Speech Recognition Using the Probabilistic Neural Network

Raymond Low and Roberto Togneri

Center for Intelligent Information Processing Systems
Department of Electrical and Electronic Engineering
The University of Western Australia

ABSTRACT

A novel technique for speaker independent automated speech recognition is proposed. We take a segment model approach to Automated Speech Recognition (ASR), considering the trajectory of an utterance in vector space, then classify using a modified Probabilistic Neural Network (PNN) and maximum likelihood rule. The system performs favourably with established techniques. Our system achieves in excess of 94% with isolated digit recognition, 88% with isolated alphabetic letters, and 83% with the confusable /e/ set. A favourable compromise between recognition accuracy and computer memory and speech can also be reached by performing clustering on the training data for the PNN.

1. INTRODUCTION

Automated Speech Recognition (ASR) is a difficult task. For a complete system, one needs several layers of processing: the signal pre-processing, feature extraction, phone recognition and syntax analysis. This task may be simplified somewhat by considering isolated word recognition. This removes the complexity of dealing with factors like word boundary segmentation.

Currently, the most successful technique being used in ASR is the Hidden Markov Model (HMM) [6]. HMMs have shown good performance with tractable computation; however, in order to reduce the computation required, they make several assumptions and thus introduce modelling limitations [1].

In order to improve the modelling of time and non-stationary behaviour of speech, the trajectory of speech has been considered [1]. Segment modelling shows the most promise in this regard. Segment modelling involves considering speech where the fundamental unit is a segment, which contrasts with the more conventional unit of a point or vector. The practical implementation of these segment models is, however, rather involved.

2. METHODOLOGY

The motivation behind segment modelling is to look at the segments of speech, these of which have trajectory like representations in vector space. A pattern matching approach can then be applied, where an unknown utterance can be classified by comparing its trajectory with that of the labelled training trajectories. The class of the "closest" training trajectory to the unlabelled one can then be said to be the class of the unknown utterance.

$$C_i = C_j \quad \text{where } \min_j \left\| V_j - V_i \right\|$$

Where C_i is the class of the unknown utterance, V_j is the trajectory of the known training vectors, V_j is the trajectory of the unknown utterance and ||.|| is some distance metric.

This leaves open the question of how to characterise these trajectories and how to find the distance between the two such trajectories. Deng [2] has attempted to fit piecewise polynomial functions to the trajectory of speech. A possible approach to recognition would be to find a fitting polynomial to the trajectory and then compare trajectories by examining the coefficients of the polynomials. The problems with this approach are the difficulty of getting a polynomial to fit satisfactorily, and the computation involved in finding a fitting polynomial.

We propose a novel solution to the task of characterising the trajectories and finding the minimum distance between two trajectories, which is simple to understand and implement. Utterance trajectories are to be treated as simply a collection of unordered points in vector space which have all been labelled with the same class.

This set of data points is then input into a modified version of the statistically based Artificial Neural Network, the PNN. Modification is required as the classical PNN is used for static data classification, whereas we use it for classifying a temporally changing utterance. This form of the PNN represents a wholly non-parametric approach to segment modelling. The modification of the PNN is discussed below.

2.1. Feature Extraction

After data has been acquired, it must be pre-processed in order to reduce the extraneous data that is not pertinent to the classification process. The resulting variables of interest are known as features. Mel-spaced Frequency Cepstrum Coefficients (MFCC) [7] are known to be successful features to use for recognition, and accordingly we have chosen to use the MFCC for our features.

Computation of the MFCC involves the following steps:

1. the signal is windowed to 256-sample frames of duration 20 ms, with 10 ms increment
2. the magnitude of 128 DFT coefficients are then computed
3. using a triangle function spaced according to the Mel-scale, calculate 40 weighted averages
4. calculate the log of the result
5. compute the first 12 inverse discrete cosine coefficients

This process results in a series of 12 dimensional vectors that may then be passed onto a pattern recognition tool.

2.2. Probabilistic Neural Network (PNN)

Artificial Neural Networks (ANNs), which have gained prominence in the area of pattern recognition, have several properties that make them attractive for speech recognition. These include a relatively simple implementation, inherently parallel algorithm (making parallel implementation a natural progression), robustness to noise and self-learning ability. We have chosen the PNN in this experiment.

The Probabilistic Neural Network, introduced by Donald Specht in 1988, is a 3-layer, feed-forward, one-pass training algorithm used for classification and mapping of data [4]. Unlike other ANNs, like the back-propagation neural network, it is based on well-established statistical principles derived from Bayes' decision strategy and non-parametric kernal based estimators of probability density functions. An advantage of the PNN is that it is guaranteed to approach the Bayes' optimal decision surface provided that the class probability density functions are smooth and continuous.

The PNN uses Parzen (or Parzen-like) probability distribution function estimators that asymptotically approach the true underlying parent density, providing it is smooth and continuous. The PNN operates by using spherical Gaussian radial basis functions centered at each training vector. The likelihood of an unknown vector belonging to a given class can be expressed as

$$f_i(\mathbf{x}) = \frac{1}{(2\pi)^{p/2}\sigma^p M_i} \sum_{j=1}^{M_i} \exp \frac{-(\mathbf{x}-\mathbf{x}_{ij})^T(\mathbf{x}-\mathbf{x}_{ij})}{2\sigma^2}$$

Where i is the class number, j is the pattern number, $\mathbf{x}_{ij}$ is the j^{th} training vector from class i, $\mathbf{x}$ is the test vector, M_i is the number of training vectors in class i, p is the dimension of vector $\mathbf{x}$, σ is the smoothing factor (the standard deviation,) and $f_i(\mathbf{x})$ is the sum of multivariate spherical Gaussians centred at each of the training vectors $\mathbf{x}_{ij}$ for the i^{th} class probability density function (pdf) estimate.

Classification decisions are consequently made in accordance with the Bayes' strategy for decision rule, which is $d(\mathbf{x})=C_i$, if

$$f_i(\mathbf{x}) > f_k(\mathbf{x}) \quad \text{for } k \neq i$$

where C_i is the class i.

Modification of the PNN. The classic PNN described above is used for static data classification and needs to be modified in order to handle speech, which is a time series of vectors.

Training utterances are a series of vectors forming a trajectory in vector space. By discounting the temporal order of the vectors, we consider utterances as a *collection* of vectors. Each of the vectors in the training set is then labeled with the class of the utterance, as in a standard PNN.

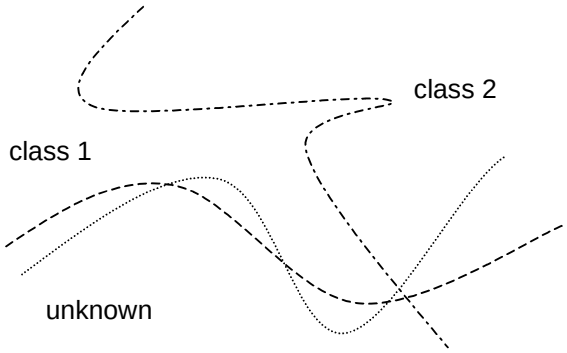

Figure 1: Utterance trajectories in vector space. Each vector in the unknown utterance is classified as either belonging to either class 1 or 2. The maximum likelihood over all the vectors is summed and the most probable class is the classification of the utterance.

Classification of unknown utterances is then a generalisation of the classification procedure of the classic PNN. Typically, a single unknown class input vector has its class likelihood calculated from all the training vectors in the PNN. As it is desired to calculate the likelihood of a collection of vectors belonging to a class, we perform the same calculation on each vector and sum the log likelihood score over all the vectors in the utterance.

$$g_i(\mathbf{y}) = \frac{1}{M_y} \sum_{k=1}^{M_y} \log f_i(\mathbf{x}_k)$$

where g_i is the likelihood of input utterance **y** belonging to class i and M_y is the number of vectors in the utterance **y**. This value is then normalised by dividing by the number of vectors in the utterance, to average over utterances with a long duration. The log function also helps with practical implementation of the PNN by avoiding numeric underflow due to the small magnitude of the values involved.

Then as before, in accordance with Baysian principles, the class with the highest log likelihood is then chosen as the class of the unknown input

$$C_i : g_i(\mathbf{x}) > g_k(\mathbf{x}) \quad \text{for } k \neq i$$

where C_i is the class i.

In the practical implementation, the normalisation of g_i by M_y is redundant as it is always the same input utterance being compared in this case. Should various input utterances (which could have different durations) be scored against each other, M_y would be required.

Whilst this approach to comparing trajectories has the advantage being very simple, a large number of training data is required for good performance of the PNN, which causes problems.

CRAD. As each training vector generates a corresponding pdf in vector space, there is a proportional relationship between the number of training vectors (of which more is better as we would be closer to the optimal class distribution), the amount of memory and computation required and consequently, a decrease in execution.

A solution, which could decrease the amount of memory and computation required, would be to replace the clusters of pdfs (ie. the training vectors) which are “close” together with a single, weighted pdf. This parameter, specifying the threshold distance of the centers of the pdfs, is called the Cluster RADius (CRAD).

We have chosen the simple Euclidean distance as the metric used for determining how close the vectors must be before they are amalgamated into the one training data point.

3. DATABASE

In order to mitigate the problems of data collection, quality control and have a standard against with which to compare the results of other research, a popular speech database was desired.

Two American speaker English databases were used to provide the data for this experiment. These being the Studio Quality Speaker-Independent Connected-Digits Corpus (TIDIGITS) and the Speaker-

dependent Isolated Word Corpus (TI-46), both from the National institute of Standards and Technology (NIST) in the USA.

The digits corpus comprises eleven isolated digits of zero through nine and “oh”, provided by 112 speakers (56 males and 57 females). These have been divided into a training set of 1232 utterances and a testing set of 1243 utterances. The provided utterances have been sampled at 20 kHz and digitized to 16 bit resolution.

TI-46 comprises a variety of data, including 46 isolated words, 10 digits, 10 computer command words and the 26 English alphabet words. 16 speakers (8 male and 8 female) comprise this database and the data has been acquired at the lower quality of 12,500 Hz sampling rate and digitized to 14 bit resolution.

Of the provided data, the digits, alphabet words and a subset of the alphabet words, the confusable /e/ set (comprising of /c/, /d/, /e/, /g/, /p/, /t/, /v/ and /z/) were chosen as the task domains.

We chose to consider speaker independent speech recognition. For the digits task, two utterances for each speaker from the training data set was used for training the PNN. The alphabet and confusable /e/ tasks used two utterances from each speaker, once again from the training set as the training data.

4. RESULTS AND DISCUSSION

With the digits database, the PNN achieves a recognition rate in excess of 94%. The standard Continuous Density HMM from the HTK package, using 5 states, one mixture and non-diagonal covariance matrices, achieves a recognition rate of 96%.

On testing with the alphabet database, the PNN achieves a recognition rate of over 88%. The same HMM gives a recognition rate of 82%. When restricting the classes to the confusable /e/ set, the PNN gives a recognition rate of 83% and the HMM is 81%.

The CRAD used for the digits classes is 100. For the alphabets and the confusable /e/ set it is 63 and 29 respectively.

	Digits	Alphabet	Confusable /e/ set
HMM	96.5%	82.0%	81.2%
PNN	94.1%	88.6%	83.0%
PNN with CRAD	88.7%	84.3%	78.1%

Table 1: Comparison of correct recognition.

The PNN incorporating the training data clustering results in considerably reduced memory requirements and faster operation, at the cost of slightly worse recognition results. The worse recognition results of the CRAD enhanced PNN can be explained by the new weighted clusters not modelling the data distribution as well as the original training data points. The details are shown in Table 2.

	Digits	Alphabet	Confusable /e/ set
Decrease in memory	20%	16%	23%
speed improvement	57%	38%	46%

Table 2: Improvements in memory consumption and time when incorporating the CRAD.

5. CONCLUSION AND FUTURE DIRECTION

This work shows that a simple pattern recognition tool, like the PNN, is able to achieve comparable results to the more conventional HMM on the simple task of word recognition, both for digits and the alphabet words.

Possible future investigations include repeating the above but considering a speaker dependent system, the extension to multisyllabic words, a larger vocabulary and parameterisation of the PNN to improve the computational and memory requirements. There is also scope for further work into investigating the CRAD adjustment to the PNN, regarding the determination of the optimal value to use, and how this could be dynamically chosen by the system, depending on the patterns of the training data.

6. REFERENCES

1. M. Ostendorf, “From HMM's to Segment Models: A Unified View of Stochastic Modeling for Speech Recognition,” *IEEE Trans. Speech and Audio Proc., Vol 4, No. 5, Sept. 1996, p 360-378*.
2. L. Deng, “Speech Recognition Using Hidden Markov Models with Polynomial Regression Functions as Nonstationary States*,” IEEE Trans. Speech and Audio Proc., Vol. 2, No. 4, Oct. 1994, p 507-520*.
3. *HTK v1.4*, Speech Group, Engineering Dept., Cambridge University. 1992.
4. D. F. Specht, “Probabilistic Neural Networks,” *Neural Networks, Vol. 3, 1990, p 109-18*.
5. L. R. Rabiner, *Fundamentals of Speech Recognition*, Prentice Hall, New Jersey, 1993, p 52-53.
6. L. R. Rabiner, A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition, *Proc. of the IEEE., Vol. 77, No. 2, Feb. 1989, p 257-286*.
7. F. J. Owens, *Signal Processing of Speech,* Macmillan, London, 1993. p 53-58.