On the Use of Time-Delay Neural Networks for Highly Accurate Classification of Stop Consonants

Jun Hou, Lawrence Rabiner and Sorin Dusan

Center for Advanced Information Processing, Rutgers University, Piscataway, NJ 08854, USA
{junhou, lrr, sdusan}@caip.rutgers.edu

Abstract

Time-Delay Neural Networks (TDNN) have been shown by Waibel et al. [1] to be a good method for the classification of dynamic sounds such as voiced stop consonants. In this paper we discuss key issues in the design and training of a TDNN, based on a Multi-Layer Perceptron (MLP), when used for classification of the sets of voiced stop consonants (/b/, /d/, and /g/) and unvoiced stop consonants (/p/, /t/ and /k/) from the TIMIT database. We show that by transforming each input parameter to the TDNN to be a zero mean, unit variance distribution (separately for each phoneme class) we can greatly improve the overall classification performance. The resulting TDNN classification accuracy for voiced or unvoiced stop consonants is around 91%. This performance is achieved without any specific discriminative spectral measurements and can be applied directly to the classification of any of the dynamic phoneme classes.

Index Terms: stop consonant classification, Time-Delay Neural Networks, parameter normalization.

1. Introduction

Phonemes in the English language can be partitioned into several broad classes according to various linguistic or acoustic criteria, e.g., vowels, unvoiced fricatives, nasals. Individual phonemes can further be classified as either static or dynamic sounds. Static sounds are those whose temporal/spectral properties are relatively steady during the central part of phoneme articulation, e.g., long vowels and fricatives. A dynamic phoneme’s feature values change significantly over the duration of the sound, either as an essential part of the phoneme itself (e.g., diphthongs, affricates) or as a result of context from the previous and/or succeeding sounds (e.g., semivowels, stops).

Due to their transient, turbulent and noise-like nature, stop consonants are among the most difficult phoneme classes to classify accurately. There have been a number of research studies that tried to accurately classify stop consonants, and most of them applied sets of ad hoc spectral and temporal measurements. For example, Ali et al. used Averaged Localized Synchrony Detection (ALSD), Maximum Normalized Spectral Slope (MNSS) and other acoustic-phonetic features for voicing and place of articulation detection of stop consonants [2], and achieved 96% accuracy for voicing detection and 90% accuracy for place of articulation detection. Suchato studied stop consonants in great detail [3] and used spectral shape, formant structure, noise frequency, etc., for place of articulation classification of stops. He partitioned the stop consonants into CV, VC and VCV structures and achieved 92.1% classification accuracy using a small database from only four speakers. Zheng et al. used formant frequencies (obtained from LPC analysis) for classification of stop consonants with a Support Vector Machine classifier and achieved 88.5% overall accuracy on the place of articulation classification for the TIMIT test set [4].

Waibel et al. [1] showed that a special class of neural networks, called Time-Delay Neural Networks (TDNN), was effective for classifying voiced stop consonants in Japanese. The specific TDNN used in [1] looked for stop burst events occurring within a set of 30 msec sliding windows within a fixed 150 msec segment, and classified the segment based on the match to a trained TDNN network. In [5], several TDNNs were connected together to recognize the complete set of Japanese phonemes. Many variations of the original TDNN have been proposed and studied, including an N-of-TDNN [6], an Adaptive Time-Delay Neural Network (ATNN) [7], a Frequency-time-shift-invariant TDNN (FTDNN) [8], etc.

This paper describes the results of a series of experiments in the use of TDNNs for accurate and automatic classification of dynamic sounds of speech. As a baseline for our investigations, we first compared the discrimination abilities of TDNN and MLP neural networks for stop consonants classification. Then we studied several key issues in TDNN training. We compared the use of a Gradient Descent algorithm with a Scared Conjugate Gradient Descent algorithm for optimization of the TDNN weights. We investigated the time-delay nature of the TDNN and showed that by incorporating dynamic speech parameters we could improve the discrimination capabilities of the network. We used a simple method to utilize both the discrimination capability of neural networks and the classification capability of pattern recognition methods by normalizing the TDNN inputs to zero mean, unit variance distributions separately for each phoneme class. Using this method, the classification accuracy of the TDNN was greatly improved. The ultimate classification accuracy for the TDNN classifier was 91.9% for unvoiced stop consonants and 90.9% for voiced stop consonant classification, when testing on the independent TIMIT test set.

2. Time-Delay Neural Networks

A TDNN introduces delays into the input of each layer of a traditional MLP as illustrated in Figure 1 [1]. When used to classify the set of voiced stop consonants (/b/, /d/, /g/), the TDNN looks for a stop burst event in the time-delayed frames using duplicated weights, and relates the current input to the past history of events in the segment-based input feature set. As shown in Figure 1, in a hidden layer an input \( w_i \) is delayed (denoted by \( D_{1,...,D_n} \)) and multiplied by a set of weights \( \left( w_{i,1,...,w_{i,n}} \right) \), and then summed and passed through a nonlinear function \( F \). The TDNN encodes temporal relationships within a segment in this manner.

A TDNN toolbox was designed and implemented using the Netlab toolbox [9] as a reference and starting point. The toolbox included routines for training and evaluation. In this
section we discuss three issues in TDNN training: the TDNN parameter optimization criterion, the need for temporal features, and the need for normalization of the input features.

2.1. Parameter optimization algorithm

We tested two optimization algorithms for error back propagation to adjust the weights, namely the Gradient Descent (GD) algorithm and the Scaled Conjugate Gradient Descent Algorithm (SCGD). We found that with the SCGD the TDNN convergence was very slow, and sometimes it did not converge at all. Using the GD algorithm each step in the optimization was small, but convergence was always obtained. Thus we used the GD algorithm for the optimization of the weights.

2.2. Incorporating dynamic features into TDNN inputs

The time-delayed nature of the TDNN takes into account fine temporal information when looking for stop burst events within 3 consecutive frames of the input by using duplicated weights. Suppose the number of delayed nodes in each layer is $D(l)$, $l=1,\cdots,L$. Each frame in the spectrogram-like input of each layer is $u(l)_{j,n}$, $n=1,\cdots,N(l)$, $m=1,\cdots,M(l)$, $l=1,\cdots,L$, where $N(l)$ is the dimension of each frame in layer $l$, $M(l)$ is the number of frames in layer $l$, and $L$ is the total number of hidden layers. At each node we use the sigmoid function

$$F(u) = \frac{1}{1+\exp(-u)}$$

(1)

For hidden layers $l=1$ and $l=2$, the activation of each node $a_{i,j}^{(l)}$ can be written as

$$a_{i,j}^{(l)} = \sum_{m=1}^{M(l)} \sum_{n=1}^{N(l)} w_{i,j,m,n} u_{i,m,n}^{(l)} + b_{i,j}^{(l)}$$

(2)

where $w_{i,j,m,n}$ are the weights and $b_{i,j}^{(l)}$ are the biases. Eq. (2) shows that the output of each node in the two hidden layers of the TDNN computes the weighted average of the hidden layer inputs using the delayed weights. From Eq. (2) we see that the TDNN has the ability to utilize temporal information within $D(l)+1$ frames in layer $l$. Thus when we add additional temporal information to the input layer of the TDNN, if the total time span of the new parameters covers more than $D(l)+1$ frames, then it potentially contains new information which can be used to improve classification performance of the TDNN.

The set of mel frequency cepstral coefficients (MFCC) and their first and second order derivatives (usually computed as simple delta and delta-delta features) are the most commonly used speech parameters in modern automatic speech recognition systems. In an effort to add additional temporal information to the TDNN we included the first and second order deltas of the MFCC coefficients to the input feature vector, using a range of 5 frames for the calculation of the first (and second) order delta (and delta-delta) MFCC features.

2.3. Transformation of the TDNN input

In statistical pattern classification, we assume that there are $C$ classes and we train a set of $C$ models, one for each class. On the other hand, standard neural networks simply use the speech parameters as the input, without any knowledge of the models trained for each class. In order to take into account knowledge of the statistical models, we can define a function to transform the input vector using the parameters in each model and train one neural network based on the transformed vectors for each of the $C$ classes. When evaluating all $C$ neural networks, each test token is transformed $C$ times according to each model, the transformed token is fed into each net, and then the maximum score is chosen as the final classification.

It is well known that neural network training is affected by the dynamic range of the input parameters. The most rapid convergence in training occurs when each input parameter has a common dynamic range. Suppose the input frame, $X$, is $N$-dimensional, i.e. $X = (x_1, x_2, \ldots, x_N)$. Assume that each parameter in the frame is independent of all other parameters, and approximately modeled by a Gaussian distribution. In this case each parameter, $X_j$, can be normalized to a zero-mean, unity variance Gaussian, $\hat{X}_j$, by the transformation

$$\hat{x}_j = \frac{x_j - \mu_j}{\sigma_j}$$

(3)

where $\mu_j = \mathbb{E}(X_j)$ is the mean, and $\sigma_j$ is the standard deviation for each parameter. In this manner, all of the training data are normalized by one mean vector and one standard deviation vector calculated from all training tokens.

Using the normalization as the transformation function, and using the mean and variance as the model parameters, we calculate $(\mu_j^{(i)}, \sigma_j^{(i)})$, $j=1,\ldots,N$ for each class $i=1,\ldots,C$, normalize all the input tokens on $(\mu_j^{(i)}, \sigma_j^{(i)})$, $j=1,\ldots,N$ for each class separately, and train a set of $C$ TDNNs with each TDNN trained on the data normalized on $(\mu_j^{(i)}, \sigma_j^{(i)})$, $j=1,\ldots,N$ for one class. Finally when evaluating the TDNNs, we select the maximum score as the final classification.

3. Experimental Results

The training set for /b/, /d/, /g/ and /p/, /t/, /k/ consisted of all the sounds in the 8 dialect regions in the TRAIN set of the TIMIT database (except for tokens from the SA sentences) in which the voiced stop sounds were preceded by any phoneme and followed by a vowel or a diphthong (i.e., utterances of the
Table 1. Number of tokens used in training and testing for both voiced and unvoiced stops.

<table>
<thead>
<tr>
<th></th>
<th>Voiced stops</th>
<th>Unvoiced stops</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>/b/</td>
<td>/d/</td>
</tr>
<tr>
<td>Train</td>
<td>1567</td>
<td>1460</td>
</tr>
<tr>
<td>Total</td>
<td>3685</td>
<td>3685</td>
</tr>
<tr>
<td>Test</td>
<td>638</td>
<td>537</td>
</tr>
<tr>
<td>Total</td>
<td>1418</td>
<td>1418</td>
</tr>
</tbody>
</table>

form *CV*. The independent test set was the TIMIT TEST set of *CV utterances (without any tokens from the set of SA sentences) from the 8 dialect regions. The number of tokens used in training and testing is given in Table 1.

We trained the TDNN (and the MLP) for voiced stops and unvoiced stops separately, and built one TDNN for /b, d, g/ classification and another for /p, t, k/ classification. The input layer, for each of the TDNN consisted of 15 frames with the beginning of the succeeding vowel at the 10th frame in the *CV segment. Each individual frame consisted of the set of 13 MFCC coefficients and its first and second order deltas. Using MFCC features only, there were 15*13=195 parameters in each segment, using MFCC+Δ+ΔΔ features, there were 15*39=585 parameters in each segment. Each frame was calculated over a 10 msec interval using a Hamming window with 5 msec window overlap between frames. Adjacent frames were averaged to smooth the data, resulting in a 10 msec frame update rate. The TDNN had 2 hidden layers. The first hidden layer had 8 nodes, with delay $D^{(0)}=2$. The second hidden layer had 3 nodes, with delay $D^{(0)}=4$. The output layer had 3 nodes, one for each of /b/, /d/, and /g/ or /p/, /t/, and /k/. The maximum output from the 3 output nodes of the TDNN was selected as the classification for the current segment.

Similar to Waibel [1] we used staged batch training. The numbers of training tokens used, at each new training level, were: 3, 6, 9, 24, 99, 249, 780 and finally all the *CV tokens in the training set. All the tokens were randomly selected from the training set. The first 7 training sets were balanced in the number of occurrences of /b/, /d/, and /g/ or /p/, /t/, and /k/; the last training set was unbalanced, using all the tokens of voiced stops or unvoiced stops from Table 1.

3.1. Comparison of TDNN with 3-Layer MLP

To determine the inherent advantage of a TDNN over a conventional MLP, we trained an MLP with an input layer that had the same segment length as that used for the TDNN, namely 150 msec, with 15 frames each having 13 MFCC coefficients, with 195 input nodes in total. The first hidden layer had 8 nodes, the second hidden layer had 3 nodes, and the output layer had 3 nodes, one each for /b/, /d/, and /g/. We measured a single mean and standard deviation for each of the 13 MFCC parameters using all the input tokens, and then we normalized each of the inputs appropriately. Table 2 shows the classification accuracy of the TDNN and the MLP networks on the training set and the test set when all 3685 tokens were used for training for this overall normalization method. We see that the MLP performed considerably better than the TDNN on the training set, but it performed considerably worse on the test set.

In order to compare the results of Table 2 with Waibel’s work [1], we need to recall that Waibel’s TDNN was trained on Japanese CV utterances, where there were only 5 vowels; hence there were only 15 possible CV combinations. Also, Waibel’s TDNNs were speaker dependent, with one TDNN trained for each speaker. The English language has 11 vowels and 4 diphthongs (in the 40-phoneme alphabet), so there are $(11+4)^3=45$ possible CV combinations. Further, the TDNN that we created was trained on multiple speakers and therefore was speaker independent. Hence there is no simple direct comparison of results; however we see that the performance obtained using the TDNN on the test set is quite high, considering the major differences in system specifications.

3.2. Results when using MFCC and its delta and delta, delta features

The delta and delta-delta parameters were calculated using standard definitions over a 5-frame window. A comparison of TDNN network classification performance (in terms of percentage accuracy of classification) is given in Table 3.

Table 3. TDNN classification accuracy (%) for /b, d, g/ on training and test sets using MFCC and MFCC+Δ+ΔΔ feature sets.

<table>
<thead>
<tr>
<th>Features</th>
<th>Training set</th>
<th>Test set</th>
</tr>
</thead>
<tbody>
<tr>
<td>MFCC</td>
<td>95.3</td>
<td>86.7</td>
</tr>
<tr>
<td>MFCC+Δ+ΔΔ</td>
<td>98.8</td>
<td>87.7</td>
</tr>
</tbody>
</table>

From Table 3 we see that the TDNN performance is improved by a small amount by incorporating the dynamic features.

3.3. Results with transformed TDNN inputs

As a second normalization method we calculated one mean and one standard deviation for each of the 39 parameters for each of the 3 voiced stop consonants, $(\mu^{(i)}, \sigma^{(i)})$, $i=1,...,3$. Then we trained the TDNN on the individually normalized training tokens. The resulting classification accuracy was 100% on the training set. Then we normalized each test token using the correct mean and standard deviation, as calculated from the training set for this phoneme class, and we achieved 100% classification accuracy on the test set. This procedure is, of course, invalid because when we do real world testing, we do not know which $\mu^{(i)}$ and $\sigma^{(i)}$ to use for feature normalization; otherwise we would have known the phoneme class. This test was performed just to determine whether the phonemes could be clearly separated when normalizing each phoneme class separately to the appropriate standard normal distribution.

As a valid evaluation we normalized the test token three times, using each of $(\mu_{b}, \sigma_{b}), (\mu_{d}, \sigma_{d})$ and $(\mu_{g}, \sigma_{g})$, and evaluated the TDNN outputs using the TDNN trained above for each of the separately normalized tokens, and selected the maximum score for classification, but only achieved 80% classification accuracy on the test set.

Using the transformation method described in the previous section, we trained one TDNN using tokens normalized on $(\mu_{b}, \sigma_{b})$, another TDNN using tokens...
normalized on \( (\mu_d, \sigma_d) \) and still another one using \( (\mu_g, \sigma_g) \). When we did the testing, we normalized the test token using each of \( (\mu_b, \sigma_b), (\mu_t, \sigma_t) \) and \( (\mu_k, \sigma_k) \) separately and calculated the outputs from the three TDNNs. Then we selected the maximum score as the final classification result. Using this method, we achieved 90.9% accuracy for voiced stop classification on the 1418-token test set, i.e., an improvement of 3.2% in accuracy.

Similarly, for unvoiced stop consonant classification, we calculated \( (\mu_v, \sigma_v) \) on the /p/ training tokens, \( (\mu_u, \sigma_u) \) on the /u/ training tokens, and \( (\mu_t, \sigma_t) \) on the /k/ training tokens. Then we trained three separate TDNN’s with all the input tokens normalized on each set of mean and standard deviation. We again tested the performance of the resulting TDNN’s by selecting the maximum output from the 3 TDNNs as the final classification. We achieved 98.6% classification accuracy on the training set and 91.9% accuracy on the 1808-token test set. The results are shown in Table 4, where N01-all denotes normalizing the training tokens using one global mean and one global standard deviation, N01-3 denotes using three means and standard deviations to normalize the inputs and train three TDNN’s.

Table 4. Classification accuracies (%) of different feature normalization methods on both voiced and unvoiced stop consonants.

<table>
<thead>
<tr>
<th></th>
<th>Training set</th>
<th>Test set</th>
</tr>
</thead>
<tbody>
<tr>
<td>Voiced stops</td>
<td></td>
<td></td>
</tr>
<tr>
<td>/b, d, g/</td>
<td>N01-all</td>
<td>98.8</td>
</tr>
<tr>
<td></td>
<td>N01-3</td>
<td>97.9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>90.9</td>
</tr>
<tr>
<td>Unvoiced stops</td>
<td></td>
<td></td>
</tr>
<tr>
<td>/p, t, k/</td>
<td>N01-all</td>
<td>99.1</td>
</tr>
<tr>
<td></td>
<td>N01-3</td>
<td>98.6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>91.9</td>
</tr>
</tbody>
</table>

3.4. Discussion

To compare our results with other research efforts for the classification of stop consonants, we first need to note the statistical divergence between different classification systems. In Ali’s work on stop consonant classification [2], the test was done on 7 dialect regions consisting of 1200 stops, and he achieved 90% accuracy for place of articulation detection of the 6 stop consonants. In Suchato’s work on stop consonant classification [3], the database was only 2 male speakers and 2 female speakers, and he achieved 92.1% accuracy on 4007 stops. Zhang et al. achieved 88.1% accuracy on the place of articulation detection of all the stops in the TIMIT test set of 5725 stops [4]. All of the above research efforts were conducted using a number of hand selected acoustic (spectral and/or temporal) features such as formant tracks.

In our work, the classification of place of articulation detection was partitioned into voiced stops and unvoiced stops, and we achieved 90.9% classification accuracy on 1418 voiced stops and 91.9% accuracy on 1808 unvoiced stop tokens of the *CV form from the 8 dialect regions in the TIMIT database. The average place of articulation classification accuracy would be 91.5%. This performance was achieved using only MFCC and its delta and delta delta features, without any specific acoustic phonetic measurements especially tailored to the problem of stop consonant classification. The results presented here represent state-of-the-art performance for stop consonants place classification, on a large database, using algorithmic methods. We believe that there is a lot of potential for utilizing the methods described in this paper for the general problem of attribute detection in speech [10].

4. Conclusions

In this paper we studied the class of Time-Delay Neural Networks and measured its performance for classification of voiced and unvoiced stop consonants in English. We showed that TDNNs, trained on the TIMIT database, generalized much better on an unknown test set than a traditional MLP network without the delay features. We found that the use of MFCC and its delta features provided additional, and highly reliable information for stop consonant classification. We also studied a transformation method for incorporating both the classification capability of statistical pattern recognition systems and the discrimination ability of neural networks. We used a simple normalization procedure as the transformation function of the neural net feature inputs, and found that by training the classifier based on different normalization methods, the classification accuracy improved above that obtained from uniformly normalized input features. Overall we achieved 90.9% classification accuracy on voiced stops and 91.9% classification accuracy on unvoiced stops. Our experiments were conducted without any specific acoustic information specially tailored for the classification of stop consonants.

5. Acknowledgements

This work is supported under the NSF ITR grant, IIS-04-27413.

6. References


