Speech Recognition of Phones Using Feature Streams

Todd A. Stephenson
Centre for Cognitive Science, University of Edinburgh
2 Buccleuch Place, Edinburgh EH8 9LW, Scotland, UK
E-mail: *Todd.Stephenson@ed.ac.uk*
Phone: +44 131 650 4414; Fax +44 131 650 6626

May 11, 1998

Abstract

Artificial neural networks (ANNs) have been used to classify phonetic features in speech. The feature streams from the ANNs are used here as the observations for Hidden Markov Models (HMMs). Using such observations allows us to build a competitive speech recogniser. This recogniser is compared to a similar recogniser that was trained on mel-frequency cepstral coefficients (MFCCs). While the cepstral HMMs perform better, the feature HMMs do exhibit their potential for further development.

1 Introduction

HTK (Young *et al.*, 1996) is a popular tool for constructing speech recognisers. HTK is used to train HMMs to recognise speech according to a specified dictionary. The normal input for training the HMMs is MFCCs (Rabiner and Juang, 1993). These have proven to be one of the best methods for training speech recognisers.

A new area of research involves the possibility of training the HMMs on feature streams. That is, instead of giving the cepstral coefficients for each frame of speech, give the feature values for that frame of speech. In this method, you would indicate values for features such as voicing, place of articulation, roundness, etc., for each frame of speech. An algorithm is needed for automatically labelling the features for each frame of speech. The neural networks from Stephenson (1998) are used here to classify the phonetic features for speech. The networks used classify according to the following features: centrality, frontback, manner, phonation, place, and roundness. Table 1 provides for 27 distinct classifications in this feature set of the phones specified in the TIMIT speech corpus (Garofolo, 1988).

The TIMIT speech corpus was used for this project. It was split in the standard way into training and testing utterances. A small portion of the training data was not used in the actual training but was reserved for fine tuning the performance of the models. The results given below are from the test set of utterances.

2 The Models

Two sets of models were trained to recognise phones, both having a dictionary of 52 phones (Table 1) plus SIL and SP. As a baseline, HMMs were trained on MFCCs (referred to as

PHONES	PHONATION	MANNER	PLACE	FRONTBACK	ROUNDNESS	CENTRALITY
sil,sp	sil	sil	sil	sil	sil	sil
ch,s,sh	unvoiced	fricative	coronal	front	unrounded	nil
f	unvoiced	fricative	dental	front	unrounded	nil
hh	unvoiced	fricative	glottal	back	unrounded	nil
t	unvoiced	occlusive	coronal	front	unrounded	nil
q	unvoiced	occlusive	glottal	back	unrounded	nil
p	unvoiced	occlusive	labial	front	unrounded	nil
k	unvoiced	occlusive	velar	back	unrounded	nil
el	voiced	approximant	coronal	front	unrounded	full
l	voiced	approximant	coronal	front	unrounded	nil
w	voiced	approximant	labial	front	rounded	nil
er	voiced	approximant	velar	back	unrounded	full
r,y	voiced	approximant	velar	back	unrounded	nil
jh,z,zh	voiced	fricative	coronal	front	unrounded	nil
dh,th,v	voiced	fricative	dental	front	unrounded	nil
hv	voiced	fricative	glottal	back	unrounded	nil
en,n,nx	voiced	nasal	coronal	front	unrounded	nil
em,m	voiced	nasal	labial	front	unrounded	nil
eng,ng	voiced	nasal	velar	back	unrounded	nil
(*)d,dx	voiced	occlusive	coronal	front	unrounded	nil
(*)b	voiced	occlusive	labial	front	unrounded	nil
(*)g	voiced	occlusive	velar	back	unrounded	nil
ow,oy,uh,uw,ux	voiced	vowel	high	back	rounded	full
ey,ih,ix,iy	voiced	vowel	high	front	unrounded	full
aa	voiced	vowel	low	back	unrounded	central
ae,ah	voiced	vowel	low	front	unrounded	full
ao,aw	voiced	vowel	medium	back	rounded	full
ay,eh	voiced	vowel	medium	front	unrounded	full
ax,ax-h,axr	voiced	vowel	medium	nil	unrounded	central

Table 1: *The monophone models. The 54 phones are listed above. They are grouped according to identical features, based on the feature set used by the ANNs. NB: The (*)'ed phones, /d/, /b/, and /g/, are voiced stops. They were divided into two parts when training the ANNs (an unvoiced closure followed by a voiced release); this is how the original TIMIT labels are done. However, the unvoiced closures were collapsed into the voiced releases for the HMM portion of this project.*

cepstral HMMs). The other set were trained on the output from the feature streams of the ANNs (referred to as feature HMMs). I will first describe the observation vectors of the cepstral HMMs and of the feature HMMs. Then I will explain how the monophone models were constructed. Finally I will go on to tell how the tied-state triphone models were trained from these monophone models.

The cepstral HMMs are the baseline for this study and have observation vectors of size 39. These are the 12 cepstral coefficients with an additional position for the energy value of the given frame. This gives the first 13 positions. The next 13 positions are the delta values for the first 13. That is, they state the difference between the current coefficient and energy values with the previous frame's coefficient and energy values. This gives an approximate value for the first derivative. The final 13 positions in the observation vector are the acceleration values. These are computed by taking the difference between the current delta values with the previous delta values. This gives the approximate second derivative value for the frame. The input into these cepstral HMMs are essentially the same as the input to the actual ANNs in Stephenson (1998). The ANNs also take the initial vector of 12 cepstral coefficients and 1 energy value; they then have hidden layers to compute the delta and acceleration values.

The feature HMMs, which are the focus of this paper, have observation vectors of size 29. They are constructed by pasting the outputs of all of the neural networks together. Each individual ANN has the following sized output vectors: centrality, 4; frontback, 4; manner, 6; phonation, 3; place, 9; roundness, 3. So, the sum of the size of these output vectors gives us the size of the observation vectors for the HMMs. I did not add any delta or acceleration positions to the feature HMMs. This is because the ANNs themselves utilised delta and acceleration values in their network architecture; so, the input into the feature HMMs should already represent these values in some way.

For each set of monophone models, the following is done, based on the tutorial given in Young *et al.* (1996). I use a 3 state model, not including the necessary entry and exit states. `H2CompV` is used both to estimate the initial prototype model and to determine variance floors; these computed variance floors are then used in all the further training. For all 54 monophone models in Table 1, `H2Init` and then `H2Rest` are applied. These take the segments of speech that pertain to the concerned monophone and uses these to train the model; it uses a provided label file to determine which frames to use for training each model. Once each monophone model is individually trained, they are all collectively trained using `H2ERest` to do two iterations of the Baum-Welch algorithm. At this point, the silence models SIL and SP are now fixed as follows: transitions are placed between states 1 and 3 and between states 3 and 1 of SIL. Furthermore, SP is made a 1 state model. Its state is tied to the centre state of SIL and also has a transition that allows its 1 state to be bypassed. Embedded re-estimation is again done twice with `H2ERest`. The monophone models are now trained.

These trained monophone models are then used as the basis for training cross-word triphone models. The HTK tool `H2LEd` is used both to create the triphone label files from the monophone label files and also to generate a list of triphones for which there are examples in the training data. `H2HEd` clones the triphones from the already trained monophones and ties the transition matrices of like triphones together. Using the list of training triphones and the cloned, tied triphones, `H2ERest` is invoked twice to train these new triphone models. Statistics on the type of training data is generated when `H2ERest` is called here. Using a specified decision tree and these statistics, `H2HEd` clusters the data according to criteria such as the left or right context of a triphone or the phonetic features of a triphone. This clustering allows both the pooling of training data for related triphones and the training of models for which

WORD: %Corr=56.84, Acc=51.83 [H=24716, D=4278, S=14490, I=2180, N=43484]

Table 2: *Results for the feature HMMs. The phones* SIL *and* SP *are assigned to the null equivalence class, but no other equivalencies are made.*

WORD: %Corr=64.07, Acc=59.93 [H=27862, D=3780, S=11842, I=1803, N=43484]

Table 3: *Results for the cepstral HMMs. The phones* SIL *and* SP *are assigned to the null equivalence class, but no other equivalencies are made.*

there is not any data available. Finally, two more iterations of the Baum-Welch algorithm are applied with `H2ERest`. The cross-word, tied-state triphone models are now trained, and it is on these models that the following results apply.

3 Results

First I will give the overall results, comparing the baseline cepstral HMMs with the experimental feature HMMs. Then I will give a case where the feature HMMs outperformed the cepstral HMMs. This will be followed by a case where the cepstral HMMs outperformed the feature HMMs. Finally, a case will be given showing how both sets of models make similar errors. You will notice that the overall deletion:insertion ratio in each of Tables 2, 3, 4, and 5 is approximantly 2:1. This means that the results are not at their optimum; however, with all four tables having a similar ratio, we can make comparisons between the two types of HMMs.

The feature HMMs were able to train themselves well (Table 2), considering the incompleteness of the feature set. They did not train as well as the cepstral HMMs (Table 3). Nevertheless, their performance is close to that of the cepstral HMMs, 56.84% versus 64.07%. Their performance is even closer when you take into account the phones that the feature HMMs saw as having equivalent feature values (see Tables 4 and 5). When the equivalence classes are formed in both sets, the difference between the results of the feature HMMs and the cepstral HMMs is smaller, 65.60% versus 69.65%.

Having now shown the two sets' overall results, I now look at a class of vowels where the feature HMMs outperformed the cepstral HMMs. They did so with high, back vowels. From Table 1, notice that the phones OW, OY, UH, UW, and UX have the same feature set. That is, the neural nets were not trained to distinguish among these phones. From Table 6, you can see the correctness results that the HMMs had in classifying each of these phones. In this case, the feature HMMs were able to distinguish among identically classified phones better than the cepstral HMMs did. So, the output from the feature classifying ANNs provided enough difference in their values for the HMMs to see a difference with the identically featured phones. Furthermore, when collapsing these phones into a single equivalence class (Table 7), the feature HMMs did well, again better than the cepstral HMMs.

WORD: %Corr=65.60, Acc=60.57 [H=28525, D=4285, S=10674, I=2187, N=43484]

Table 4: *Results for the feature HMMs with like featured phones collapsed into equivalence classes. In addition to the phones* SIL *and* SP *being assigned to the null equivalence class, additional equivalencies are made. An equivalence class is formed for each line in Table 1.*

WORD: %Corr=69.65, Acc=65.38 [H=30285, D=3831, S=9368, I=1854, N=43484]

Table 5: *Results for the cepstral HMMs with like featured phones collapsed into equivalence classes. In addition to the phones* SIL *and* SP *being assigned to the null equivalence class, additional equivalencies are made. An equivalence class is formed for each line in Table 1.*

PHONE	Feature HMMs	Cepstral HMMs	Num Training Samples	Num Test Samples
OW	56.1 %	55.0 %	1589	600
OY	36.0 %	35.0 %	289	127
UH	17.9 %	23.9 %	482	215
UW	25.0 %	42.3 %	511	157
UX	54.4 %	50.1 %	1371	415

Table 6: *Correctness for vowels classified identically in the feature set. Notice that while the the feature streams do not distinguish among these phones, the HMMs were able to learn to distinguish them. In most cases, the feature HMMs actually classified these with a higher correctness than the cepstral HMMs.*

Phones	Feature HMMs	Cepstral HMMs
OW, OY, UH, UW, UX	57.8 %	54.0 %

Table 7: *Correctness for vowels classified identically in the feature set. The above vowels have equivalent features. In this table, the listed vowels have been collapsed into an equivalence class. That is, when expecting a certain vowel within this group, if you get any of the vowels in the group, classify it as correct. The feature HMMs did better than the cepstral HMMs.*

PHONE	Feature HMMs	Cepstral HMMs	Num Training Samples	Num Testing Samples
CH	50.2 %	73.4 %	784	259
S	81.0 %	88.4 %	5888	2172
SH	39.8 %	81.2 %	1257	460

Table 8: *Correctness for unvoiced, coronal fricatives, which are classified identically in the feature set. The feature streams do not distinguish among these phones. The cepstral HMMs were able to learn to distinguish them well. The feature HMMs did worse in identifying them. Notice however, the bias in the training data towards* S.

Phones	Feature HMMs	Cepstral HMMs
CH, S, SH	90.4 %	91.72 %

Table 9: *Correctness for unvoiced, coronal fricatives, which are classified identically in the feature set. The feature streams do not distinguish among these phones. When collapsed into an equivalence class, the feature HMMs do almost as well as the cepstral HMMs.*

While the feature HMMs did slightly better with the high, back vowels than the cepstral HMMs did, the cepstral HMMs did better overall. For example, the feature HMMs had problems with unvoiced, coronal fricatives (Table 8). With the given feature set the feature HMMs had great trouble distinguishing among them and a lot of the time would only correctly classify the phone S, which had the greatest number of samples of this equivalence class in the training data. However, by collapsing the CH, S, and SH into an equivalence class, the feature HMMs do almost as well as the cepstral HMMs (Table 9).

The above results deal with the correct classifications by the HMMs. Now both sets also made intelligent incorrect classifications. That is, if they wrongly identified a phone, they tended to do so by giving a related phone. Table 10 shows how for G, both sets had related stops as their next two most common classifications for frames that should be G.

4 Discussion

The above results show that feature streams have the potential for constructing highly accurate HMMs. While these feature HMMs do not perform as well as the cepstral HMMs, they do show signs that if properly tuned they can outperform the cepstral HMMs. I will talk about the advantages and disadvantages of the used feature set in the following areas: the ability of the feature HMMs to distinguish among some phones with identical features; the inability of the feature HMMs to distinguish among some phones with identical features; the intelligence of the HMMs to misclassify a phone as a related phone. Finally, I will close with

	G	D	K	Correctness
Feature	190	29	126	38.1%
Cepstral	255	57	67	50.4%

Table 10: *For all frames that should be labeled* G*, the number that were correctly classified as* G *and the two highest incorrect labels for a* G *frame. The total number of testing samples of* G *is 557.*

a proposal for future work.

Tables 6 and 7 showed the ability of the feature HMMs to distinguish among identically valued phones. The particular choice of features in this study worked well in recognising these high, back vowels. It would not have been surprising if the feature HMMs got very confused in classifying these vowels because the feature set does not distinguish among them. However, the feature HMMs did better than expected. They were able to do a better job, overall, within this group of vowels than the cepstral HMMs.

This ability of the feature HMMs to distinguish among the phones here may be due to the following. While all of the vowels in Table 6 should have a large 'high' value of 1.0 from the 'place' ANN, one phone may have its value clustered around, say, 0.90 and another around 0.80 because the ANN saw the former as being actually slightly higher than the latter. These may also have had values for 'medium' of, say, 0.10 and 0.20, respectively. So, these results imply the importance of having feature values that are related to each other. This may allow the ANNs to classify sounds that are a mixture of any of them.

Not every set of identically valued phones did as well in the feature HMMs as those above. However, the feature HMMs still show their competitiveness to the cepstral HMMs. In Table 8 I showed how the CH and SH in the feature HMMs lagged significantly behind their performance in the cepstral HMMs. This can be partly attributed to their shown distribution in the training data; this can not, however, be the whole reason because Table 6 shows how the feature HMMs were able to distinguish among the high, back vowels despite their bias in the training set. These fricatives, therefore, could possibly benefit from an expanded feature set and feature notation. S and SH are really at two distinct places in the mouth. So, the 'place' ANN may need to be more detailed regarding consonants. Furthermore, the affricates, such as CH, may need to have two 'place' values: fricative and occlusive. Nevertheless, the feature HMMs were able to recognise the broader equivalence class in Table 9 almost as well as the cepstral HMMs.

Having highlighted differences between feature and cepstral HMMs above, they also were similar. The feature HMMs had similar errors to the cepstral HMMs. Table 10 showed that both sets of HMMs made logical errors. That is, if they did not classify something correctly, they were likely to classify the phone as a related phone. This is in line with the discussion in Stephenson (1998); when neural networks would wrongly classify a frame, they sometimes classified the frame as a related feature value.

Finally, for future work I would recommend using a more precise feature set. While the vowels could use more precise features, I would first investigate looking into using a feature set that distinguishes more among the consonants. This is because in classifying phones, I believe that it is more important to be precise in labelling consonants than in labelling vowels. For example, labelling a /ih/ phone as an /ix/ does not produce a drastically different sounding word; but labelling a /ch/ phone as an /s/ is a greater change.

5 Acknowledgements

I would like to thank Simon King for his guidance in working with these HMMs and for his help in interpreting the results.

References

Garofolo, J. S. (1988). *Getting started with the DARPA TIMIT CD-ROM: An acoustic phonetic continuous speech database*. National Institute of Standards and Technology (NIST), Gaithersburgh, MD.

Rabiner, L. and Juang, B.-H. (1993). *Fundamentals of Speech Recognition*. Prentice Hall PTR.

Stephenson, T. A. (1998). Artificial neural networks in recognition of phonetic features in speech. M.Sc. Project.

Young, S., Jansen, J., Odell, J., Ollason, D., and Woodland, P. (1996). *HTK manual*. Entropic.