):688–704, 1992.