DYNAMICS OF TONGUE GESTURES EXTRACTED AUTOMATICALLY FROM ULTRASOUND

Jeff Berry & Ian Fasel
University of Arizona
Tucson, AZ 85721

ABSTRACT

We describe a system for automatically extracting dynamics of tongue gestures from ultrasound images of the tongue using translational deep belief networks (tDBNs). In tDBNs, a joint model of the input and output vectors are learned during a generative pretraining stage, and then a translation step is used to transform input-only vectors into this joint representation. A final fine-tuning stage is then used to reconstruct the desired outputs given input vectors. We show that this technique dramatically improves performance on segmenting ultrasound image sequences of continuous speech into individual consonant gestures compared with the original DBN method of [1] as well as alternative methods using PCA and support vector machines.

Index Terms—Deep Belief Networks, Ultrasound

1. INTRODUCTION

Speech scientists have recently made advances in the understanding of speech production using new methods for imaging the vocal tract, which allow the dynamics of organs such as the tongue and velum to be seen during speech [2]. Currently, analysis of dynamics of tongue gestures is difficult because it is expensive to hand-label each image. Linguists often use the extracted shape of the tongue surface to study differences in speech sound [3]. Using state-of-the-art active-contour “snakes” based computer assisted tracing methods [4], extracting the tongue surface of a single second of 30fps video can take over 10 minutes by a trained expert. Once traces have been extracted, further time is needed to manually isolate and extract gestural peaks for analysis. To overcome this bottleneck in the processing of vocal tract images, we propose a method to automatically extract tongue gestures and tongue surface traces using deep belief networks (DBNs).

DBNs are probabilistic generative models composed of multiple restricted Boltzmann machines (RBMs) stacked on top of each other. Each RBM consists of a visible input layer and a hidden layer of binary random variables, and can be efficiently trained using the contrastive divergence algorithm [1]. DBNs are typically trained in two stages: first a generative stage, in which the training data without labels is used to find the weights of the interlayer connections, and second a discriminative stage, in which the labels are used to fine-tune the network using backpropagation.

In this work we explore an alternative method for training DBNs, in which labels are used in both the generative and discriminative stages. This is similar to the idea of [5, 6, 1] in that the labels are present at every stage of training, however in the tDBN, inference is easily performed in a single feedforward pass through the network. The idea is to perform the standard DBN pre-training using concatenated sensor and label vectors, but then swap out the first RBM of the network with a new one that is trained to map sensor-only data to the hidden representation learned by the RBM trained on the concatenated data. With this strategy the representation learned by the middle layers of the network is influenced by the labels even before any fine tuning occurs. We show that this technique can be used to segment consonants and extract contours from ultrasound images of the tongue, resulting in dramatic gains in performance over the original deep net technique as well as alternative PCA and support vector machine-based methods.

2. PRELIMINARIES

We begin by describing DBNs and autoencoders, largely recapitulating [1, 5]. An autoencoder is a system which takes an input vector $x$ and maps it to a hidden representation $h$ using an encoder network, followed by a decoder network which reconstructs the input from the hidden representation. A common way of constructing an autoencoder is through restricted Boltzmann machines (RBMs). An RBM is a specific type of Markov random field in which the $V$ dimensional input vector $x$ and $H$ dimensional hidden feature vector $h$ are modeled by products of conditional Bernoulli distributions:

$$ f_{\phi, j}(x) = p(h_j = 1 \mid x; \phi) = \sigma \left( \sum_{i=1}^{V} W_{i,j} x_i + a_j \right) $$

$$ f'_{\phi, i}(h) = p(x_i = 1 \mid h; \phi) = \sigma \left( \sum_{j=1}^{H} W_{i,j} h_j + b_i \right) $$
where $\sigma(u) = (1 + e^{-u})^{-1}$, and $\phi = (W, a, b)$. Thus when used as an autoencoder, $f_\phi$ is the encoder network and $f'_\phi$ is the decoder network, using shared parameters $\phi$. The marginal distribution over visible vector $x$ is:

$$p(x; \phi) = \frac{\sum_h e^{-E(x, h; \phi)}}{\sum_u \sum_h e^{-E(u, h; \phi)}}$$  \hspace{1cm} (1)

where $E$ is the energy of the joint configuration $(x, h)$:

$$E(x, h; \phi) = \sum_{i=1}^{V} \sum_{j=1}^{H} W_{i,j} x_i h_j + \sum_{i=1}^{V} b_i x_i + \sum_{j=1}^{H} a_j h_j$$

Parameters $\phi$ can be estimated by taking gradient steps determined by the contrastive divergence (CD) rule [1], which yields parameter updates:

$$\Delta w_{i,j} = \langle x_i h_j \rangle_{data} - \langle \tilde{x}_i \tilde{h}_j \rangle$$  \hspace{1cm} (2)

$$\Delta a_i = \langle x_i \rangle_{data} - \langle \tilde{x}_i \rangle, \quad \Delta b_j = \langle h_j \rangle_{data} - \langle \tilde{h}_j \rangle$$

where expectation $\langle x_i h_j \rangle_{data}$ is the frequency that $x_i$ and $h_i$ are on together in the training set, and $\langle \tilde{x}_i \tilde{h}_j \rangle$ is the frequency that $\tilde{x}_i$ and $\tilde{h}_j$, which represent samples of $x_i$ and $h_i$ obtained by running the Gibbs sampler initialized at the data for one step, are on together. Thus one gradient update involves first sampling $h$ from $f_\phi(x)$, then generating the “reconstructions” $\tilde{x}$ by sampling from $f'_\phi(\tilde{h})$, and sampling new features $h$ from $f_\phi(\tilde{x})$, giving all the terms needed to compute (2). Details of this procedure can be found in [1]. When inputs are real-valued, Gaussian units can be used [7]. Provided each input dimension $x_i$ is normalized to zero mean and unit variance, the only change needed to the above equations is:

$$f'_{\phi}(h) = p(x_i = x_i | h; \phi) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}((x_i - b_i - \sum_{j=1}^{H} W_{i,j} h_j)^2)}$$

In most cases, this will converge to a maximum likelihood solution [8].

### 3. TRANSLATIONAL DEEP BELIEF NETS

A sequence of $N$ RBMs (with parameters for the $i$th RBM written $\phi_i$) can be trained to form a deep belief network (DBN). First, an RBM is trained on the input data, then for each datapoint in the training set, the hidden layer activations of the RBM for that datapoint (i.e., the outputs of encoder network $f_{\phi_i}$) are treated as a training input for a higher-level RBM [1]. This procedure is repeated $N$ times.

The resulting DBN can then be used in a variety of ways. For instance, the DBN can be used as an autoencoder at runtime, in which case data is run through each encoder function $f_{\phi_i}$ in sequence, and the top-layer hidden activations are then run through the decoder functions $f'_{\phi_i}$ in reverse order. Alternatively, the activations of encoder portions of the DBN can be fed into a final layer of classification units in order to make categorical decisions about the inputs. In both cases, the generative “pre-training” of the stack of RBMs is first used, forming a representation in the hidden weights that preserves information about the relationships between the input variables, and then a supervised gradient descent procedure is used to discriminatively “fine-tune” the network for a task.

In this work, our intuition is that for many tasks, the generative pre-training stage can be just as helpful in preserving the structural relationships in the target outputs and between targets and inputs as it is for discovering structure in just the inputs. Therefore, instead of training a deep net on the inputs only and then fine-tuning to predict the outputs, we train the deep net on concatenated sensor-label inputs (i.e., the inputs to the first layer RBM $x = (v, l)$), thus forcing the network to learn a representation that captures relationships between both the inputs (sensors) and targets (labels). We then face the task of learning how to predict the labels $l$ at runtime from only sensor data $v$ using this shared representation.

We do this using a “translational” RBM ($t$RBM). First, a full DBN is trained on concatenated sensor-label data $x = (v, l)$, with bottom-layer RBM parameters $\phi_1$. A new set of parameters $\theta$ are then created for a special RBM (the tRBM) which takes as input sensor-only data $v$, and has the same number of hidden units as the $\phi_1$ RBM. Let the encoder and decoder functions for the tRBM be denoted $g_\theta$ and $g'_\theta$. A new set of parameters $\theta$ are then created for a special RBM (the tRBM) which takes as input sensor-only data $v$, and has the same number of hidden units as the $\phi_1$ RBM. Let the encoder and decoder functions for the tRBM be denoted $g_\theta$ and $g'_\theta$. The tRBM is now trained with CD, but with a twist: for each gradient step, instead of training the term $h$ on the left hand side of eq. (2) from $g_\theta(v)$, we instead sample from $f_{\phi_i}(v, l)$. Then sensor reconstructions $\tilde{v}$ are sampled from $g'_\theta(h)$, and finally $h$ are sampled from $g_\theta(\tilde{v})$. Learning the tRBM is otherwise identical to normal RBM learning with CD. The original RBM in the DBN is then replaced with the tRBM to form a tDBN. The final step is then to perform fine-tuning, as in standard DBNs, for the specific task. For this paper, the tDBN was configured both as an autoencoder to predict contours (as in [9]), and as depicted in Fig. 1 to predict category labels.

### 4. EXPERIMENTS: TONGUE GESTURE DYNAMICS

We used the tDBN to classify tongue shapes into consonant phonemes with markedly different tongue shapes. Due to the dynamic nature of speech, choosing how to label any given ultrasound image can be challenging. For example, in consonant clusters, such as the “str” in the word “street”, it is difficult to determine where the /s/ begins and the /t/ begins, or where the /t/ begins, etc., since phonemes in speech do not act like a sequence of steady-states, but are rather blended together. In most cases, however, it is possible to determine when the tongue reaches the peak of the articulation of a phoneme, although the shape may be distorted due to the surrounding context. Therefore, our approach is to train the network only using peaks, and then test test how well the tDBN can represent the dynamics on a sequence.

The data set contained 1,893 ultrasound tongue images
from two native English speakers. These speakers read nonsense words containing the sounds /p, t, k, r, l/, such as [pataka]. Training images were selected by manually choosing video frames that best represented the peak for each phoneme of the word, resulting in images with one of the 5 labels. A sixth “none” category was formed by randomly selecting images that were at least 5 frames away from a manually labeled phoneme peak. The input to the network consisted of a central region cropped from each ultrasound image and scaled to 19 × 34 pixels, and the output was the consonant label.

We also used tDBNs for extracting tongue contours, as in [9]. In this setting, the inputs are still ultrasound images, but the labels are the human traces of the tongue contour, represented as images. The tDBN was configured as an autoencoder, so that the tongue contours are reconstructed using the fφ decoder functions (which were then fine-tuned with backprop). This network was trained on a different database of ultrasound images than those used for training the consonant recognition network. This dataset consisted of 10,437 images from 9 speakers, including 7 English speakers, 1 Georgian speaker, and 1 French speaker. The 2 English speakers who were in the [pataka] database were not included in this set. All speakers read words with /l/ sounds in their respective languages, as part of a separate cross-linguistic study on /l/ sounds, and each image was hand-traced to extract the contour shape.

5. RESULTS

Our first experiment compared standard DBNs and tDBNs for phoneme classification. We first trained a standard DBN of size 646-646-646-3230-6 nodes (the inputs were the 34 × 19 = 646 pixel ultrasound images). After pretraining, the 6-category labels were used to tune the DBN with backpropagation. Five-fold cross-validation gave an average accuracy of 42.99%, with 44.44% best accuracy. We repeated this for a network of size 646-1096-1096-5480-6, with similar cross validation results of 42.10% and best score of 44.06%.

We then trained a tDBN of size 1096-1096-1096-5480-6 nodes. The extra 450 nodes on the first visible layer are because the ultrasound was concatenated with the the 6-category label (encoded as one-vs-all, e.g. if image a was category 2, the label would be [0 1 0 0 0 0]), replicated 75 times so that the ultrasound and labels had similar size. After pretraining the DBN, a tRBM of size 646-1096 was trained using only the pixel values as inputs. This tRBM was then swapped for the first RBM of the pretrained DBN. Finally, the labels were used to fine tune the network using backpropagation. The same five-fold cross-validation method was used for measuring performance. The final tDBN showed a dramatic improvement over the standard DBN, with average accuracy of 86.48% and a high score of 87.86%. To test whether the difference in performance was due to the tRBM or the difference in layer sizes, we also trained a tDBN of size 1096-646-646-3230-6, resulting in an average accuracy of 86.64% and a 88.39% best score. The results show that the difference in performance in classification is due to the use of the tRBM in training, rather than differences in layer sizes.

We further tested performance of the tDBN on its ability to correctly classify sequences of images. Any sequence of tongue images includes tongue shapes that do not exhibit a phonemic peak, since the tongue must move from peak to peak during speech. Since the tDBN trained only on images containing phonemic peaks, testing the network on a sequence tests its ability to deal with transitional tongue shapes and the dynamics of tongue gestures during speech.

Figure 2 shows the activation for each of the 6 categories /p, t, k, r, l/ and ‘none’ during the utterance [pataka]. The tongue shapes for the vowel /a/ are classified as /p/. This is due to the lack of /a/ examples in the training data. The fact that /a/ is consistently classified as /p/ is itself interesting, and seems to support the idea that /p/ has no lingual shape specification, only labial shape specification [10]. Since /p/ does not have a large effect on the shape of the tongue, the tongue shape anticipates the following vowel, which in this case is an /a/. Since most of /p/s in the training data are followed by /a/, the tongue shape learned for /p/ is actually that for /a/, and /a/s following other consonants are therefore classified as /p/, as shown in figure 2.

5.1. Comparison to SVMs

We compared the tDBN method of classification to other popular methods using support vector machines (SVMs). We chose two different types of feature extraction techniques for comparison. The first method consisted of using a trace of the tongue surface manually extracted from the image. Each
Fig. 2. The 1096-646-646-3230-6 tDBN’s activation for a sequence of 36 tongue images during the utterance of the sounds [pataka].

The tongue trace was sampled at 50 points resulting in a 100 unit input for the SVM, and LIBSVM [11] was used for training and testing. This resulted in 70.36% accuracy on a holdout set after optimizing the SVM parameters on a training set.

The second method of feature extraction used principal components, inspired by Turk and Pentland’s [12] eigenfaces approach, which [13] have shown is effective for ultrasound feature extraction for speech recognition purposes. We used the coefficients of the first 100 eigenvectors as input to an SVM, resulting in 53.60% after optimizing the SVM.

In addition to its superior performance for classification, an advantage of the tDBN is its flexibility to be used for structured outputs like tongue contours. In [9], the tongue contour extraction method was able to achieve mean sum of distances (MSD) from the ground truth label of $2.5443 \pm 0.056$ pixels, equivalent to human inter-labeler agreement, on a holdout set from the same database. Here we test the ability to generalize by testing the network, trained on /l/ data, on the [pataka] sequence. The results are shown in figure 3. In general the traces are accurate, with the exception of the /l/ shape images, shown in the second image of figure 3. This is due to the sparsity of /l/ in the training data, which was not balanced to include examples of all phonemes. On the other hand, the performance of the network on shapes that were well represented in the training data shows promise for the ability of the network to generalize across speakers.

6. REFERENCES


