MIT-CSAIL-TR-2007-007 February 1, 2007
CBCL-266

Phonetic Classification Using Hierarchical, Feed-forward, Spectro-temporal Patch-based Architectures

Ryan Rifkin, Jake Bouvrie, Ken Schutte, Sharat Chikkerur, Minjoon Kouh, Tony Ezzat, and Tomaso Poggio

Phonetic Classification Using Hierarchical, Feed-forward, Spectro-temporal Patch-based Architectures

Ryan Rifkin
Honda Research Institute, Boston
rif@mit.edu

Jake Bouvrie
MIT Center for Biological and Computational Learning
MIT McGovern Center for Neuroscience
bouvrie@mit.edu

Ken Schutte
MIT Spoken Language Systems Group
MIT CSAIL
kschutte@csail.mit.edu

Sharat Chikkerur
MIT Center for Biological and Computational Learning
MIT McGovern Center for Neuroscience
sharat@mit.edu

Minjoon Kouh
MIT Center for Biological and Computational Learning
MIT McGovern Center for Neuroscience
kouh@mit.edu

Tony Ezzat
MIT Center for Biological and Computational Learning
MIT McGovern Center for Neuroscience
tonebone@mit.edu

Tomaso Poggio
MIT Center for Biological and Computational Learning
MIT McGovern Center for Neuroscience
tp@ai.mit.edu

Abstract

A preliminary set of experiments are described in which a biologically-inspired computer vision system (Serre, Wolf et al. 2005; Serre 2006; Serre, Oliva et al. 2006; Serre, Wolf et al. 2006) designed for visual object recognition was applied to the task of phonetic classification. During learning, the system processed 2-D wideband magnitude spectrograms directly as images, producing a set of 2-D spectro-temporal patch dictionaries at different spectro-temporal positions, orientations, scales, and of varying complexity. During testing, features were computed by comparing the stored patches with patches from novel spectrograms. Classification was performed using a regularized least squares classifier (Rifkin, Yeo et al. 2003; Rifkin, Schutte et al. 2007) trained on the features computed by the system. On a 20-class TIMIT vowel classification task, the model features achieved a best result of 58.74% error, compared to 48.57% error using state-of-the-art MFCC-based features trained using the same classifier. This suggests that hierarchical, feed-forward, spectro-temporal patch-based architectures may be useful for phonetic analysis.

I. Introduction & Motivation

Humans are adept at recognizing speech in a variety of noisy environments and in the presence of competing speakers (Allen 1994; Lippmann 1997). It has proven a very hard task to make computers do the same (Morgan, Zhu *et al.* 2005). The current paradigm of using mel-frequency cepstral coefficients (MFCCs) (Rabiner and Juang 1993) as speech features has so far failed to produce computer recognition levels that achieve human performance in a variety of speech-related tasks. Common characteristics of the MFCC representations are that they: 1) capture only aspects of the *spectral envelope*, and not the spectral frame itself; 2) are global, in that the final cepstral parameters represent the entire envelope, and not just local portions of it; and 3) are frame-based, in that they model spectral features that fall within the 10-30 msec window of time over which the MFCC features are computed.

A large body of recent work has attempted to develop feature representations which address one or more of these shortcomings. These attempts include: capturing dynamic information by including MFCC derivatives (Furui 1986); using subband frame MFCC features (Bourlard and Dupont 1997; Morris, Hagen *et al.* 1999); using long, thin time-slices of subband spectral activity (Hermansky and Sharma 1999; Hermansky 2003); smoothing the MFCCs in time (Greenberg and Kingsbury 1997; Kingsbury, N.Morgan et al. 1998); modeling the temporal envelope (Athineos and Ellis 2003; Athineos, Hermansky *et al.* 2004); extracting spectro-temporal edge features (Amit, Koloydenko et al. 2005); extracting temporal boundary features (Glass, Chang et al. 1996; Glass 2003); extracting localized spectro-temporal patterns (Kleinschmidt and Gelbart 2002; Kleinschmidt 2003)

In this work, we also seek to explore alternative approaches to speech recognition, by making recourse to recent progress made in computational visual neuroscience. Although the link between vision and audition has been made in the past (Mendelson and Cynader 1985; deCharms, Blake et al. 1998), engineering contributions from the visual domain have yet to be demonstrated for speech recognition. We argue that the recent advances in visual neuroscience and computer vision may in fact be useful for computational audition.

The first basic parallel between human vision and audition is in the input: sound is converted from a one-dimensional pressure wave to a two-dimensional pattern of neural activity, distributed over time along a tonotopic (frequency) axis (Chi, Ru *et al.* 2005). The two-dimensional pattern (frequency vs. time) constitutes a 2-D *auditory image* (Patterson, Robinson *et al.* 1992) which is presented by the auditory nerve to the auditory cortex for further processing. While the nature of the two input “images” is different, at a small scale local patterns within both the retinotopic grids and the auditory grids may be similar, and this suggests common cortical computational elements may be able to detect features in both visual and auditory images.

Work by a number of auditory neurophysiologists (Theunissen, Sen *et al.* 2000; Sen, Theunissen *et al.* 2001; Linden, R.C.Liu *et al.* 2003; Chi, Ru *et al.* 2005) indicates that there is a secondary level of analysis in the auditory cortex (AI), in which cells in AI analyze and process elements of the underlying input auditory image. Measurements of the so-called *spectro-temporal receptive fields* (STRFs) of cells in AI indicate that they can be tuned to different optimal frequencies, have different spectral scales, and also respond to different temporal rates. An analogy suggests itself between cells in this layer and the oriented Gabor-like filters of the first layer in the visual cortex (Hubel and Wiesel 1962; Hubel and Wiesel 1968). The cells at this stage seem to be responding to harmonics and/or spectral envelopes at different spectro-temporal frequencies, orientations, and time scales. This is the second basic parallel between human vision and audition at the biological level.

The parallel between vision and audition extends to a perceptual viewpoint (Bregman 1990; Amit, Koloydenko et al. 2005): We observe that humans are capable of recognizing specific visual objects despite variations in position, scale, rotation, and presence of clutter. As such, the visual brain forms a representation of visual objects that is invariant to a number of transformations that can be applied to that object. On the other hand, the representation of that object has to be specific enough to be able to differentiate one particular object from another. The visual brain learns to trade-off, in some sense, selectivity to certain input patterns and invariance to certain transformations.

Similarly, human speech perception can also be construed to be a form of auditory object recognition. Humans are capable of recognizing specific auditory objects such as phonemes despite variations in pitch, duration, location in time, and the presence of noise. As such, the brain also forms a representation of auditory objects that is invariant to a number of transformations that can be applied to that object. On the other hand, the representation of that object has to be specific enough to be able to differentiate one particular auditory object from another, as for example we do when we differentiate */but/* from */boot/*.

In this work, we seek to leverage our success over the past few years in building a biologically-inspired computer vision system that performs visual object recognition (Serre, Wolf et al. 2005; Serre 2006; Serre, Wolf et al. 2006). The system, which was designed to closely match the currently known architecture and physiology of the visual cortex, processes images in a feed-forward, hierarchical way producing a set of features at different positions, scales, orientations, and of varying complexity. The features computed by this system are then used as inputs to a classifier which is trained to determine the category of a particular object within that image. The system is capable of identifying the object in the image irrespective of variations in position, scale, orientation, and even in the presence of clutter. Recently this system has been shown to achieve human-level performance on rapid animal/non-animal categorization tasks.

Motivated by the success of our computer vision system in recognizing visual objects, we decided to explore whether the *same* 2-D, hierarchical, feed-forward architecture, coupled with a regularized least squares classifier, can be useful in a 20-class TIMIT vowel classification task. To judge the performance of our feature set more accurately, we compared classification performance of our model with a set of conventional state-of-the-art MFCC-based features (Halberstadt and Glass 1998) trained using the same classifier.

It is worth mentioning that recent work by (Domont, Heckmann et al. 2007) presents a system that is very similar to ours, in which a hierarchical system inspired from visual neuroscience is applied to the task of syllable recognition (specifically, their task is a 25 monosyllabic word recognition task, in which the words are embedded in various SNRs of babble noise). They report results in which their system outperforms Sphinx (Walker, Lamere et al. 2004) by 10-20% on word error rate for moderate to low SNR ratios.

In the following sections, we describe our task and data, our model architecture, and our experimental results in greater detail.

II. Task & Data

For our task, we chose a subset of the TIMIT phonetic classification corpus (Garofolo, Lamel *et al.* 1993). The classes were 20 vowel sounds /ay/, /ae/, /ow/, /ey/, /ao/, /uw/, /aa/, /ah/, /aw/, /ax/, /ax-h/, /axr/, /eh/, /er/, /ih/, /ix/, /iy/, /oy/, /uh/, and /ux/. Shown in Appendix 1 are the respective numbers of training, testing, and development examples for each phoneme (We describe the purpose of the development set in section III-D below).

For all sounds, a wideband, high frame-rate, short-time Fourier transform (STFT) is computed with 4ms hamming windows every 2ms, resulting in 500 frames per second. Each frame is zero-padded to 256 points, resulting in 129 point spectra. The STFT magnitude is smoothed with a single pole low-pass filter with time constant 8ms to remove effects of the pitch periods. The final spectrograms used are the log of this image, normalized to zero-mean, unit-variance. The images form the direct input to the model as will be described in the next section.

All sounds were time-normalized to 85 columns (170 msec), which is the median duration of the vowel classes under consideration. The matlab function *imresize()* was used to perform this time-normalization. At the completion of our pre-processing, all inputs sounds were thus 129 bins high and 85 bins wide.

Shown in Appendix 2 are example images of the spectrograms for each vowel class, before amplitude and time normalization.

III. Hierarchical, Feed-forward, Spectro-temporal Patch Based Model Architecture

A. Overview

The model consists of eight hierarchical layers of computational units composed of alternating simple S units and complex C units. In the next sections, we review the functionality of the S and C units, and then briefly describe each layer. Since our model architecture is exactly as the one described in (Serre, Kouh et al. 2005; Serre, Oliva et al. 2006) (with some minor modifications which are listed in Appendix 3) we only summarize the basic ideas, and refer the reader to (Serre, Oliva et al. 2006) for the details.

B. Simple and Complex Computational Units

There are two types of functional layers in the model: the S layers which are composed of *simple* units are interleaved with C layers which are composed of *complex* units.

Simple units in each S_k layer pool over afferent units from a topologically related local neighborhood in the previous C_{k-1} layer with different selectivities. As a result, the complexity of the preferred stimulus of units increases from layer C_{k-1} to S_k. The pooling operation at the S level is a Gaussian-like tuning function. That is, the response y of a simple unit receiving the pattern of x, is

$$y = \exp\left(-\frac{1}{2\sigma^2}\sum_{j=1}^{N}(w_j - x_j)^2\right) \qquad \textbf{Equation 1}$$

where σ defines the sharpness of the tuning around the preferred stimulus of the unit corresponding to the weight vector w. The response of the unit is thus maximal (y = 1) when the current pattern of input x matches exactly the synaptic weight vector w and decreases with a bell-shaped tuning profile as the pattern of input becomes more dissimilar (Note: Equation 1 is actually implemented by a more biologically-plausible tuning operation of the form of a normalized dot-product followed by a sigmoid function. See (Serre, Kouh et al. 2005) for more details).

Complex units in the C_k layer pool over afferent units from the previous S_k layer with the same selectivity but at slightly different positions and scales to increase the tolerance to 2D translation and scale transformations from layer S_k to C_k. The pooling operation at the complex C level is a MAX operation. The response y of a complex C unit thus corresponds to the response of the strongest of its afferents x from the previous S_k layer. An idealized mathematical description of the complex unit operation is given by

$$y = MAX(x_1 \dots x_N) \qquad \textbf{Equation 2}$$

(Note: In practice Equation 2 is implemented using a soft-max operation. See (Serre, Kouh et al. 2005) for more details).

C. Layered Architecture

S_1 units: The 2-D spectrograms are first analyzed by an array of simple hand-coded S_1 units which correspond to the classical simple cells of Hubel & Wiesel (Hubel and Wiesel 1962; Hubel and Wiesel 1968) found in primary visual cortex (V1). S_1 units take the form of Gabor functions (Daugman 1988; Lee 1996), which have been shown to provide a good model of cortical simple cell receptive fields. Intuitively, the S_1 units perform a localized, oriented-edge detection on the image, where each S_1 unit corresponds to a 2-D Gabor filter with a particular orientation, scale, and position. The population of units consists of 68 types of units: 4 orientations x 17 sizes. Shown in Figure 1 are the set of Gabor filters used at this layer. Our Gabors are even-symmetric and have amplitude normalized between 1 and -1.

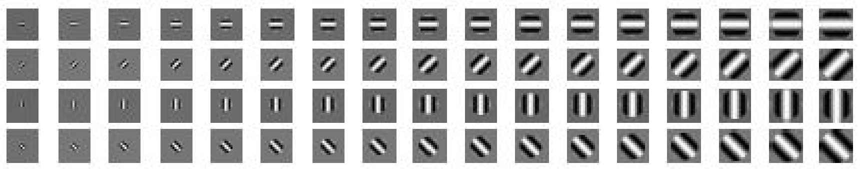


Figure 1: Gabor filters used at Layer S_1

C_1 units: The next C_1 level corresponds to striate complex cells (Hubel and Wiesel 1959). Each of the complex C_1 units receives the outputs of a group of simple S_1 units with the same preferred orientation but at slightly different positions and sizes (or peak frequencies). Each complex unit pools over its inputs using a MAX operation; i.e., the response of a complex unit corresponds to the response of the strongest of its afferents from the previous S_1 layer. The result of the pooling over positions is that C_1 units become insensitive to the location of the stimulus within their receptive fields, which is a hallmark of complex cells (Hubel and Wiesel 1959).

S_2 units: At the S_2 level, units pool the activities of 10 complex C_1 units at different preferred orientations in a local neighborhood via a tuning operation as in Equation 1. As a result, the complexity of the preferred stimuli is increased: At the C_1 level units are selective for single bars at a particular orientation, whereas at the S_2 level, units become selective to more complex patterns, such as the combination of oriented bars to form contours or boundary-conformations.

We define each S_2 unit as a *patch*, and the entire collection of S_2 units as a *patch dictionary*. The learning stage consists of setting the w weights for each of K patches within that layer. We describe below in Section D how patch "learning" (or development) occurs, ie, how the weights w are set for each patch, and also how the number K of patches is determined. Subsequently in Section E, we describe how the patch dictionaries are used during training and testing phases.

C_2 units: In the next C_2 stage, units pool over S_2 units that are tuned to the same preferred stimulus (they correspond to the same combination of C_1 units and therefore share the same weight vector w) but at slightly different positions and scales. C_2 units are therefore selective for the same stimulus as their afferents S_2 units. Yet they are less sensitive to the position and scale of the stimulus within their receptive field.

S_3 and C_3 stages: Beyond the S_2 and C_2 stages, the same process is iterated once more to increase the complexity of the preferred stimulus at the S_3 level, where the response of 100 C_2 units with different selectivities are combined with a tuning operation to yield even more complex selectivities. In the next stage, the complex C_3 units, obtained by pooling S_3 units with the same selectivity at neighboring positions and scales, are also selective to moderately complex features as the S_3 units, but with a larger range of invariance. The S_3 and C_3 layers provide a representation based on broadly tuned shape components. As in the S_2 layer, each S_3 unit is termed a patch, and the entire set of units a S_3 patch dictionary.

S_{2b} and C_{2b} stages: S_{2b} units (where "b" stands for "bypass") combine the response of several complex C_1 units at different orientations just like S_2 units. Yet their receptive field is larger (2 to 3 times larger) than the receptive fields of the S_2 units. Importantly, the number of afferents to the S_{2b} units is also larger (100 *vs.* 10), which results in units which are more "elaborate" than the S_2 units, yet, less tolerant to deformations. The C_{2b} is a final stage in that their outputs are not passed on to a higher stage.

D. Learning Stage

The learning stage determines the set of weight vectors w (see Eq. 1) for each patch within the patch dictionary in layers S_2, S_{2b} and S_3. Additionally, the learning stage determines the number of patches K for each S layer.

Learning in the model is *sequential up the hierarchy:* first the patch dictionaries at layers S_2/S_{2b} are constructed, followed by the construction of the patch dictionary at layer S_3. To construct each dictionary, the spectrograms in the development set are presented in random order, and features are propagated up the hierarchy up to the S_k layer of interest. At this point, the S_k dictionary is constructed. In the next iteration, the same development spectrograms are presented again in random order, but now the features are propagated to the next S_{k+1} layer. The dictionaries thus need to be created in sequential fashion, but this process only needs to be performed once for each layer.

During the patch dictionary construction process for layer S_k, the weights w of each of K patches are learned using an *imprinting* process: each unit stores in its synaptic weights w the current pattern of activity from its afferent inputs (from the previous layer) in response to the part of the spectrogram that falls within its receptive field. This is done by setting w to be equal to the current pattern of pre-synaptic activity x. As a result, the patch x that falls within the receptive field of the unit w becomes its preferred stimulus. Note that units in higher layers are thus tuned to larger patches.

We explored two different patch imprinting strategies: In one strategy (minimal average response), *a patch is only imprinted if the average value of its afferents is larger than some minimal value.* In another strategy (minimal max response), *a patch is only imprinted if the maximum value of its afferents is larger than some minimal value*. If a patch's afferents do not satisfy either the minimal average response or the minimal max response constraints, then that patch is not imprinted.

The number of K patches imprinted per S layer is determined as a heuristic tradeoff between the number of patches with minimal average/max afferent responses, and the number of features that the regularized least squares classification architecture can handle. Typically the number of patches with minimal average response is too large for a classifier to handle, so a heuristic pruning is performed in order to reduce their size to a more manageable number. Typically, the final number of patches retained in each dictionary ranges from 2000 to 4000. See Appendix 3 for more details for the size K at each S layer.

E. Training and Testing Stages

The training stage follows the development stage, and consists of presenting spectrograms from a training set to the model. In this stage the features are computed all the way up the hierarchy, since the S_2, S_{2b}, and S_3 dictionaries have already been created in the previous development stage.

It is important to note that the responses computed at the C layers form the input features to the classifier, not the responses at the S layers. Each C layers pools over its afferents in the previous layer using a MAX operation, thus producing either a *reduction or an expansion* of the previous S layer's dimensionality. A reduction or expansion can happen depending on the type of pooling operations defined for that C layer. For example, layers C_1 and C_{2b} are architected to reduce the dimensionality of the previous S_1 and S_{2b} layers, but layer C_2 expands the dimensionality of the previous S_2 layer. In all cases, however, the dimensionality of the feature inputs to the classifier are determined by the C layers.

The outputs of the C stages are used as the inputs to the regularized least squares classifier. The training stage also optimizes the parameters of the RLSC classifier (described in Secion IV below).

During testing, a separate set of spectrograms from a testing set are presented to the developed model and the trained classifier, and used to compute our final scores, described in Section V below.

IV. Linear Regularized Least Square Classifier

Linear regularized least squares is an instance of Tikhonov regularization (Poggio and Girosi 1990; Wahba 1990), a very general framework for learning that includes many common discriminative learning algorithms, including support vector machines (Vapnik 1995). The regularized least squares classification framework adopted in this work is presented in greater detail in (Rifkin, Yeo et al. 2003; Rifkin, Schutte et al. 2007), so we only summarize the relevant details here.

We are given a data set $\{(x_1, y_1), ..., (x_n, y_n)\}$ where the x_i represent points to be classified, while the y_i are the desired *labels*. In our case, each x_i will consist of a row of features from any one of the C layers ($C1$, C_2, C_{2b}, or C_3). The label y_i will consist of a label for the vowel class of that training data point. The data points are collected in a matrix X and the labels are collected in vector Y.

The linear regularized least squares problem is to find a vector w minimizing:

$$\frac{1}{2}\|Y - Xw\|_2^2 + \frac{\lambda}{2}\|w\|_2^2 \qquad \textbf{Equation 3}$$

where λ is a positive regularization parameter controlling the tradeoff between fitting the observations and finding a w with small norm, and L_2 norm is defined as $\|v\|_2 = \sqrt{v^T v}$. This is a differentiable, convex optimization problem, and straightforward calculus and linear algebra shows that the optimal w is given by:

$$w = (X^T X + \lambda I)^{-1} X^T Y \qquad \textbf{Equation 4}$$

λ is chosen as detailed in (Rifkin, Schutte et al. 2007) using an SVD procedure.

Given a multi-class data set, we train a binary RLS classifier for each pair of classes (for c classes, we train c(c − 1)/2 binary classifiers). For each pair of classes i and j (i < j), we train a binary classifier using the points in class i with y = 1 and the points in class j with y = −1. Given a new test point, we threshold the outputs of all the classifiers at 0, forcing each classifier to make a hard vote for one of the two classes it was trained on. We then classify the example into the class that received the most votes. In the case of ties, we restrict our attention to the k classes that received the most votes and recount votes from only the k(k − 1)/2 classifiers on these classes (in the simple case of a two-way tie between classes i and j, we choose in accord with the i-vs-j classifier). If a tie remains after this restriction, we pick the class with the highest prior (most training examples) from among the classes receiving the most votes.

V. Experiments & Results

We performed 12 classification experiments in total, 10 using the dictionaries learned by our model from labeled examples, and 2 using comparison feature sets which we term the "MFCC" and the "Raw Spectral" feature sets. Both comparison feature sets are described in more detail in Appendix 4.

Five of the experiments using our model features consisted of using as input to the classifier either the C_1, C_2, C_{2b}, C_3, or a concatenation of the full C_1+C_2+C_{2b}+C_3 features, where the dictionaries are computed using a minimal average response patch imprinting strategy. These results are shown in Table 1.

Another four of the experiments using our model features consisted of using as input to the classifier either the C_1, C_2, C_{2b}, C_3, where the dictionaries are computed using a minimal maximum response patch imprinting strategy. These results are shown in Table 2.

The last of the experiments using our model feature set consisted of using only C_1 features obtained from a slightly re-architected Gabor filter set. This experiment was a follow-up experiment designed to identify the influence of the type of Gabor filters on performance.This result is shown in Table 3. Details of the re-architected Gabors are described in Appendix 3.

Finally, the results using the comparison feature sets are shown in Table 4.

In all experiments, an RLS classifier was trained using the method described in Section IV. Training and testing sets were identical in all experiments. The development set was only used in order to build the patch dictionaries.

Feature Layer	Feature Dimensionality	Classification Error
C_1 model features	8252	59.68%
C_2 model features	10,000	67.32%
C_{2b} model features	2000	63.09%
C_3 model features	4000	76.51%
C_1+C_2+C_{2b}+C_3 model features	24252	59.85%

Table 1: The model feature experiments using a minimal average response patch imprinting strategy.

Feature Layer	Feature Dimensionality	Classification Error
C_1 model features	8252	59.68%
C_2 model features	10,000	62.32%
C_{2b} model features	4000	65.91%
C_3 model features	4000	72.36%

Table 2: The model feature experiments using a minimal maximum response patch imprinting strategy.

Feature Layer	Feature Dimensionality	Classification Error
C_1 model features	7932	58.74%

Table 3: The model feature experiments using a re-architected Gabor filter set.

Feature Layer	Feature Dimensionality	Classification Error
“MFCC” features	61	48.57%
“Raw Spectral” features	646	51.26%

Table 4: The comparison “MFCC” and “Raw Spectral” feature experiments.

VI. Discussion

Our best result using the model features is using only the C_1 layer, which yields either 58.74% or 59.68% classification error depending on the Gabor filter set we used. This compares favorably with the error rates using the comparison “MFCC” and “Raw Spectral” features, which are 48.57% and 51.26% respectively. However, the feature dimensionality in our C1 layer increases dramatically to 8252, as compared to 61 and 646 in the “MFCC” and “Raw Spectral” cases respectively.

Our experiments with C_2, C_{2b}, and C_3 features indicated that nothing was being gained by adding these additional layers, and performance in fact degraded when those features were added. There are two possible reasons for this:

Firstly, it might mean that the spectrograms themselves do not contain much information beyond the S_1-C_1 scale.

Alternatively (and more likely), it might mean that the construction of the S-layer dictionaries and/or the pooling of the C layers is being performed in a detrimental manner.

This leads to the following set of questions which are avenues of future exploration:

1. The model as it was architected in (Serre, Oliva et al. 2006) was designed with position- and scale-invariance in mind, which are two of the most common types of invariances found in human vision.

What are the analogous invariances in audition, and how may these be appropriately coded for in a re-architected version of the model?

2. What are the appropriate number of layers to use in the model for audition? Is the S1-C1 layer sufficient, or do we need higher layers to account for more complex selectivities in the patch dictionaries?

3. The current inputs to our model are wideband spectrograms which effectively represent the spectral envelopes in speech. What about speech harmonics? Can a lower level of harmonic detail be useful at all for recognition? If so, how would the model perform on raw spectrograms instead of envelopes? Also, how would a "tandem" approach perform, where tandem hierarchical models process the harmonics layer and envelope layers separately, and then combined?

4. How would the model features perform in classifying phonemes in the presence of noise?

5. How would the model features perform in classifying other types of phonemes besides vowels?

VII. Conclusion

We described a set of preliminary experiments in which a biologically-inspired computer vision system (Serre, Wolf et al. 2005; Serre 2006; Serre, Oliva et al. 2006; Serre, Wolf et al. 2006) designed for visual object recognition was applied to the task of phonetic classification. On a 20-class TIMIT vowel classification task, the model features achieved a best result of 58.74% error, compared to 48.57% error using state-of-the-art MFCC-based features trained using the same classifier. This suggests that hierarchical, feed-forward, spectro-temporal patch-based architectures may be useful for phonetic analysis.

VIII. Acknowledgments

The authors would like to thank Thomas Serre for valuable comments and suggestions.

References

Allen, J. (1994). "How do humans process and recognize speech?" IEEE. Trans. on Speech and Audio Processing 2(4): 567-577.

Amit, Y., A. Koloydenko, et al. (2005). "Robust Acoustic Object Detection." Journal of the Acoustical Society of America 118(4): 2634-2648.

Athineos, M. and D. P. W. Ellis (2003). Frequency Domain Linear Prediction for Temporal Features. Proc. ASRU Worskshop.

Athineos, M., H. Hermansky, et al. (2004). LP-TRAP: Linear Predictive temporal patterns. Proc. ICSLP.

Bourlard, H. and S. Dupont (1997). Subband-Based Speech Recognition. Proc. ICASSP, Munich, Germany.

Bregman, A. (1990). Auditory Scene Analysis. Cambridge, MA.

Chi, T., P. Ru, et al. (2005). "Multiresolution Spectrotemporal Analysis of Complex Sounds." Journal of the Acoustical Society of America 118: 887-906.

Daugman, J. G. (1988). "Complete discrete 2-D Gabor transforms by neural networks for image analysis and compression." IEEE Trans. Acoustics, Speech, and Signal Processing 36: 1169.

deCharms, R. C., D. T. Blake, et al. (1998). "Optimizing Sound Features for Cortical Neurons." Science 1439.

Domont, X., M. Heckmann, et al. (2007). A Hierarchical Model for Syllable Recognition. ESANN, to appear, Bruges, Belgium.

Furui, S. (1986). Speaker-Independent Isolated Word Recognition Using Dynamic Features of Speech Spectrum. IEEE Trans. Acoustics, Speech, and Signal Processing 34: 52--59.

Garofolo, J. S., L. F. Lamel, et al. (1993). TIMIT Acoustic-Phonetic Continuous Speech Corpus. Philadelphia, Linguistic Data Consortium.

Glass, J. R. (2003). "A probabilistic framework for segment-based speech recognition." Computers, Speech, and Language 13: 137.

Glass, J. R., J. Chang, et al. (1996). A probabilistic framework for feature-based speech recognition. Proc. ICSLP.

Greenberg, S. and B. E. D. Kingsbury (1997). The modulation spectrogram: in pursuit of an invariant representation of speech. Proc. ICASSP, Munich, Germany.

Halberstadt, A. (1998). Heterogeneous Acoustic Measurements and Multiple Classifiers for Speech Recognition. EECS. Cambridge, MA, MIT. PhD Thesis.

Halberstadt, A. and J. Glass (1998). Heterogeneous Measurements and Multiple Classifiers for Speech Recognition. Proceedings of ICSLP.

Hermansky, H. (2003). Data-driven Extraction of Temporal Features from Speech. Proc. ASRU Worshop.

Hermansky, H. and S. Sharma (1999). Temporal Patterns (TRAPS) in ASR of Noisy Speech. Proc. ICASSP.

Hubel, D. and T. Wiesel (1959). "Receptive fields of single neurons in the cat's striate cortex." Journal of Physiology (London) 148: 574-591.

Hubel, D. and T. Wiesel (1962). Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. J. Phys. 160: 106--54.

Hubel, D. H. and T. N. Wiesel (1968). Receptive fields and functional architecture of monkey striate cortex. J.Phys. 195: 215--243.

Kingsbury, B. E. D., N.Morgan, et al. (1998). " Robust speech recognition using the modulation spectrogram." Speech Communication 25(1): 117-132.

Kleinschmidt, M. (2003). Localized Spectro-temporal Features for Automatic Speech Recognition. Proc. Eurospeech.

Kleinschmidt, M. and D. Gelbart (2002). Improving Word Accuracy with Gabor Feature Extraction. Proc. ICSLP.

Lee, T. S. (1996). Image Representation Using 2-D Gabor Wavelets. IEEE Trans. on Pattern Analysis and Machine Intelligence 18: 959.

Linden, J. F., R.C.Liu, et al. (2003). "Spectrotemporal Structure of Receptive Fields in Areas AI and AAF of Mouse Auditory Cortex." Journal of Neurophysiology 2660.

Lippmann, R. P. (1997). "Speech recognition by machines and humans." Speech Communication 22: 1--15.

Mendelson, J. R. and M. S. Cynader (1985). "Sensitivity of cat primary auditory cortex (AI) to the direction and rate of frequency modulation." Brain Research 327(1).

Morgan, N., Q. Zhu, et al. (2005). "Pushing the Envelope - Aside." IEEE Signal Processing Magazine 22(5): 1053-1088.

Morris, A., A. Hagen, et al. (1999). The Full Combination Sub-bands Approach to Noise Robust HMM/ANN-based ASR. Proc. Eurospeech 99.

Patterson, R. D., K. Robinson, et al. (1992). Complex Sounds and Auditory Images. Proc. Ninth Intl Symp on Hearing.

Poggio, T. and F. Girosi (1990). Networks for Approximation and Learning. Proceedings of the IEEE 78: 1481--1497.

Rabiner, L. and B. H. Juang (1993). Fundamentals of Speech Recognition. Englewood Cliffs, NJ.

Rifkin, R., K. Schutte, et al. (2007). Noise Robust Phonetic Classification with Linear Regularized Least Squares and Second-Order Features. Proc. ICASSP 2007, under submission.

Rifkin, R., G. Yeo, et al. (2003). Regularized Least Squares Classification. Advances in Learning Theory: Methods, Models, and Applications. H. B. M. V. L. R. Stuykens.

Sen, K., F. Theunissen, et al. (2001). Feature Analysis of Natural Sounds in the Songbird Auditory Forebrain. Journal of Neurophysiology 86: 1445.

Serre, T. (2006). Learning a dictionary of shape components in visual cortex:Comparisons with neurons, humans, and machines. Brain and Cognitive Science. Cambridge, MA, MIT. PhD Thesis.

Serre, T., M. Kouh, et al. (2005). A theory of object recognition: computations and circuits in the feedforward path of the ventral stream in primate visual cortex. Cambridge, MIT: CBCL Paper #259/AI Memo #2005-036.

Serre, T., A. Oliva, et al. (2006). A Feedforward Theory of Visual Cortex Accounts for Human Performance in Rapid Categorization. Submitted.

Serre, T., A. Oliva, et al. (2006). A Feedforward Theory of Visual Cortex Accounts for Human Performance in Rapid Categorization,. CBCL Paper # MMVI-02. Cambridge, MA, Massachusetts Institute of Technology.

Serre, T., L. Wolf, et al. (2006). "Object Recognition with Cortex-Like Mechanisms." IEEE Trans. on Pattern Analysis and Machine Intelligence

Serre, T., L. Wolf, et al. (2005). Object Recognition with features inspired by visual cortex. Proceedings IEEE Conf. on Computer Vision and Pattern Recognition.

Theunissen, F., K. Sen, et al. (2000). "Spectral-Temporal Receptive Fields of Nonlinear Auditory Neurons Obtained Using Natural Sounds." Journal of Neuroscience 20(6): 2315.

Vapnik, V. (1995). The Nature of Statistical Learning Theory. New York, NY.

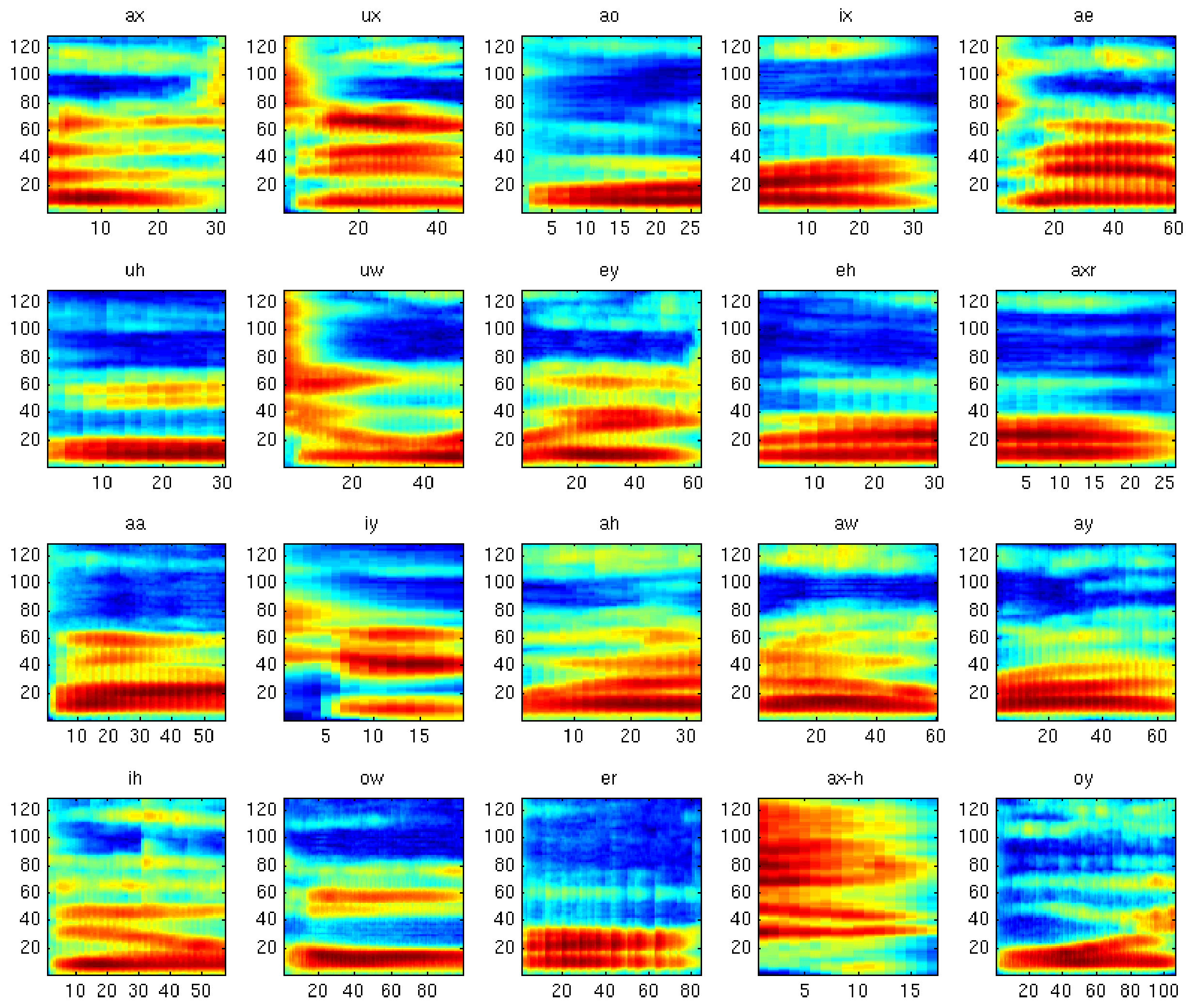
Wahba, G. (1990). Splines Models for Observational Data. Philadelphia.

Walker, W., P. Lamere, et al. (2004). Sphinx-4: A flexible open source framework for speech recognition. Technical Report, Sun Microsystems.

Appendix 1: Training, Testing, and Development Set Sizes

Phonetic Label	TEST SET	TRAINING SET	DEV SET
aa	131	1000	236
ae	105	1000	237
ah	135	1001	248
ao	97	1000	223
aw	30	728	61
ax	172	1000	366
ax-h	14	357	37
axr	141	1000	321
ay	89	1000	214
eh	189	1000	391
er	93	1000	193
ey	114	1000	230
ih	203	1000	444
ix	377	1004	757
iy	243	1000	564
ow	89	1000	168
oy	16	304	30
uh	29	500	54
uw	22	529	48
ux	52	1001	116

Appendix 2: Example Spectrogram Inputs



Examples of spectrograms for the 20 vowel classes, before time and amplitude normalization.

Appendix 3: Modifications to the Parameters of the Model

The majority of the model parameters used in the experiments presented in this memo were identical to those chosen by (Serre, Oliva et al. 2006). The particular instances where we modified model settings from Serre and Oliva's "defaults" are noted as follows:

a) The sigmoid nonlinearity applied to normalized dot-products at the S layers takes the form $g(z) = (1 + \exp(-a(z-b)))^{-1}$. At layers S2 and S3 the sigmoid's "a" parameter was changed from a default of 20 to 8.
b) Layer S2b was configured to look at only the top (largest) 4 scales from C1, rather than the top 5 scales, while layer S3 was configured to look at only the top 2 scales from C2, rather than the top 3 scales.
c) Layer S3 was configured to examine 2x2 spatial blocks at the above C2 scales, rather than blocks of size 3x3.

Modification (a) was made to accommodate extremely small normalized dot-product responses, wile modifications (b) and (c) were made to adjust for smaller input stimuli (spectrograms).

For the experiments described in Table 1, the size of the patch dictionaries learned at layers S2, S2b, and S3 were 2000, 2000, and 4000 respectively. Because the size of the receptive fields at layers S2b and S3 are large compared to the size of the input stimuli, "local" pooling is effectively global and the dimensionality of the model's output at layers C2b and C3 is equal to the size of the patch dictionary. Receptive fields are still small at layer C2, however, and the dimensionality of the output is much larger than the number of patches in the S2/C2 dictionary.

For the experiments described in Tables 2 and 3, the patch dictionaries were of size 4000 at all learnable S-layers (S2, S2b, and S3).

The experiments presented in Table 3 reflect three additional modifications to the S1 and C1 layers. In particular, we first changed the set of Gabor filter orientation angles from {0,45,90,-45} to {0,10,20,90,-10,-20} (note that we have modified both the number of filters and the choice of angles). We have found that the majority of vowel phonemes in the TIMIT corpus involve formants which slope up- or down-wards at an angle between 0 and 20 degrees, assuming spectrograms which span a time-frequency region of size 8kHz by 170ms. A shift from 45 degrees to 20 degrees is thus equivalent to shifting our preferences from formants sloping at 47 Hz/ms to those which slope somewhere near 19 Hz/ms. Second, because it is usually the case that information at high-frequencies in vowels does not provide much discriminative power, we additionally eliminated all frequency bins above 5kHz, giving a spectrogram "image" of size 80x85 rather than 129x85. This modification simplifies the learning problem by eliminating non-informative features, and also reduces the computation time needed to train and test the model. The final change to the model involved a shifting of the spatial pooling sizes at layer C1. The default configuration used by (Serre, Oliva et al. 2006) called for spatial pooling over blocks with edges of length {8,10,12,14,16,18,20,22}. Visual inspection of Gabor filtered spectrograms revealed that there may be important information at a resolution higher than that given by 8x8 pooling. We therefore chose to shift the entire spatial pooling range down by 4, giving a set of pooling resolutions equal to {4,6,8,10,12,14,16,18}. Note however that pooling at C1 involves combining information from multiple S1 features, and thus does not refer to blocks in the original image. An 8x8 pool of S1 features therefore corresponds to a receptive field which looks at an area of the original image much larger than 8x8.

Appendix 4: “MFCC” and “Raw Spectral” Features

The “MFCC” features used are the ``S2'' features from (Halberstadt 1998). Short-time Fourier analysis is done with a 30ms Hamming window every 5ms. For each frame, we compute 12 Mel frequency cepstral coefficients (MFCCs). To get a fixed-length feature vector for each phonetic segment, the MFCCs are averaged over five regions: the 30ms before and after the segment, and three regions within the segment (in 3-4-3 proportion). The log duration is also included, giving a total of 5*12+1=61 dimensions. These features are then whitened using a principle component analysis (PCA) matrix derived from the training set.

MFCC’s are computed in the traditional way: Over each spectral slice, a bank of triangular filters spaced according to the mel-frequency scale is applied. The log-energy under each filter is retained, and the resulting vector (typically approximately 40 dimensions) is referred to as a set of MFSCs, Mel-Frequency Spectral Coefficients. In each frame, a discrete cosine transform (DCT) of the MFSCs is taken, and only the first 12 coefficients are retained.

The “Raw Spectral” features are computed on the same spectrograms that were used as input to the model (before time and amplitude normalization) was applied, and using the same time-averaging technique as in the “MFCC” features: the 30ms before and after the segment, and three regions within the segment (in 3-4-3 proportion). The log duration is also included, giving a total of 5*129+1=646 dimensions. No PCA was performed on the “Raw Spectral” features.