Applying Modular Neural Networks to Speech Recognition: A Decomposition by Phone

Ednaldo B. Pizzolato and T. Jeff Reynolds
Department of Computer Science
University of Essex, Colchester CO4 3SQ, UK
email reynt@essex.ac.uk

Abstract

Multinet is a modular connectionist classifier architecture. It is layered, with a first layer consisting of at least one primary 'detector' trained individually per class. Its second layer consists of one combining net per class which estimates the posterior probability for that class. In this paper we first consider the motivations for using a modular approach to connectionist classifiers. We then show how Multinet may be used as part of a hybrid HMM-NN speech recognition system. We present results showing that some of the anticipated gains from modularisation are actually delivered, using experiments on TIMIT and RM phone and word recognition tasks.

1 Introduction

Large vocabulary automatic speech recognition systems employing hybrid HMM-NN acoustic modelling have been around for a number of years [14]. A common approach couples a monolithic Multi-Layer Perceptron (MLP) with one output per phone with single-state Markov phone models. Input to the MLP is a sequence of speech frames centred on the current. This allows the MLP to take over the role of modelling the time development of the phone which would otherwise be done by a multi-state HMM. This modelling can then be done with less restrictive statistical assumptions. The MLP is trained with a target of 1 for the output indicating the current phone and 0 for all other outputs. It is well-known that such an MLP can estimate posterior probabilities for all phones, conditional on the input space, provided it has sufficient resources, infinite training data is available and training does not get stuck in a local minimum [3]. Posterior probabilities can be scaled for use instead of emission likelihoods in an HMM framework.

There are a number of practical difficulties which stand in the way of this simple view. In the first section of this paper we shall argue that these indicate that a monolithic MLP may not be the best approach in practice, and that a modular solution could be better. We shall then describe our modular architecture, Multinet, and motivate the detail of its design. The rest of the paper will be given over to a presentation of results from an evaluation of Multinet on TIMIT and RM tasks.

2 The Arguments for a Modular Architecture

2.1 Limited Training Data

However much training data we have we will always be challenged by the sparsity of the data for some phones in some regions of the input space. This is because the data is scattered in a highly non-uniform manner through the space. A potential strength of an MLP classifier is that it can capture correlations in the high-dimensional space of sequences of input frames. Unfortunately this exacerbates the data sparsity problem considerably. We must estimate the parameters of a more complex model from the evidence of the same number of data points, when these points are scattered in a space of higher dimension.

When we have limited training data it is not the case that best estimates of posterior probabilities are obtained when error is at a minimum on the training data. This is because the actual scatter of training data in the input space gives an unreliable estimate of the underlying distribution if observed in too fine a detail. Any proposed strategy for fitting to limited data must be analysed in terms of the expectation of its error over many fits to many data sets, drawn from the same underlying distribution. At each point in the input space this expected error breaks down into three components: an intrinsic error, a bias(squared) and a variance [9]. A close-fit strategy can minimise expected bias but will have excessive variance. Conversely a strategy which fails to fit closely enough may minimise variance but will have excessive bias. An optimum strategy would jointly minimise bias and variance, simultaneously, at every point in the input space. In practice we are not likely to know enough about the statistics of our data to do this directly.

A pragmatic approach to the problem is known as cross-validation. This is a (possibly repeated) procedure where some partition of the available data is made into a training subset and a cross-validation set. During training the error on the cross-validation set is monitored. Typical behaviour is for this error to decrease initially during training, and then at some point, before error is minimum on the training set, to begin to rise again. The minimisation of this error is used as the criterion for early stopping of training. Cross-validation schemes must all compromise to some extent between the need for data to guide the error descent of the net and data to indicate when an early stop should be made.

Once it is realised that optimal modelling from limited data requires the correct bias/variance trade-off at every point in the data space simultaneously, it becomes questionable whether one model fitted to the entire space can ever be optimal. Thus modular methods which sub-
divide the input space for separate modelling may achieve better results.

2.2 Training Times and Modelling Wider Acoustic Context

It is observed that the performance of a hybrid HMM-NN system using an MLP improves as more input frames are employed by the network. There are two sources for this improvement:

- A better view of the target phone is being obtained, even when the input frames to the net are mis-aligned with the boundaries of the observed phone.
- The net begins to see the adjacent phones and can take their expression into account when estimating the posterior for the current phone.

As input frames to the net increase the latter takes over from the former in significance, and we begin to model the target phone in its wider acoustic context. Unfortunately, as the number of inputs gets larger, a monolithic MLP gets harder to train. It is not practical at present to train a monolithic MLP to estimate posterior probabilities on a field-of-view which encompasses phones either side of the current. The natural solution is to find a decomposition of the problem into sub-problems, each solved with a separately-trained network. Potential advantages include the following:

- Training of individual nets may be done in parallel.
- Nets may be individually retrained.
- Network resources can be tuned to need, saving training time, and improving accuracy for limited resources.

Notice that this approach to context-dependent acoustic modelling should be contrasted with the standard approach in an HMM framework. The standard approach splits the modelling of a phone into a set of context-dependent models, each individually estimated from examples of the phone in known phonetic contexts. A possible objection to having a single context-dependent model for each phone is that it is necessarily more complex, as it has to model all of the variability of the phone. However, separate estimation of phone models for all contexts has its problems, as insufficient data is available for estimation in rare contexts. A joint estimation has the value of smooth interpolation across contexts.

2.3 Front-End Processing

Typically speech is processed in fixed, overlapping, frames to give a compact feature vector at each frame. The feature extraction process has been optimised over the years and is now closely matched to the needs of an HMM modelling emission likelihoods with mixtures of Gaussians. The implicit assumption is that speech is a piecewise stationary process, and the emphasis is on accurately modelling the emission of each individual frame of speech given the current Markov state is known. However it has become common practice to augment the feature vector with first and second derivatives. This is a crude but simple approach which allows an HMM to model speech dynamics using state emission rather than state transition. The success of this addition encourages us to think further gains may be possible.

Connectionist acoustic modelling forces some reconsideration of this standard approach. As we have stated, in common hybrid HMM-NN systems an MLP is asked to model the time development of the phone and may be asked to estimate posteriors conditional on a wider acoustic context. The MLP's inputs are typically a sequence of feature vectors, extracted as if for standard HMM modelling. However, there are less redundant ways to represent the time-frequency behaviour of a large window of speech. A whole family of methods is discussed by Cohen in [4].

It is also possible to employ a more heterogeneous approach to feature extraction. It only makes sense to track some speech features, such as voicing and pitch, over fairly long timescales. However the click-like events present in plosives need to be detected using an analysis which gives good time resolution, but consequently poor frequency resolution. Certain types of information may be obtained on timescales of syllables or even words [5][21].

An appealing approach is to tune the feature extraction on a phone by phone basis. A separate MLP can then be trained to distinguish each phone from all others in this special input space. The challenge is to show how modular classifiers, using different input spaces, can be integrated into an architecture which generates posterior probabilities.

3 Modular/Ensemble Approaches to Classification

Modular neural networks [19][1][2] exploit the principle of divide-and-conquer to solve classification problems. An important benefit is that the resources for each unit can be allocated explicitly, and matched to need. In early work Waibel applied modular time-delay neural networks to phone classification [19]. Neural net ensembles [10][12][13], on the other hand, consist of different classifiers nominally trained for the same task. Some method such as averaging must be used for combining their outputs. An ensemble can be a more accurate classifier if its members represent weakly-correlated attempts at the same classification task [18]. However, greater benefits are likely to be obtained if ensemble members concentrate on different regions of the input space. This can be achieved directly using schemes like boosting [17]. Another approach is the Hierarchical Mixture of Experts (HME) architecture of Jordan [11]. The HME architecture explains how experts can be trained whilst simultaneously training gating networks to combine their outputs. Trained experts and gates form a tree structure which may be fixed, or adapted during the development of the classifier [20]. The HME architecture has been applied to speech with good results [7]. The best results are obtained when the hierarchical structure can be adapted to the classification task. This is not easy and it may be necessary to prune the tree structure as well as grow it.

Another approach to hierarchical connectionist acoustic
modelling is the ACID architecture [8]. This architecture is particularly directed at context-dependent acoustic modelling where many 1000s of HMM states may have to be modelled. The approach adopted is a cascaded factorisation of the desired posteriors, giving rise to a tree structure. This also partitions the training data into subsets which are used for the individual estimations of factors. An information divergence criterion is employed to decide exactly how to perform data partitions. A successful application of the criterion is crucial.

4 The Multinet Architecture

The Multinet architecture [15] is a framework for combining heterogenous modular and ensemble classifiers. The modularisation applied is by class and thus, in the case of speech, by phone.

![Figure 1: The Multinet Architecture](image)

The Multinet architecture is shown in figure 1. It is a layered classifier architecture, with individually trained primary 'detectors' followed by a layer of combining nets, one to estimate the posterior probability for each class.

4.1 The Primary Detectors

A variety of primary detectors may be trained. Most simply, a primary phone detector can be trained to distinguish its phone from all others. This can be done by training it on examples of all phones, in their normal proportions. Such a detector will approximate the posterior probability of its phone on the chosen input space.

Phone detectors may be trained on a subset of the available data. So, for example, vowel detectors may be trained on data which excludes the diphthongs. This enables a much smaller field-of-view to be used as input, since time development is not such a crucial issue for vowel discrimination. Phone detectors may also be trained on a re-sampling of the input space.

As we discussed in section 2, front-end processing may be matched to phone. Luckily long phones like diphthongs do not require fine time resolution for identification. Thus we can employ a relatively slow frame rate in preparing training data for diphthongs. This time the plosives and affricatives are excluded from the training. A further benefit is that larger fields-of-view can be represented with less input feature vectors. This can make a large number of detectors very much more efficient.

Two simple approaches to this are available with Multinet: primary detectors can be trained to use such evidence, or the evidence might be regarded as primary, to be passed to all posterior nets for weighting. Either way the evidence can be smoothly integrated.

Whenever more than one primary detector can be designed for a given phone there is the possibility that they make different kinds of errors when classifying that phone. This can easily be detected, and is a sufficient condition for considering the inclusion of both detectors in the architecture. This is the same concept as employed in a neural network ensemble. Combining their opinions in the correct balance will be trained in the posterior layer of the architecture.

4.2 The Posterior Nets

There are as many posterior nets as there are phone classes. Each is connected to all of the outputs of the primary detectors and trained using all of the training data in a separate exercise with the primary detectors utilised, but held fixed. It has a target of 1 for its phone and 0 for all other phones. Its task is to estimate the posterior probability of its phone on the complete input feature space. Notice that it does this using only the opinions of the primary detectors as data. It does not see the original input space, as this would make each posterior net more costly than we desire.

The posterior nets re-scale primary detectors which have been trained on biased input spaces. They combine the opinions of ensembles for the same phone. They also balance detectors which have been trained on different input feature spaces. They are small nets which are quick to train, because the primary detectors do most of the work.

5 Experiments and Results

5.1 The Experimental Set

In order to make comparisons with standard context-independent systems we have implemented an HMM system which employs 3-state models for phones. Emission likelihoods are estimated with continuous mixtures consisting of 8 Gaussians for each state. We have also implemented a hybrid HMM-NN system which can be used to compare Multinet with a standard monolithic MLP. Single-state Markov models are employed, with either of the NN systems providing emission likelihoods from scaled posteriors.

Our training and test data are taken from the TIMIT and RM speech corpora. For TIMIT we train using the recommended training set (3696 si and sx sentences). We then use the full test set (1344 si and sx sentences) to obtain our quoted results. For RM we employ the Oct89 recommended training and test sets. We employ the common 39 phone label set which collapses the 61 TIMIT labels [16]. Each 16ms frame of speech is processed into 15 mel-frequency cepstral coefficients (MFCCs), plus energy, at an 8ms frame rate. We also calculate delta-MFCCs and delta-energy using a window of 5 frames, giving a feature vector
with 32 coefficients in all. All networks are trained with a sum-squared error criterion, whilst monitoring sum-squared error on the cross-validation set. The learning rate is held fixed until error increases on the cross-validation set. It is then halved at each subsequent epoch. A maximum of 10 training epochs are used for each net. We report three kinds of result:

For the TIMIT database only, we evaluate performance using phone classification: all one-state phone models are allowed to compete to label each phone segment as given by the supplied TIMIT markup and the most likely wins. We then count the percentage correct. Notice we implicitly accept the TIMIT markup decision as to the segment label, rather than consulting the lexicon.

For TIMIT and RM databases we perform phone recognition. This consists of finding the most likely phone sequence for each test sentence via a Viterbi alignment. This sequence is then compared with the nominally correct phone sequence as obtained by using the lexicon. A standard string alignment is employed and insertions, deletions and substitutions are counted. We compute an average percentage accuracy for the entire test set.

On the RM database alone we evaluate performance from word accuracy. This is obtained by finding the most likely word sequence for each test sentence via a Viterbi-based decoder. We then extract an average percentage word accuracy in the same fashion as for phones.

5.2 Simple Multinet results for TIMIT

In this section we present an evaluation of Multinet used in its simplest modular form. We trained primary detectors for each phone on the same input feature space of 9 frames, removing the necessity for posterior nets. Three sets of primary detectors were trained, with 50, 100 and 150 hidden nodes each. The monolithic MLP trained for comparison was given 1000 hidden nodes and also 9 frames of input. Table I gives a comparison between the MLP system, the Multinet system with 150 hidden nodes and our HMM mixture of Gaussians system (HMM-MG), performing phone classification on TIMIT.

<table>
<thead>
<tr>
<th>SYSTEM</th>
<th>PHONE CLASSIFICATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>HMM-MG</td>
<td>63.1%</td>
</tr>
<tr>
<td>Monolithic MLP</td>
<td>73.2%</td>
</tr>
<tr>
<td>Multinet-150</td>
<td>75.0%</td>
</tr>
</tbody>
</table>

Table I: Phone Classification on TIMIT (%Correct).

It is always difficult to compare systems, even on the same task. However our HMM-MG system is roughly comparable to context-independent HMM systems. Our hybrid system using a monolithic MLP is also comparable to systems reported in [14]. Therefore the improvement obtained from Multinet is significant. Table II gives a comparison between all the Multinet systems trained and the monolithic MLP performing phone recognition on TIMIT. We can see that Multinet with 50 hidden nodes per primary detector is comparable to the monolithic MLP. Multinet with 150 hidden nodes per primary detector outperforms the monolithic MLP significantly. Total training time for all the primary detectors in Multinet-50 is comparable to the training time for the monolithic net. However Multinet-150 takes around 3 times as long to train, and has around 6 times as many parameters.

<table>
<thead>
<tr>
<th>SYSTEM</th>
<th>PHONE ACCURACY</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monolithic MLP</td>
<td>61.9%</td>
</tr>
<tr>
<td>Multinet-50</td>
<td>62.0%</td>
</tr>
<tr>
<td>Multinet-100</td>
<td>63.1%</td>
</tr>
<tr>
<td>Multinet-150</td>
<td>63.4%</td>
</tr>
</tbody>
</table>

Table II: Phone Recognition on TIMIT.

This obviously poses the question as to what performance might be obtainable with a monolithic MLP of 6000 hidden nodes. Luckily some very useful experiments on large MLPs have been performed by Ellis and Morgan [6]. We reproduce their table here:

<table>
<thead>
<tr>
<th>hidden nodes</th>
<th>Training set size (hrs)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>9.25</td>
</tr>
<tr>
<td>500</td>
<td>42.8</td>
</tr>
<tr>
<td>1000</td>
<td>41.8</td>
</tr>
<tr>
<td>2000</td>
<td>40.4</td>
</tr>
<tr>
<td>4000</td>
<td>40.3</td>
</tr>
</tbody>
</table>

Table III: Word error rate (%) on the Broadcast News Corpus as a function of training set and network size.

This table shows clearly that word error rates decline as more training data is employed and as more hidden nodes are used. It also shows (less clearly) that extra hidden nodes only provide improved performance up to a limit which increases with the size of data set. Our data set is only around 3 hrs, so we must extrapolate from the table. However it looks possible that an MLP of 1000 hidden nodes is already large enough for our data set, or maybe a larger MLP would provide a very slight improvement.

In comparison Multinet-150 delivers a significant improvement in phone accuracy over Multinet-50. What is more, we shall see later with results on RM that a small improvement in phone accuracy, at these levels of performance, leads to a correspondingly greater improvement in word accuracy.

This result supports the views we put forward in section 2. It shows that even a very simple strategy of dividing resources equally by phone allows the training of a better classifier. It shows the power of being able to fit the classifier to different regions of the input space, as independent optimisation procedures. No doubt an even better result could be obtained with the same total resources, if the allocation of hidden nodes was tuned to each phone's requirements.

5.3 Analysing Network Outputs

Some further insights into the relative performance of the two architectures can be obtained if we analyse the outputs of the networks on any data independent of the training
data. Figure 2 shows the average output of the primary detector for /s/ on a frame-by-frame basis. We can see that it responds on average at around 0.7 to frames of /s/ and around 0.35 to frames of /z/ etc. It gives a good measure of the discrimination of the detector, and highlights confusions. However it gives no indication as to whether the output is a genuine posterior probability. Figures 3, 4, and 5 show histograms of the relative frequency of occurrence for example phones against all phones, graphed against the actual primary detector output. We used the test data to produce these graphs. An approximately linear relationship is what is expected if the net is producing posterior probabilities i.e. the region of the input space labelled by the net with an output of 0.2 should have an expected relative density of 20% for the phone. A graph which deviates systematically from this line indicates the net is not producing posterior probabilities.

The expected approximate linear relationship, however, does not actually prove posterior probabilities are being produced. Our two pictures of the behaviour of the net are independent. Maximising the quality of a detector consists of maximising average in-class response subject to a satisfactory histogram. We can see that the output profiles for these primary detectors are reasonably consistent with them producing posterior probabilities.

Figures 6, 7 and 8 show histograms compiled for the outputs from the monolithic MLP for the same example phones. We can see that none of these histograms is consistent with posterior probabilities. The output for /s/ systematically overestimates the posterior over its whole range, and this will cause it to mis-label some segments of other phones. This is confirmed by an investigation of the labelling of phone segments by the two systems during phone classification. The monolithic MLP actually labels a further 152 segments of /s/ correctly (92% over 85% for Multinet). However this is at the expense of mis-labelling 173 segments of /z/, 41 segments of /t/, 32 segments of /sh/ etc as /s/. The optimum trade-off between any pair of phones is the posterior. Over-estimating posteriors for any one phone may label more segments for that phone correctly, but this gain is always more than offset by errors labelling other phones. The monolithic MLP is not uniformly inaccurate in estimating posteriors. It is as accurate as Multinet for some phones. As explained in section 2, it is difficult to get it simultaneously accurate on all phones.

5.4 Simple Multinet results for RM

Phone accuracy is a useful measure of the performance of an acoustic classifier. However we would also like to see how our improvements feed through to word accuracy. To do this we employ the RM database, for which a word-pair grammar is supplied. A Multinet system (Multinet-120) where each primary detector has 120 hidden nodes is compared with a monolithic MLP with 1000 hidden nodes. We chose to use 120 hidden nodes as a compromise based on our experience with TIMIT. No experimental runs were performed with other numbers of hidden nodes. Again, both systems were give 9 frames as input. Alignment of the RM data to prepare it for training was done with our own HMM-MG system.

Table IV presents phone recognition results for the Oct89 RM test set. We get a similar improvement for Multinet over the monolithic MLP to those we obtained with TIMIT.

<table>
<thead>
<tr>
<th>SYSTEM</th>
<th>PHONE ACCURACY</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monolithic MLP</td>
<td>57.1%</td>
</tr>
<tr>
<td>Multinet-120</td>
<td>58.4%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SYSTEM</th>
<th>WORD ACCURACY</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monolithic MLP</td>
<td>69.0%</td>
</tr>
<tr>
<td>Multinet-120</td>
<td>71.8%</td>
</tr>
</tbody>
</table>

5.5 Resampling the input space

One obvious drawback of the simple Multinet arrangement described in the last section is that each phone detector must be trained with a very unbalanced training set. This is unfortunately necessary if the phone detector is to directly produce a posterior probability. However we can freely re-balance or otherwise modify the training set, provided we are prepared to train posterior nets to deliver the final estimates. This can reduce overall training time as each phone detector becomes quicker to train. Training the posterior nets does not add significantly to the overall training time as they can be very small. We may also produce a better classifier as we are concentrating each net's resources more in the region of the phone of interest and less on other regions of the space. We cannot make
Figure 3: Primary Detector Histogram for /sl/

Figure 4: Primary Detector Histogram for /sh/

Figure 5: Primary Detector Histogram for /sil/

Figure 6: Monolithic MLP Histogram for /sl/

Figure 7: Monolithic MLP Histogram for /sh/

Figure 8: Monolithic MLP Histogram for /sil/
ensembles will deliver superior performance.

The first author gratefully acknowledges the support of the AustralianJournalofIntelligentInformationProcessingSystems Summer 1998

Table VI: Phone and Word Accuracy for ROCS on RM.

<table>
<thead>
<tr>
<th>SYSTEM</th>
<th>PHONE ACCURACY</th>
<th>WORD ACCURACY</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROCS-120</td>
<td>60.5%</td>
<td>72.1%</td>
</tr>
</tbody>
</table>

It can be seen that the ROCS-120 system delivers at least the same performance as the original Multinet-120 system. However it takes about half the time to train.

In fact there is a simple analytic relationship between the output expected for ROCS training and the actual posterior, since we have made a uniform reduction in the out-class density. We could have scaled our primary detector outputs directly. However this is not in general true and what we have done illustrates the use of the posterior nets.

6 CONCLUSION

Most of the experiments we have performed so far show Multinet operating in a fairly simple fashion. Nevertheless they show that the performance of a connectionist acoustic classifier can be improved through modular decomposition by phone. The main reason for the improvements observed is that more parameters can be effectively used by the separate networks. This supports one of the arguments we gave in section 2.1 that separate training (and early stopping) of phone detectors allows the creation of a better overall classifier than is possible with a monolithic MLP. We have yet to explore the potential benefit of adjusting the number of hidden nodes given to each phone. This would likely improve classification for any given total resource allocation.

We have also shown that primary detectors for phones need not be trained to generate posterior probabilities directly, and that satisfactory posteriors can be generated by a further layer of simple nets. In our experiment this allowed us to train primary detectors on re-sampled data, and thus reduce training time. We can anticipate that further experimentation with the use of primary detectors as ensembles will deliver superior performance.

Acknowledgement

The first author gratefully acknowledges the support of the Brazilian Agency CAPES (Grant No. 1125/94-8), and the Department of Computer Science, Universidade Federal de Sao Carlos, Brazil.

References


