DEEP-LEVEL ACOUSTIC-TO-ARTICULATORY MAPPING FOR DBN-HMM BASED PHONE RECOGNITION

Leonardo Badino, Claudia Canevari, Luciano Fadiga, Giorgio Metta*

Istituto Italiano di Tecnologia
RBCS
Genova, Italy

ABSTRACT

In this paper we experiment with methods based on Deep Belief Networks (DBNs) to recover measured articulatory data from speech acoustics. Our acoustic-to-articulatory mapping (AAM) processes go through multi-layered and hierarchical (i.e., deep) representations of the acoustic and the articulatory domains obtained through unsupervised learning of DBNs. The unsupervised learning of DBNs can serve two purposes: (i) pre-training of the Multi-layer Perceptrons that perform AAM; (ii) transformation of the articulatory domain that is recovered from acoustics through AAM. The recovered articulatory features are combined with MFCCs to compute phone posteriors for phone recognition. Tested on the MOCHA-TIMIT corpus, the recovered articulatory features, when combined with MFCCs, lead to up to a remarkable 16.6% relative phone error reduction w.r.t. a phone recognizer that only uses MFCCs.

Index Terms— Acoustic-to-articulatory mapping, phone recognition, deep belief networks

1. INTRODUCTION

The well-known regular behavior of the vocal tract during speech production has motivated the use of production knowledge in ASR (see [9] for a review). When measured articulator positions are used, the main challenge is the construction by learning of a good acoustic-to-articulatory mapping (AAM) function that allows the recovery of the articulatory information from speech acoustics.

Typically the AAM is performed between acoustic coefficients (either MFCCs or mel-filtered filterbank coefficients, henceforth MFSCs) and sagittal plane positions of the vocal tract articulators. Here we experiment with AAM methods based on Deep Belief Networks (DBNs, [6]). The unsupervised learning of DBNs produces multi-layered hierarchical representations of their input domains that we then use to: (i) pre-train Multi-layer Perceptrons (MLP) that perform AAM; (ii) create alternative representations of the original "shallow" articulatory domain then used as new target in the AAM.

One of the main motivations behind the use of measured articulatory data recovered through AAM is the extraction of features complementary to standard acoustic features (e.g., MFCCs). If we could exactly reconstruct the articulatory data and add it to the observation vectors then we would produce a significant increase in word recognition accuracy [22, 21]. While an optimal learner would extract all the necessary features from acoustics only, the use of (reconstructed) articulatory information can be a good strategy to improve learning (while waiting for the optimal learner).

An alternative use of measured articulatory data consists in the creation of a speech-production based phonemic structure (e.g., [11]). Even in this case, articulatory data need to be recovered, although implicitly, from acoustics (in this case typically from MFCCs).

Neurophysiological evidence suggests that the recovering of articulatory information is a strategy used by the human brain, by far the best ASR system. Trans-cranial magnetic stimulation studies have shown that the activity of the motor cortex affects speech perception (see, e.g., [2]).

Recent work on AAM has been concerned with methods that appropriately address the ill-posedness of the AAM problem (e.g.,[17, 19]). Identical sounds can be produced by posing the articulators in a range of different positions [10] and in some cases the conditional probability of the position of an articulator given the acoustic evidence is even multi-modal [18]. However, [16] showed that, although the non-uniqueness of AAM is normal in human speech, most of the time the vocal tract has a unique configuration when producing a given phone. The fact that MLPs, which cannot properly handle non-uniqueness, are one of the best performing methods for AAM [12], suggests that non-linearity (another well-known characteristic of AAM) is a more important aspect than non-uniqueness.

Most of the parameters of an MLP can be pre-trained by first training in an unsupervised fashion a DBN and then "transforming" the (stochastic) DBN into a (deterministic) MLP (see section 2). MLP pre-training produces improved

*The authors acknowledge the support of the European Commission project POETICON++ (grant agreement 288382). Thanks to Marco Jacono and Alessandro Bruchi for support on CUDA and GPUs

performance in different tasks [3], including the AAM task [20]. The unsupervised pre-training followed by a supervised training (typically carried out through backpropagation), can be seen as a semi-supervised approach, where the knowledge of the statical properties of the input domain (i.e., $P(X)$) effectively guides the search for input-output relations (i.e., $P(Y|X)$). One of the goals of this paper is to experiment with two kinds of unsupervised DBN training for MLP pre-training: (i) one aiming at capturing statistical properties of the acoustic domain only (like in [20]); (ii) and one aiming at capturing statistical properties of the joint acoustic and articulatory domain (similarly to [15]).

A different (but also complementary) use of DBNs that we explore in this paper, is to train DBNs to transform the original "shallow" articulatory domain, which consists of articulator trajectories (plus their first and second derivatives), into a new "deep" domain that would become the new target of the AAM function.

The motivation behind such a domain transformation is that a better and more useful reconstruction of the articulatory data might be achieved if we apply transformations to the articulatory data that compactly encode the behavior of each articulator in relation to other articulators or parts of the vocal tract (e.g., upper teeth). In fact, in articulatory phonetics, the articulatory features (e.g., the consonant places of articulation) that distinguish phonemes almost always refer to the relative position of a critical articulator w.r.t. another critical articulator or a specific part of the vocal tract. The *vocal tract* features proposed by articulatory phonology [1] are an encouraging example of relative articulatory features. They have interesting properties, e.g., a smaller non-uniqueness than the articulator positions, and significantly improve word recognition in noisy speech conditions [13].

Obviously any feature set that is extracted from the original feature set can contain at most the same amount of information contained in the original feature set. However, by trying to reconstruct features that capture strong relations between parts of the vocal tract we may remove information that is irrelevant for phone discrimination and retain most of the information that is needed. Removing irrelevant information is important both to recover the articulatory information (because we are not forced to learn an AAM function to recover irrelevant and possibly noisy information) and to effectively use the reconstructed articulatory information to estimate the phone posterior probabilities (because, again, we ignore irrelevant and potentially noisy information). The first argument is one of the main motivations to directly recover transformed articulatory features rather than recovering shallow articulatory features first and subsequently transforming them.

Once we have recovered the articulatory features (either deep or shallow) we use them as observations in a hybrid DBN-HMM based phone recognizer. DBN-HMM systems (using MFCCs only) are state of the art in phone recognition [14]. In such systems the DBN is used to pre-train the MLP that computes the phone posteriors.

2. DEEP BELIEF NETWORKS

Although a DBN is a hybrid probabilistic graphical model it can be trained by approximating it to a stack of Restricted Boltzmann Machines (RBMs). An RBM is an undirected graphical model with a layer of visible nodes ($\mathbf{v}$) and a layer of hidden nodes ($\mathbf{h}$) with intra-layer connections and without any within-layer connection.

The joint probability of an RBM is:

$$P(\mathbf{v},\mathbf{h}) = \frac{1}{Z} exp(-E(\mathbf{v},\mathbf{h})) \quad (1)$$

where Z is the partition function and the energy function $E(\mathbf{v},\mathbf{h})$ for an RBM with both binary visible and hidden variables is:

$$E(\mathbf{v},\mathbf{h}) = -\sum_{i,j} v_i W_{ij} h_j - \sum_i b_i v_i - \sum_j c_j h_j \quad (2)$$

where W_{ij} are the connection weights and b_i and c_j are the biases on the visible and hidden nodes respectively.

Since there are no within-layer connections the probabilities $P(\mathbf{v}|\mathbf{h})$ and $P(\mathbf{h}|\mathbf{v})$ factorize and are given by:

$$P(v_i = 1|\mathbf{h}) = sigmoid(\sum_j W_{ij} h_j + b_i) \quad (3)$$

$$P(h_i = 1|\mathbf{v}) = sigmoid(\sum_i W_{ij} v_i + c_j) \quad (4)$$

The unsupervised learning of the parameters is performed by maximizing the $log(P(\mathbf{v})) = log(\sum_{\mathbf{h}} P(\mathbf{v},\mathbf{h}))$. The gradient update rule for a parameter θ_k is :

$$\Delta\theta_k \propto < \frac{\partial E(\mathbf{v},\mathbf{h})}{\partial \theta_k} >_{data} - < \frac{\partial E(\mathbf{v},\mathbf{h})}{\partial \theta_k} >_{model} \quad (5)$$

where $< ... >_{data}$ stands for expected value under the empirical distribution and $< ... >_{model}$ for expected value under the model distribution. The latter can be computed by running block Gibbs sampling where $P(\mathbf{h}|\mathbf{v})$ and $P(\mathbf{v}|\mathbf{h})$ are sampled. Rather that running Gibbs sampling until equilibrium we can still effectively train RBMs by using contrastive divergence [5] where the Gibbs sampler can run for just one step.

RBMs with Gaussian distributed visible (or hidden) variables can be also trained by applying simple changes to some of the equations above.

A DBN can be trained by using layer-wise training where the output (i.e., the values of the hidden nodes) of a trained RBM is used as input for the RBM above. Then unsupervised parameter fine-tuning can be applied where the DBN is considered as a whole deep architecture.

The hidden nodes of an RBM capture strong correlations between visible variables (i.e., the statistical structure of the input domain), so when we move bottom-up in a DBN we move to increasingly abstract representations.

When the DBN is used for pretraining an MLP all the DBN's stochastic nodes become deterministic and an output layer is added on the top. Finally supervised-fine tuning (e.g., backpropagation) is applied.

In this work, when the MLP is used for AAM the output unit activation function is a linear regressor with linear basis functions while when it is used for phone posteriors estimation the output unit activation function is a softmax function.

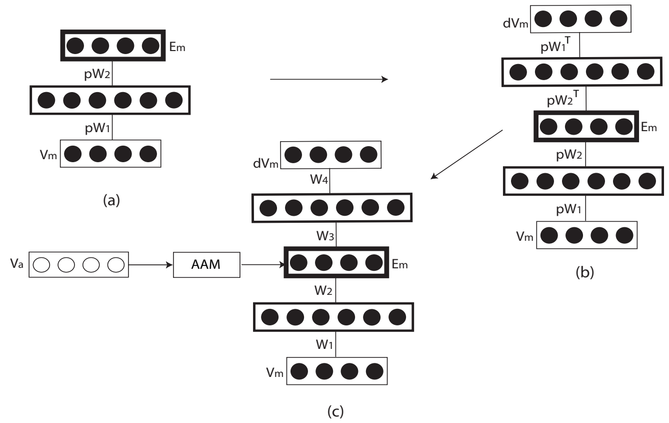

Fig. 1. DBN-based autoencoding of the (shallow) articulatory domain. (a) A 2-layer DBN is trained on the articulatory domain (Vm). (b) The DBN is "unfolded" by transposing the weight matrices pW1 and pW2. dVm is the articulatory feature set obtained after encoding (Vm → Em) followed by decoding (Em → dVm) (c) The unfolded DBN is fine-tuned by using backpropagation. The fine-tuning goal is minimizing the decoding error, i.e., the error between Vm and dVm. The final Em feature set is used as target in the acoustic-to-articulatory mapping. Va represents the acoustic space.

3. DBN-BASED AAM

3.1. DBN-based domain transformation

Once a DBN has been trained on the articulatory domain as described above, the nodes of its topmost layer (Em layer in figure 1a) seem good candidates to represent the new target domain of the AAM as they capture strong correlations between features of the original domain (represented by the Vm visible layer in figure 1). However the units of the topmost layer are stochastic, a not desirable behavior as the top layer can take different values given the same value of Vm.

One first solution would be that of simply replacing stochastic activities by deterministic activities. However if we want the new deep representation Em to encode all the relevant information contained in the input domain Vm (i.e., if we want to accurately recover the original representation from the encoded representation) we can use the DBN to pretrain a deep autoencoder as described by [7]. First the DBN is "unfolded" to create an encoding and a decoding network sharing the same weights (figure 1b). Subsequently the unfolded DBN is transformed into an MLP (the final autoencoder) and is fine-tuned using backpropagation to minimize the error reconstruction of the input space (figure 1c).

Finally the units of the encoding layer Em can be used as new target feature set in AAM.

In our experiments, the input layer of the DBN has 42 nodes (see section 4), the first hidden layer has 42x3 = 126 nodes and the second hidden layer has 42 nodes, so that Vm and Em have the same number of nodes. Em had Gaussian units as well as Vm. We did not experiment with different numbers of hidden layers or different numbers of nodes in the hidden layers.

3.2. MLP pretraining for AAM

We studied two configurations of RBM/DBN-based pretraining of the MLPs that performed AAM.

In the first configuration, similarly to [20], we trained a 3-layer DBN on the acoustic domain, replaced the stochastic activities by deterministic activities, added on top of the net a linear regressor layer and finally performed supervised fine-tuning where the articulatory features were the target. Before training the whole net we first trained the linear regressor to avoid the pretrained parameters being "dismantled" because of the initial large value of the error function.The MLP pretrained with this first configuration (henceforth pMLP) is a 3-layer MLP with 300 nodes on each hidden layer and 300 nodes in the visible layer (see section 4).

The goal of the second pretraining configuration, largely inspired by [15], was to "inform" the MLP performing AAM of the properties of the joint acoustic and articulatory domain. The pretraining (figure 2a) is carried out by training three different RBMs. The first RBM is trained on the acoustic domain only while the second RBM on the articulatory domain only. The third RBM is trained on the joint output of the first two RBMs. The stochastic activities of each RBM are then replaced by deterministic activities. Subsequently the three (deterministic) RBMs are combined to create the MLP (henceforth pifMLP) shown in figure 2b where the Hj hidden layer can be seen as an intermediate layer (between the acoustic and the articulatory layers) that encodes the joint acoustic and articulatory domain.

Note that, when using the third RBM (rightmost RBM in figure 2a) to create the pifMLP some edges need to be removed, specifically the edges connecting Ha and Hj in one case and the edges, with transposed weight matrix, connecting Hj to Hm. Unlike [15] we did not only remove those edges

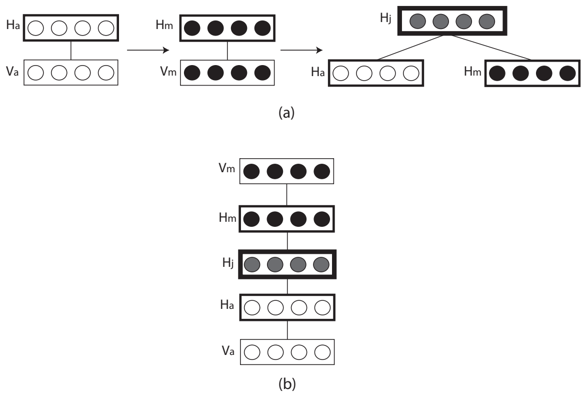

Fig. 2. RBM-based joint representation for pretraining of an MLP (pifMLP) used to perform AAM. (a) Left: RBM trained on the acoustic domain. Center: RBM trained on the articulatory domain. Right RBM trained on the output of the first 2 RBMs. (b) The three RBMs are then transformed into an MLP which is subsequently trained to perform AAM. Note that to carry out the transformation some weight matrices are transposed and some edges are removed.

but also recomputed the weights of the "pruned" RBM by actually replacing the RBM with a linear regressor with, in one case, the Ha input and the Hj output of the RBM as input-out pair, and, in the other case the Hj output and the Hm input of the RBM as input-output pair.

In our experiments the first RBM (figure 2a left) is a 300-300 RBM (i.e., an RBM with 300 nodes in the visible layer and 300 nodes in the hidden layer), the second (figure 2a center) is a 42-42 RBM , while the third is a 600-300 RBM so that the 3-layer pifMLP has exactly the same number of nodes per layer as pMLP.

4. EXPERIMENTAL SETUP

We used the msak0 dataset of the MOCHA-TIMIT corpus, consisting of simultaneous recordings of speech and Electromagnetic Articulographic (EMA) data (plus other articulatory data that we ignored).

The msak0 consists of 460 British English sentences uttered by a male speaker. The phonemes in the dataset are 44.

Speech was segmented into 25ms Hamming windows sampled every 10ms, from which we extracted (using HTK [8]): (i) 20 mel-scaled filterbank coefficients (i.e., mel-filtered spectra coefficients, MFSC) plus their deltas and delta-deltas and (ii) the first 12 MFCCs and 1 energy coefficient (plus their deltas and delta-deltas) from the same 20 mel-scaled filterbank channels and with a 0.97 pre-emphasis coefficient. The MFSCs were used for AAM while the MFCCs were used for phone posterior estimation.

Concerning the EMA data, we considered the x-y positions of upper and lower teeth, upper and lower lips and tongue tip, blade and dorsum. The 14 trajectories were first downsampled and smoothed using a moving average filter with a 15 ms smoothing window and their deltas and delta-deltas were computed (for an overall 42 features).

Training and testing datasets were created using the 5-fold cross-validation used in [21]. The training datasets were used to train the AAM function and the phone state posteriors estimator, and to compute phone state unigrams and bigrams at the frame-level.

The input to the MLPs that performed AAM consisted of 5 vectors of MFSCs (= 300 coefficients). The output consisted of a 42 articulatory feature vector (corresponding to the frame on which the 5 acoustic frames were centered) either for the shallow or the deep articulatory domain.

We used 3-state monophones, state boundaries were computed using the HInit, HRest and HERest functions of HTK.

Frame-wise state posteriors were computed by a DBN-pretrained 3-hidden-layer MLP, with 1500 nodes per hidden layer and 132 (= 44 phonemes x 3 states) output units. The number of units per layer, as well as the values of the other hyperparameters (e.g., the RBM training epochs), was selected through grid search in order to have the strongest possible baseline (i.e., phone recognizer using MFCCs only). The input to the MLPs consisted of 9 acoustic vectors of MFFCs plus, when articulatory features were used, the corresponding 9 vectors of articulatory features. All input features were normalized to have zero mean and unit variance.

The estimated state posteriors were fed into a Viterbi decoder. Dividing the state posteriors by the state priors resulted in a slightly poorer phone recognition performance so we kept the unscaled posteriors. The probabilities of state unigrams and bigrams were computed using Good-Turing discounting, and back-off for missing bigrams (excluding the bigrams that cannot occur, as, e.g., the bigrams of the central states of two different phonemes) .

To accelerate DBN unsupervised training and backpropagation we used the GPUmat Matlab toolbox [4] running on a Fermi S2050 Graphical Processing Unit.

	SA		DA	
	$RMSE_{avg}$	r_{avg}	$RMSE_{avg}$	r_{avg}
pMLP	0.74	0.65	0.45	0.62
pifMLP	0.74	0.66	-	

Table 1. Results (mean values) on articulatory feature reconstruction on a 5-fold cross validation. SA = shallow articulatory feature set (+ Δ, $\Delta\Delta$) . DA = deep articulatory features obtained through DBN-based auto-encoding of SA.

5. RESULTS

5.1. Reconstruction error: pMLP vs. pifMLP

The reconstruction of the articulatory data is usually evaluated using two measures: Root-mean-square error ($RMSE$) and Pearson product moment correlation coefficient (r).

The $RMSE$ for a given articulatory feature f is defined as:

$$RMSE_f = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(o_{f,i} - t_{f,i})^2} \qquad (6)$$

where N is the number of examples in the test dataset, $o_{f,i}$ is the estimated value of the feature f in the i-th example and t_i is its actual value. Here we use $RMSE_{avg}$, the RMSE averaged over all articulatory features ($RMSE_{avg} = \frac{1}{N_f}\sum_f^{N_f} RMSE_f$).

The Pearson product moment correlation coefficient for a given articulatory feature f is defined as:

$$r_f = \frac{\sum_{i=1}^{N}(o_{f,i} - \bar{o}_f)(t_{f,i} - \bar{t}_f)}{\sqrt{\sum_{i=1}^{N}(o_{f,i} - \bar{o}_f)^2 \sum_{i=1}^{N}(t_{f,i} - \bar{t}_f)^2}} \qquad (7)$$

where $\bar{o}_f$ and $\bar{t}_f$ are the mean value for the estimated feature and the actual feature respectively. Again we use the r averaged over all articulatory features ($r_{avg} = \frac{1}{N_f}\sum_f^{N_f} r_f$).

pMLP and pifMLP were compared on the reconstruction of the shallow articulatory feature set, i.e., the set consisting of articulator positions velocities and accelerations (see the SA column of table 1). pifMLP slightly outperforms, although not significantly, pMLP.

5.2. Shallow vs. Deep Articulatory Features

A first comparison between the shallow and the deep articulatory feature set was carried out by observing which feature set was easier to reconstruct (using pMLP, see the pMLP row of table 1). In terms of r the shallow features are easier to reconstruct while the $RMSE$s cannot be actually compared because of the two different ranges of the two variable types. So we normalized the two variable types to lie in the [0 1] range and the resulting normalized $RMSE$ was smaller when reconstructing shallow features. However, in a phone recognition task, the actual utility of an articulatory feature set only depends on the phone accuracy recognition it produces.

5.3. Phone Recognition

Table 2 shows frame-wise phone classification accuracy and PER for different observation sets used by the DBN-HMM based phone recognition system.

The most evident result is that measured articulatory information does improve phone recognition. When MFCCs are combined with recovered articulatory features the relative PER reduction w.r.t. to the baseline features set (MFCCs only) ranges from 12.7% to 16.6%. A perfect reconstruction of the articulatory information would lead to a 30.7% relative PER reduction.

Feature set	Mocha-Timit msak0	
	FwPCA (%)	PER
MFCCs	62.2	51.7
MFCCs + real SA	72.1	35.8
MFCCs + pMLP-recovered SA	67.3	43.9
MFCCs + pifMLP-recovered SA	67.3	43.8
MFCCs + pMLP-recovered DA	**67.8**	**43.1**

Table 2. Frame-wise phone classification accuracy (FwPCA) and Phone Error Rate (PER) on the Mocha-Timit msak0 dataset.

We believe this is a remarkable result for at least two reasons. First, previous work on the use of articulatory information for speech recognition has almost always shown a non-utility of recovered articulatory information in clean speech conditions [9] and only few exceptions exist (e.g., [11]).

Second, the baseline system we used, a DBN-HMM phone recognition system using MFCCs is very strong. DBN-HMM systems using MFCCs only are state of the art in phone recognition [14]. However it must be noted that our baseline PER is disappointingly not as low as that reported in previous work (e.g., [21], where a standard Gaussian Mixture Model-HMM system using triphones was used) where it is just below 40%. Actually after additional informal tests (i.e., not carried out using the 5-fold cross validation setting) we realized that by massively increasing the number of training epochs of the RBMs containing Gaussian units (which were trained using a much smaller learning rate than the all-binary units RBMs) it may still be possible to reduce the PER (approximately a 3-4% absolute reduction) of the systems listed in table 2, including the baseline, at the cost of a much slower training, but still the baseline would be higher than 40%. We speculate that the higher PER of our baseline might be due to the fact that DBNs perform at their best when trained on large corpora (like the TIMIT corpus) while on small corpora their potential is not fully exploited. The higher PER is also probably due to differences in the language model. E.g., while in [21] the phone language model was computed on the whole MOCHA-TIMIT corpus and its weight adjusted to obtain a lower PER, our state language model was computed on the training data sets only and we did not play with the language model weight.

In our view, one of the main causes of the largely significant utility of the articulatory information in a DBN-HMM phone recognition system is that the DBN-based state classifier exploits the articulatory information better than other techniques used to estimate the emission probabilities (e.g., Gaussian mixtures in a standard HMM phone recognizer).

Another interesting result is that deep articulatory features slightly outperform shallow articulatory features. This result suggests that the automatic and DBN-based transformation of the articulatory space is a promising direction for future studies.

6. CONCLUSION

In this paper we (i) experimented with acoustic-to-articulatory mapping (AAM) methods based on Deep Belief Networks (DBNs) and (ii) tested the utility of the articulatory data, recovered through AAM, when used in a DBN-HMM phone recognition system. A DBN-HMM phone recognizer is an HMM system that uses a DBN pretrained Multi-layer Perceptron (MLP) to compute the state emission probabilities. We have shown a significant reduction of phone error rate in a speaker dependent task when MFCCs are combined with recovered articulatory features. Finally results also suggest that DBN-based transformations of the articulatory domain (considered as target in the AAM process) can further increase phone recognition accuracy.

7. REFERENCES

[1] Browman, C.P. and Goldstein, L., “Articulatory phonology: an overview”, Phonetica 49 (34): 155-180, 1992.

[2] D‘Ausilio, A., Pulvermller, F., Salmas, P., Bufalari, I., Begliomini, C., and Fadiga, L., “The motor somatotopy of speech perception”, Current Biology, 19, 381-385, 2009.

[3] Erhan, D., Bengio, Y., Courville, A., Manzagol, P.A., Vincent, P., and Bengio, S., “Why Does Unsupervised Pre-training Help Deep Learning?”, in J. of Machine Learning Research, 11 (625–660), 2011

[4] Available at http://gp-you.org/ .

[5] Hinton, G.E., “Training products of experts by minimizing contrastive divergence”, Neural Computation,vol. 14, pp. 1771–1800, 2002.

[6] Hinton, G.E., Osindero, S. and Teh, Y., “A fast learning algorithm for deep belief nets”, Neural Computation, vol. 18, pp. 1527–1554, 2006.

[7] Hinton, G. E., and Salakhutdinov, R. R., “Reducing the dimensionality of data with neural networks”, Science, Vol. 313. no. 5786, pp. 504 - 507, 28 July 2006.

[8] Available at http://htk.eng.cam.ac.uk/ .

[9] King, S., Frankel, J., Livescu, K., McDermott, E., Richmond, K., and Wester, M.,“Speech production knowledge in automatic speech recognition”, J. of the Acoust. Soc. Am., vol. 121(2), pp. 723-742, 2007.

[10] Lindblom, B., Lubker, J., and Gay, T.,“Formant frequencies of some fixed-mandible vowels and a model of speech motor programming by predictive simulation”, J. of Phonetics, vol. 7, 146–161, 1979.

[11] Markov, K., Dang, J. and Nakamura, S., “Integration of articulatory and spectrum features based on the hybrid HMM/BN modeling framework” Speech Communication, 48, 161–175, 2006.

[12] Mitra, V., Nam, H., Espy-Wilson, C., Saltzman, E., Goldstein, L., “Retrieving tract variables from acoustics: a comparison of different machine learning strategies”, IEEE J. of Selected Topics in Signal Processing, vol. 4 (6), 1027–1045, 2010.

[13] Mitra, V., Nam, H., Espy-Wilson, C., Saltzman, E., Goldstein, L., “Recognizing articulatory gestures from speech for robust speech recognition”, J. Acoust. Soc. Am. 131 (3), 2270–2287, 2012

[14] Mohamed, A. R., Dahl, G. E. and Hinton, G. E.“Deep belief networks for phone recognition”, NIPS 22, workshop on deep learning for speech recognition, 2010.

[15] Ngiam, J., Nam, M., Lee, J., Khosla, H., Kim, A., and Ng, A.Y., “Multimodal deep learning”, in ICML, 2011.

[16] Qin, C., and Carreira-Perpinan, M.A. “An Empirical Investigation of the Nonuniqueness in the Acoustic-to-Articulatory Mapping”, Proc. Interspeech, 2007

[17] Richmond, K., King, S., and Taylor, P., “Modelling the uncertainty in recovering articulation from acoustics”, Computer Speech and Language, vol. 17(2), pp. 153-172, 2003.

[18] Roweiss, S., “Data driven production models for speech processing”, PhD thesis, California Institute of Technology, Pasadena, California, 1999.

[19] Toda, T., Black, A., and Tokuda, K., “Statistical Mapping between Articulatory Movements and Acoustic Spectrum Using a Gaussian Mixture Model”, Speech Communication, vol. 50(3), 215–222, 2007.

[20] Uria, B., Renals, S., and Richmond, K.,“A deep neural network for acoustic-articulatory speech inversion”, In Proc. NIPS Workshop on Deep Learning and Unsupervised Feature Learning, 2011.

[21] Wrench, A.A. and Richmond, K.,“Continuous speech recognition using articulatory data”, in Proceedings of the International Conference on Spoken Language Processing, pp. 145-148, 2000.

[22] Zlokarnik, I., “Adding articulatory features to acoustic features for automatic speech recognition”. J. of the Acoust. Soc. Am., vol. 97(2), pp. 3246, 1995.