Available online at www.sciencedirect.com
ScienceDirect
Computer Speech and Language xxx (2015) xxx–xxx
COMPUTER SPEECH AND LANGUAGE
www.elsevier.com/locate/csl

Integrating articulatory data in deep neural network-based acoustic modeling☆

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

Istituto Italiano di Tecnologia, via Morego 30, 16163 Genova, Italy

Received 19 June 2014; received in revised form 6 May 2015; accepted 26 May 2015

Abstract

Hybrid deep neural network–hidden Markov model (DNN-HMM) systems have become the state-of-the-art in automatic speech recognition. In this paper we experiment with DNN-HMM phone recognition systems that use measured articulatory information. Deep neural networks are both used to compute phone posterior probabilities and to perform acoustic-to-articulatory mapping (AAM). The AAM processes we propose are based on deep representations of the acoustic and the articulatory domains. Such representations allow to: (i) create different pre-training configurations of the DNNs that perform AAM; (ii) perform AAM on a transformed (through DNN autoencoders) articulatory feature (AF) space that captures strong statistical dependencies between articulators. Traditionally, neural networks that approximate the AAM are used to generate AFs that are appended to the observation vector of the speech recognition system. Here we also study a novel approach (AAM-based pretraining) where a DNN performing the AAM is instead used to pretrain the DNN that computes the phone posteriors. Evaluations on both the MOCHA-TIMIT msak0 and the mngu0 datasets show that: (i) the recovered AFs reduce phone error rate (PER) in both clean and noisy speech conditions, with a maximum 10.1% relative phone error reduction in clean speech conditions obtained when autoencoder-transformed AFs are used; (ii) AAM-based pretraining could be a viable strategy to exploit the available small articulatory datasets to improve acoustic models trained on large acoustic-only datasets.

Keywords: DNN-HMM; Acoustic-to-articulatory mapping; Deep neural networks; Acoustic modeling; Electromagnetic articulography; Autoencoders

1. Introduction

The steady increase of training data and computational resources combined with the use of new machine learning strategies for acoustic modeling has been continuously improving ASR performance in the last few years. Deep neural networks (DNNs) (Hinton et al., 2006), either combined with HMMs or used in a recurrent architecture, are the best strategy for acoustic modeling (Mohamed et al., 2012; Dahl et al., 2012; Graves et al., 2013).

☆ This paper has been recommended for acceptance by Shrikanth Narayanan.

* Corresponding author. Tel.: +39 010 71781975.
E-mail address: leonardo.badino@iit.it (L. Badino).

http://dx.doi.org/10.1016/j.csl.2015.05.005
0885-2308/

However, despite the impressive results shown by DNN-based ASR, there are several real usage scenarios where ASR technology still needs large improvements. In general, ASR accuracy significantly decreases in mismatched training-testing conditions, as it has been shown for traditional Gaussian mixture model (GMM)-HMMs systems in, e.g., speaking style mismatched conditions (Yu et al., 1999), and for DNN-HMM systems in, e.g., environment and microphone mismatched conditions (Seltzer et al., 2013).

Other than simply increasing the number of training conditions we can explicitly address the speech modeling limitations responsible for the lack of generalization underlying the mismatched conditions problem. For example, context-dependent (CD)-DNN-HMMs, as well as GMM-HMMs, handle context effects (like, e.g., coarticulation effects) using hundreds/thousands of tied context dependent sub-phonetic states, i.e., senones (Dahl et al., 2012). The selection, either automatic or manual, of the number of senones (and, consequently, of learning parameters) may be affected by the number of conditions in the training dataset and, at the same time, by the invariance of the input feature set to those conditions (see, e.g., (Schaaf and Metze, 2010) where the portion of gender-dependent senones depends on the feature set used).

The senones themselves result from the need to reduce learning parameters and are created by exploiting some speech production knowledge in the form of speech production-based questions in the state clustering tree. However ASR may benefit from a more explicit use of speech production knowledge where speech production can be used as, e.g., additional observations appended to the vector of acoustic observations, or as hidden structure connecting the phonological level (i.e., the HMM hidden phonetic states) to the observed speech acoustics.

Such approaches are motivated by the fact that complex phenomena observed in speech, for which a simple purely acoustic description has still to be found, can be easily and compactly described in speech production-based representations (notably Browman and Goldstein, 1992; Jakobson et al., 1952; Chomsky and Halle, 1968). For example, in Articulatory Phonology (Browman and Goldstein, 1992) or in the distinctive features framework (Jakobson et al., 1952; Chomsky and Halle, 1968) coarticulation effects can be compactly modeled as temporal overlaps of few vocal tract gestures. The vocal tract gestures are regarded as invariant, i.e., context- and speaker-independent, production targets that contribute to the realization of a phonetic segment. Obviously the invariance of a vocal tract gesture partly depends on the degree of abstraction of the representation but speech production representations offer compact descriptions of complex phenomena and of phonetic targets that purely acoustic representations are not able to provide yet (see, e.g., Maddieson, 1997).

Additional motivations to the use of speech production in ASR come from theories of speech perception such as the well known Motor Theory of speech perception (Liberman et al., 1967; Galantucci et al., 2006) which assumes that the perception of speech is the perception of motor gestures and involves access to the motor system. Such claims are partly supported by neurophysiological studies that show the contribution of the activity of the motor cortex to speech perception (DAusilio et al., 2009; Bartoli et al., 2013).

In the last two decades several strategies have been proposed for an explicit use of speech production knowledge in ASR (see King et al., 2007, for an extensive review). Here we review studies where measured articulatory data are used for ASR. Such studies require simultaneous recordings of audio and articulatory data. Articulatory movements are recorded using techniques such as electro-magnetic articulography (EMA) (Wrench, 2000), X-rays (Westbury, 1994), ultrasounds (e.g., Grimaldi et al., 2008), and MRI (Narayanan et al., 2004).

The approaches that use measured articulatory data can be roughly grouped into two categories. In the first category (e.g., Stephenson et al., 2000; Markov et al., 2006; Mitra et al., 2012) articulatory information is represented as discrete latent variables which are observed during training but hidden during testing. The idea behind this approach is to explicitly and compactly model speech production processes that are among the main causes of acoustic variability (e.g., variability due to coarticulation effects). In the second category (e.g., Zlokarnik, 1995; Wrench and Richmond, 2000), which the present work belongs to, articulatory features (AFs) are recovered from speech acoustics and then appended to the vector of observed acoustic features. In this case the working hypothesis is that the recovered articulatory domain (combined with the acoustic domain) represents a transformation of the acoustic domain into a new speech-production constrained domain which is more invariant over different conditions and where phonetic-articulatory targets can be more easily discriminated.

We first review some of the studies belonging to the first category. In Stephenson et al. (2000) the articulatory information is represented by a single discrete articulatory variable within a dynamic Bayesian network (DBN). Its values are computed by clustering data points in a space defined by eight articulator sagittal positions (upper lip, lower lip, four tongue positions, lower front teeth, lower back teeth). The acoustic observation probability distribution is

Please cite this article in press as: Badino, L., et al., Integrating articulatory data in deep neural network-based acoustic modeling. Comput. Speech Lang. (2015), http://dx.doi.org/10.1016/j.csl.2015.05.005

both conditioned on the phone state and on the articulatory variable which in turn depends on the phone state and the previous articulatory value.

In Markov et al. (2006), not only the articulator position but also velocity and acceleration are taken into account and a latent discrete variable is used for each of them within a Bayesian Network that substitutes the traditional GMM to model the state-dependent observation probability distributions in HMM-based ASR. Contrary to Stephenson et al. (2000), the articulatory variables are not conditioned on their previous values. Both Stephenson et al. (2000) and Markov et al. (2006) show an increased phone recognition accuracy when latent articulatory variables are used.

In the Gesture-based DBN (G-DBN) proposed by Mitra et al. (2012), articulatory features are derived from the Articulatory Phonology theory (Browman and Goldstein, 1992). The most interesting contribution of the paper is the use of articulatory features that attempt to explicitly describe the phonetic-articulatory targets. The G-DBN integrates articulatory information at two levels, which can be seen as a motor planning and a motor execution level. The motor planning is represented as six latent binary variables, where each variable encodes the activation state of an Articulatory Gesture (e.g., glottis constriction, tongue body constriction, lip aperture). The motor execution level is represented as observed tract variables (TVs) appended to the acoustic observation vector. The TVs define the kinematics of the vocal tract determined by the activations of the articulatory gestures. In realistic ASR settings the TVs, although represented as observed features, are not available during testing and need to be recovered from acoustics through an acoustic-to-articulatory mapping (AAM). Results on the Aurora-2 corpus (Pearce and Hirsch, 2000) showed that the G-DBN is more robust to noise than the acoustics-only DBN. A current limitation of the approach is that the ASR system can be trained on synthetic (acoustic and articulatory) speech, and consequently, the speech variability of the training data is quite limited.

The first studies belonging to the second category, where measured articulatory data are only used as observations, are Zlokarnik (1995) and Wrench and Richmond (2000) where recovered AFs are appended in a GMM-HMM system. The two studies report conflicting results, AFs are of no utility in Wrench and Richmond (2000) whereas produce a large WER reduction in Zlokarnik (1995). The improvement in Zlokarnik (1995) may be due to the very large acoustic context (51 frames) used to reconstruct the AFs. The WER reduction may be simply due to the implicit observation of a larger acoustic context.

A critical factor for the success of measured AFs used as observation is the accuracy of the Acoustic-to-Articulatory mapping (AAM, also referred to as speech inversion problem). Some studies on AAM have proposed methods that appropriately address the non-uniqueness of the AAM problem (Richmond et al., 2003; Richmond, 2006; Toda et al., 2007). The non-uniqueness implies that identical sounds can be produced by posing the articulators in a range of different positions (Lindblom et al., 1979). As a consequence the conditional probability density function of the position of an articulator given a speech sound can exhibit more than one mode (Roweiss, 1999). In other words, the AAM can be a one-to-many mapping. However, Qin and Carreira-Perpiñán (2007) 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. Non-linearity seems to be a more relevant aspect to address. That is partly supported by the fact that feed-forward neural networks, which cannot properly approximate one-to-many mappings but can approximate non-linear functions, are one of the best performing methods for AAM (Mitra et al., 2010).

The successful learning of the AAM can depend on the type of representation of the articulatory data. For example, the representation may affect the degree of non-uniqueness and non-linearity of the AAM. Representations where a feature encodes the coordinated movement of two or more vocal tract parts can facilitate the learning of the AAM as opposed to representations where each feature encodes the movement of one single vocal tract part (e.g., tract variables vs. articulator flesh points (Mitra et al., 2011)).

The idea of transforming the acoustic domain by using measured articulatory data can also be accomplished without an explicit AAM. Multi-view learning based on canonical correlation analysis (CCA) has been proposed (Bharadwaj et al., 2012; Arora and Livescu, 2013, 2014) to find pairs of maximally correlated projected data in the acoustic and articulatory view. Then the acoustic projection is retained and appended to the acoustic observation vector. CCA-extracted features reduce PER in both speaker-dependent, and cross-speaker and cross-domain settings in a GMM-HMM phone recognition system (Arora and Livescu, 2013, 2014).

The present work follows up our previous work (Badino et al., 2012; Canevari et al., 2012) where we showed that appending AFs to the acoustic observation vector reduces the error rate of a speaker-dependent hybrid DNN-HMM phone recognition system, in the MOCHA-TIMIT corpus (Badino et al., 2012) and over different datasets (Canevari et al., 2012).

Please cite this article in press as: Badino, L., et al., Integrating articulatory data in deep neural network-based acoustic modeling. Comput. Speech Lang. (2015), http://dx.doi.org/10.1016/j.csl.2015.05.005

Compared to previous work, Badino et al. (2012) is the first attempt to integrate measured articulatory data in DNN-HMM acoustic models. In Badino et al. (2012) DNNs serve different purposes. A first DNN is used to learn the AAM and two different DNN pretraining strategies are compared. A second DNN is trained to compute phone state posteriors (henceforth shortened to phone posteriors) given the combined acoustic and recovered (through AAM) articulatory observations. Additionally a third DNN, in the form of a deep autoencoder (AE), is used to extract, from the space of single independent movements of each articulator, a new articulatory feature space that captures the most relevant "gestures" of the vocal tract and ignores the irrelevant ones. Such domain transformation aims at facilitating the learning of the AAM and at improving phone posterior estimation. That is in the same spirit of Mitra et al. (2012) where features derived from Articulatory Phonology represent the articulatory domain, although our approach is entirely data-driven and theory-free.

Here we advance (Badino et al., 2012; Canevari et al., 2012) in many important aspects, both algorithmic and experimental. Our goals are: improve (DNN-based) methods that perform the AAM and asses the impact of pretraining in the AAM task; find better (autoencoder based) data-driven transformations of the articulatory domain and understand what they represent; assess the utility of the articulatory data in mismatched conditions where the phone recognizer is trained on clean speech and tested in different noisy environments; find DNN-based methods to exploit the very limited availability of articulatory data in cross-speaker and cross-domain settings, i.e., settings where the articulatory data of one or few speakers are used to improve DNN-based acoustic models, trained on purely acoustic corpora.

The remainder of this paper is organized as follows. Section 2 briefly introduces deep neural networks and autoencoders. In the first part of Section 3 we review the pretraining strategies for DNNs performing AAM described in Badino et al. (2012) and propose some new variants. In the second part of Section 3 we review deep autoencoder-based transformations of the articulatory domain and propose supervised autoencoding, where autoencoders exploit information about the phonetic class associated to their input vector. Section 4 describes the two strategies applied to exploit articulatory data for phone posterior estimation: the standard approach where reconstructed AFs are appended to the observation vector and a novel approach, which we have named AAM-based pretraining, where articulatory features are not explicitly observed to compute phone posteriors. Section 5 describes the experimental setup. Section 6 shows results in AAM accuracy, autoencoder-based articulatory feature extraction, and phone recognition accuracy. Finally we analyze results and discuss future directions in Section 7.

2. Deep neural networks and autoencoders

2.1. Deep neural networks

In their standard formulation DNNs are feed-forward neural networks whose parameters are first "pre-trained" using unsupervised training of deep belief networks (Hinton et al., 2006). DNNs can be seen as an improved version of feed-forward neural networks that exploits the knowledge of the statistical properties of the input domain (i.e., $P(X)$) to effectively guide the search for input-output relations (i.e., $P(Y|X)$).

The DNN training is carried out as follows. First a deep belief network (DBN, henceforth DBN refers to deep belief network and not to dynamic Bayesian network as above) is trained in an unsupervised fashion. Subsequently the DBN is transformed into a DNN by converting the stochastic activation function of each node into a deterministic function. If the DNN is used to perform regression or classification an output layer is added on top of the deterministic net. Finally, supervised fine-tuning of the parameters is applied, typically using backpropagation.

The DBN can be trained by approximating it to a stack of restricted Boltzmann machines (RBMs). An RBM (Smolensky, 1986) 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:

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

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:

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

Please cite this article in press as: Badino, L., et al., Integrating articulatory data in deep neural network-based acoustic modeling. Comput. Speech Lang. (2015), http://dx.doi.org/10.1016/j.csl.2015.05.005

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

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 update rule for a parameter θ_k is:

$$\Delta\theta_k \propto \langle \frac{\partial E(\mathbf{v}, \mathbf{h})}{\partial \theta_k} \rangle_{data} - \langle \frac{\partial E(\mathbf{v}, \mathbf{h})}{\partial \theta_k} \rangle_{model} \tag{3}$$

where $\langle \dots \rangle_{data}$ stands for expected value under the empirical distribution and $\langle \dots \rangle_{model}$ for expected value under the model distribution (Hinton and Sejnowski, 1986). They are computed using contrastive divergence (Hinton, 2002). RBMs with Gaussian distributed visible (or hidden) variables can be also trained by applying simple changes to some of the equations above (Welling et al., 2005).

2.2. Autoencoders

An AE is a particular neural network that consists of an encoding and a decoding part. The encoder maps an input vector $\mathbf{x}$ into a hidden/encoding representation $\mathbf{h}$:

$$\mathbf{h} = f_\theta(\mathbf{x}) = s(\mathbf{W}\mathbf{x} + \mathbf{b}) \tag{4}$$

where $\mathbf{W}$ is a weight matrix, $\mathbf{b}$ a bias vector and s is typically the sigmoid function.

The decoder maps back the hidden vector $\mathbf{h}$ to a "reconstructed" input $\mathbf{y}$:

$$\mathbf{y} = g_{\theta'}(\mathbf{h}) = l(\mathrm{W}'\mathrm{h} + \mathbf{b}') \tag{5}$$

The AE is trained to minimize the distance between its input and its output (Fig. 3a), i.e., the reconstruction error. If the input data are assumed to be Gaussian distributed, as in the present work, l is typically an identity function and the AE is trained, usually through backpropagation, to minimize the squared error function $\|\mathbf{x} - \mathbf{y}\|^2$.

An AE can be either used to reduce the dimensionality of the input domain or to generate overcomplete representations where the number of encoding nodes (i.e., extracted features) is larger than the number of input features.

Single-layer AEs can be stacked to create a deep AE. An effective strategy to train deep AEs was proposed by Hinton and Salakhutdinov (2006) where a DBN corresponding to the encoding part of the deep AE is first trained (Fig. 3a), then it is "unrolled" to create the decoding part of the deep AE (Fig. 3b) and the resulting unrolled net is fine-tuned to minimize the reconstruction error.

In the present work we use deep AEs where encoding nodes are as many as the input nodes and their values lie in the [0 1] range.

2.2.1. Denoising autoencoders

A simple variant of the standard AE is the denoising AE (DAE) (Vincent et al., 2010), where the training input to the AE is transformed in $\bar{\mathrm{x}}$ by corrupting the input, while the training objective is kept unaltered ($\|\mathbf{x} - \mathbf{y}\|^2$, Fig. 3b). For Gaussian distributed input the input is corrupted by adding Gaussian noise (e.g., with 0 mean and 0.5 standard deviation as in the present work). The expectation is that the corruption of the input will not only make the AE more robust to noise but will also force the AE to capture the most stable and relevant dependencies between input features and ignore the irrelevant ones.

3. DNN-based acoustic-to-articulatory mapping

In this section we review methods proposed in our previous work (Badino et al., 2012) and propose variants and novel methods to: (i) learn the AAM with DNNs (Section 3.1); (ii) extract, through an AE, a new set of articulatory features (Section 3.2).

3.1. Learning the acoustic-to-articulatory mapping with DNNs

We experimented with three different DNNs to perform AAM. We named the three nets as aDNN1, jDNN1 and jDNN2 (depicted in Fig. 1c, d and e respectively). The lower case letter preceding DNN indicates the domain on which the DNN is pretrained, where letter *a* stands for acoustic domain and letter *j* stands for joint acoustic and articulatory

Please cite this article in press as: Badino, L., et al., Integrating articulatory data in deep neural network-based acoustic modeling. Comput. Speech Lang. (2015), http://dx.doi.org/10.1016/j.csl.2015.05.005

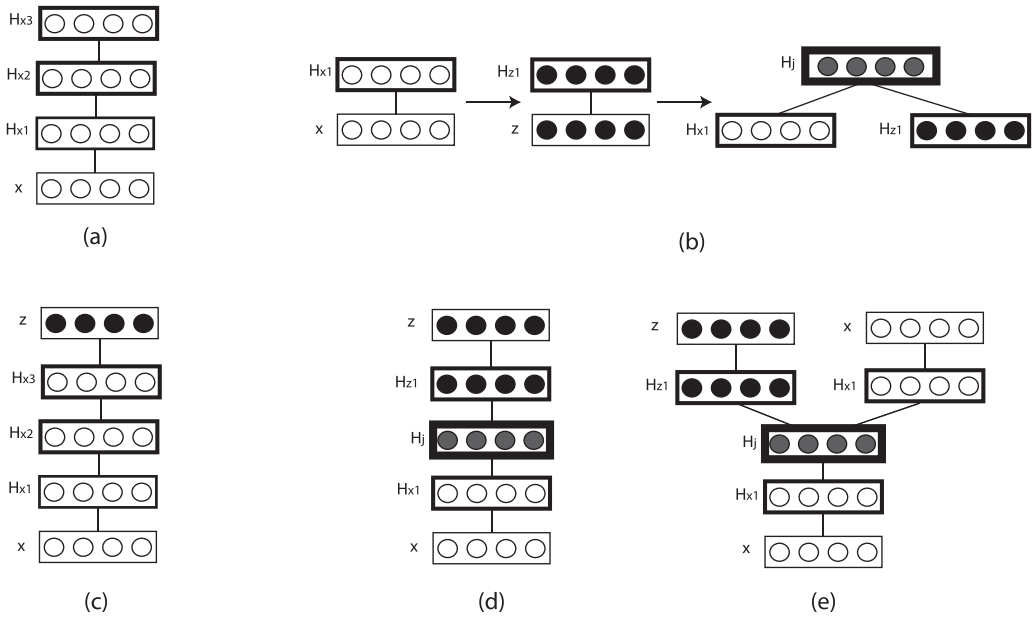

Fig. 1. DNNs for AAM. Empty nodes are acoustic nodes (i.e., only refer to the acoustic space), black nodes are articulatory nodes and gray nodes are acoustic/articulatory nodes. (a) DBN of aDNN1 (aDNN1 pretraining). (b) RBMs of jDNN1 and jDNN2. (c) aDNN1. (d) jDNN1. (e) jDNN2

domain. The number following DNN indicates the type of architecture (thus, e.g., aDNN1 and jDNN1 share the same architecture but are differently pretrained).

As in Uria et al. (2011) and Badino et al. (2012), aDNN1 is pretrained with a DBN trained on the acoustic domain (Fig. 1a) and then trained to perform AAM after adding a layer of linear regressors on top of the DBN (Fig. 1c).

jDNN1 (first proposed in Badino et al., 2012) is trained as follows. The pretraining is carried out by training three different RBMs (Fig. 1b). The first RBM (acoustic RBM) is trained on the acoustic domain while the second RBM (articulatory RBM) is trained on the articulatory domain. The third RBM (joint 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 jDNN1 shown in Fig. 1d, which in turn is trained to learn the AAM (note that no linear output layer is added). When combining the RBMs, some edges of the joint RBM are removed, resulting into a "pruned" RBM. Specifically the input to the joint RBM is $\mathbf{v} = [\mathbf{x}\,\mathbf{z}]$ where $\mathbf{x}$ is the input vector of acoustic features and $\mathbf{z}$ is the input vector of AFs. The input to the pruned joint RBM only consists of $\mathbf{x}$ and its weight matrix $\mathbf{W}_R$ is a reduced version of the $\mathbf{W}$ weight matrix of the full joint RBM as all edges linking $\mathbf{z}$ to $\mathbf{h}$ are removed. Below we propose a novel strategy to "preserve" the RBM training in the pruned joint RBMs.

jDNN2 is pretrained exactly as jDNN1 but the RBMs are differently combined. As for jDNN1 the joint RBM is used twice, but its "transpose" is not pruned, and the acoustic RBM is used twice (its transpose is on top of the transpose joint RBM). jDNN2 can be regarded as a multi-task network that tries to both perform AAM and reconstruct the acoustic domain (like an autoencoder). The idea behind multi-task learning is that of improving generalization by leveraging the domain-specific knowledge learned from related tasks.

A common goal of jDNN1 and jDNN2 pretraining is to leverage articulatory information in the pretraining phase by driving the representation of the acoustic domain towards a speech production-constrained representation. Both DNN types are largely inspired by Ngiam et al. (2011). However in Ngiam et al. (2011) no action is taken to recompute the weights of the joint RBM in order to preserve its training when some of its edges are removed. Here we propose an alternative recomputation strategy to that proposed in Badino et al. (2012).

If, given the input acoustic vector $\mathbf{x}$, we want the weight matrix of the pruned joint RBM $\mathbf{W}_R$ to generate hidden activations $\tilde{\mathbf{h}}$ as much as possible similar to those generated by the full joint RBM ($\mathbf{h}$) when fed with input $\mathbf{v} = [\mathbf{x}\,\mathbf{z}]$, then a possible strategy consists in updating the pruned RBM parameters to maximize $\log(P(\mathbf{h}|\mathbf{x})) = \log(\exp(-E(\mathbf{h},$

$\mathbf{x}))/\sum_{\mathbf{h}} \exp(-E(\mathbf{h}, \mathbf{x})))$ (where $\mathbf{h}$ is the target value, obtained from the full joint RBM). That requires the computation of $\partial \log P(\mathbf{h}|\mathbf{x})/\partial\theta$

$$\frac{\partial \log P(\mathbf{h}|\mathbf{x})}{\partial\theta} = \frac{\partial(-E(\mathbf{h}, \mathbf{x}) - \log \sum_{\mathbf{h}} \exp(-E(\mathbf{h}, \mathbf{x})))}{\partial\theta} \tag{6}$$

$$\frac{\partial \log P(\mathbf{h}|\mathbf{x})}{\partial\theta} = -\frac{\partial E(\mathbf{h}, \mathbf{x})}{\partial\theta} + \sum_{\mathbf{h}} \frac{\exp(-E(\mathbf{h}, \mathbf{x}))}{\sum_{\widehat{\mathrm{h}}} \exp(-E(\widehat{\mathrm{h}}, \mathbf{x}))} \frac{\partial E(\mathbf{h}, \mathbf{x})}{\partial\theta} \tag{7}$$

$$\frac{\partial \log P(\mathbf{h}|\mathbf{x})}{\partial\theta} = -\frac{\partial E(\mathbf{h}, \mathbf{x})}{\partial\theta} + \sum_{\mathbf{h}} P(\mathbf{h}|\mathbf{x}) \frac{\partial E(\mathbf{h}, \mathbf{x})}{\partial\theta} \tag{8}$$

Considering that in a RBM $P(\mathbf{h}|\mathbf{x})$ factorizes, the partial derivative $\partial \log P(\mathbf{h}|\mathbf{x})/\partial w_{ij}$, where w_{ij} is the weight parameter of the edge that connects node x_i to node h_j is:

$$\frac{\partial \log P(\mathbf{h}|\mathbf{x})}{\partial w_{ij}} = x_i h_j + \sum_{\tilde{h}_1} P(\tilde{h}_1|\mathbf{x}) \cdots \sum_{\tilde{h}_j} P(\tilde{h}_j|\mathbf{x}) \cdots \sum_{\tilde{h}_H} P(\tilde{h}_H|\mathbf{x})(-x_i \tilde{h}_j) \tag{9}$$

$$\frac{\partial \log P(\mathbf{h}|\mathbf{x})}{\partial w_{ij}} = x_i h_j - \sum_{\tilde{h}_j} P(\tilde{h}_j|\mathbf{x}) x_i \tilde{h}_j \tag{10}$$

$$\frac{\partial \log P(\mathbf{h}|\mathbf{x})}{\partial w_{ij}} = x_i h_j - \langle x_i \tilde{h}_j \rangle_{P(\mathrm{h}|\mathbf{x})} \tag{11}$$

and we can use a simplified version of contrastive divergence to approximate the partial derivative where $x_i h_j$ is already given by the full RBM and for the second term we sample $P(\tilde{\mathrm{h}}|\mathbf{x})$ from the pruned RBM just once to approximate $\langle x_i \tilde{h}_j \rangle_{P(\mathrm{h}|\mathbf{x})}$.

Contrary to jDNN1 and jDNN2, the topmost layer of aDNN1 is not pretrained as it is added after pretraining. A rule of the thumb to properly train aDNN1 is that of first training the topmost layer for few training epochs and than fine-tuning the entire net.

3.2. DNN-based transformation of the articulatory domain

Rather than simply representing the articulatory domain as the set of independent movements of each articulator flesh point recorded, e.g., by an EMA, we may aim at representing the articulatory domain as a set of features that encode the coordinated movements of different flesh points. In other words features that represents gestures of the vocal tract, where here gesture is meant as a statistically relevant movement or configuration of the vocal tract. Such transformation may facilitate the DNN-based AAM. In fact, in a DNN performing the reconstruction of all AFs, the topmost layer is a bank of independent regressors, each predicting one output feature value independently of all the other output features (given the values of the topmost hidden layer, which provides a representation of the acoustic domain shared by all regressors). When using flesh point movements, DNN maps speech acoustics on each single articulator flesh point movement/position whose effect on speech acoustics is marginal and strongly depends on the other flesh point movements/positions. By using features that encode the combined movements/positions of different flesh points we attempt to reconstruct features that may have a more direct relation to speech acoustics.

The tract variables of Articulatory Phonology are an example of such kind of features, where the vocal tract behaviour is described as a set of constriction degrees and locations (e.g., lip aperture, tongue tip constriction degree and location). Here, rather than extracting theory-derived AFs, we follow a data-driven approach where new features are automatically extracted by an AE.

As in Badino et al. (2012) we first train an AE to extract new AFs given the flesh point features and then learn the AAM on the new AFs as shown in Fig. 2. Henceforth we will refer to the flesh point positions velocities and acceleration as raw articulatory features (rAFs) and to the AE-extracted features as autoencoder articulatory features (aAFs).

In this paper we do not only experiment with standard AEs but also with denoising AEs (DAEs) and AEs that exploit supervised information, i.e., information about the phonetic class associated to the vector of raw AFs.

Please cite this article in press as: Badino, L., et al., Integrating articulatory data in deep neural network-based acoustic modeling. Comput. Speech Lang. (2015), http://dx.doi.org/10.1016/j.csl.2015.05.005

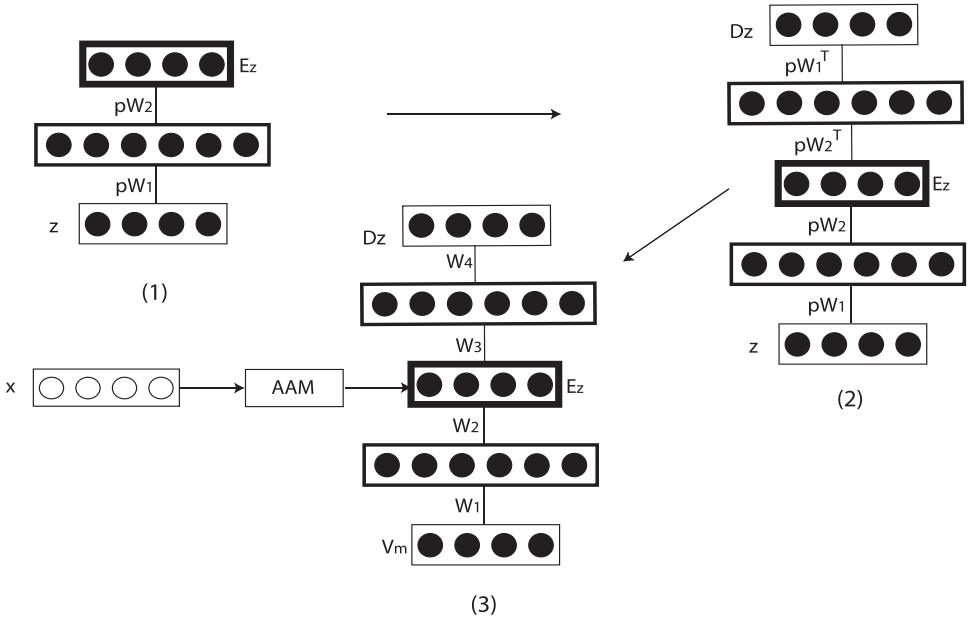

Fig. 2. AE-based creation of a new articulatory feature set for AAM. (1) A 2-layer DBN is trained on the articulatory domain **z**. (b) The DBN is "unfolded" by transposing the weight matrices pW1 and pW2. The unfolded DBN is trained, through backpropagation, to minimize the Euclidean distance between the articulatory input vector **z** and the autoencoder output vector **Dz** (i.e., decoded **z**). (c) The final **Ez** feature set (i.e., the encoded **z**) is used as target in the AAM. **x** represents the acoustic space.

The use of DAEs is partly motivated by the fact that EMA measurements, which are the measurements used in the present work, are noisy, both because of intrinsic noise in the coil trajectory measurements and because of occasional displacements of the coils (Richmond et al., 2011).

3.2.1. Supervised autoencoders

Both standard and denoising AEs are trained in an unsupervised fashion. However some supervised information can be used to force the AE to only learn segmental articulatory feature dependencies, i.e., dependencies that are related to phonetic articulatory targets, and discard all the other dependencies that are related to non-segmental aspects (e.g., supra-segmental aspects, speaker peculiarities, etc). Here we propose two types of supervised AEs, a segmental AE (SAE) and a segmental contractive AE (SCAE).

The rationale behind the first supervised AE, SAE, is largely inspired by DAEs. When training a SAE, randomly selected input vectors are substituted by vectors that belong to the same phone state s, i.e., x_t^s may be transformed, with probability p ($p=0.33$ in the present work), into $\overline{x_t^s} = x_n^s$ (where t and n specify some points in time) while the training objective is kept unaltered ($||\mathbf{x}_t - \mathbf{y}_t||^2$, Fig. 3b). The working assumption is that the substitution $x_t^s \rightarrow x_n^s$ that we (randomly) apply, forces the AE to learn dependencies that most characterize that phonetic unit and remove the phonetically irrelevant differences.

We experimented with two different strategies to select x_n^s, which turned out to produce almost identical results. In the first strategy, x_n^s is the next vector if it falls within the same phone state, i.e., $x_n^s = x_{t+1}^s$ (otherwise, if $\mathbf{x}_{t+1}$ belongs to a different phone state, then $x_n^s = x_{t-1}^s$). In the second strategy we randomly selected x_n^s.

The SAE training does not necessarily force the autoencoder to have similar encodings for input vectors sharing the same phone state and different encodings for input vectors of different phone states. A training that (rein-)forces such similarity can be a desirable property. The segmental contractive AE that we propose enjoys such property (Fig. 3c).

Using binary encoding units, which have a Bernoulli conditional probability distribution, we can define the SCAE training error function as:

$$E_t = ||x_t - y_t||^2 - \lambda \sum_{i=1}^{H} h_{t,i}^s \log(h_{n,i}^s) + (1 - h_{t,i}^s) \log(1 - h_{n,i}^s) \quad (12)$$

Please cite this article in press as: Badino, L., et al., Integrating articulatory data in deep neural network-based acoustic modeling. Comput. Speech Lang. (2015), http://dx.doi.org/10.1016/j.csl.2015.05.005

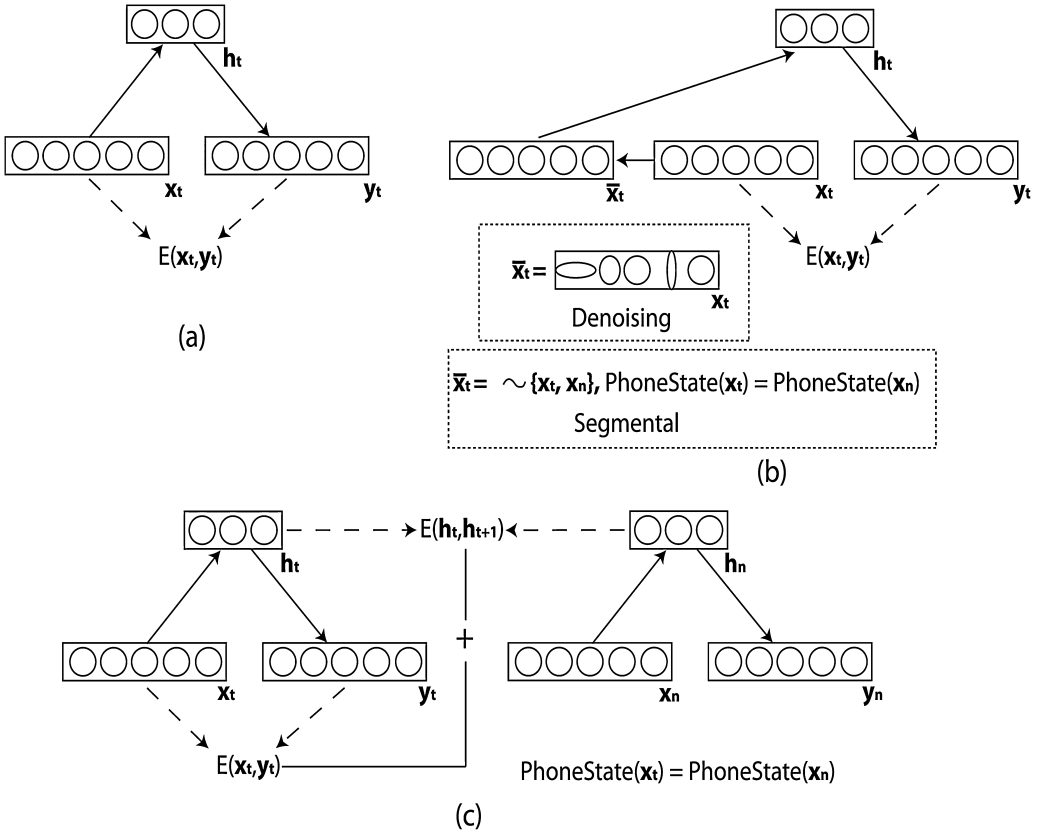

Fig. 3. Autoencoders: (a) standard autoencoder; (b) denoising and segmental autoencoders and (c) segmental contractive autoencoder.

where λ is a constant, H is the number of encoding nodes, and the second term of the error function is a sum of cross-entropies $CE(h_{t,i}, h_{n,i})$.

$CE(h_{t,i}, h_{n,i})$ can also be interpreted as a lower bound of the Kullback–Leibler divergence between two Bernoulli variables with means $h_{t,i}$ and $h_{n,i}$ respectively.

An additional positive term $(\lambda_2 \sum_{i=1}^{H} h^s_{t,i} \log(h^l_{n,i}) + (1 - h^s_{t,i}) \log(1 - h^l_{n,i}))$ could be added to the error function to penalize similar encoding activations for vectors belonging to different phones (rather than phone states). However, that may require to weight phone similarity (meaning a different λ_2 for each phoneme pair). That might explain why, when adding such term, the SCAE training did not converge unless we used very small λ_2 values.

The computation of the partial derivatives $\partial CE(h_{t,i}, h_{n,i})/\partial\theta_{\cdot,i}$, where $\theta_{\cdot,i}$ is an element of the encoding $\mathbf{W}$ matrix or the $\mathbf{b}$ vector affecting the encoding node h_i, is complicated by the fact that a change of $\theta_{\cdot,i}$ affects both $h_{t,i}$ and $h_{n,i}$.

$$\frac{\partial CE(h_{t,i}, h_{n,i})}{\partial \theta_{\cdot,i}} = \frac{\partial h_{t,i}}{\partial \theta_{\cdot,i}} \log\left(\frac{1 - h_{n,i}}{h_{n,i}}\right) + \frac{\partial h_{n,i}}{\partial \theta_{\cdot,i}} \left(\frac{h_{t,i} - h_{n,i}}{h_{n,i}(1 - h_{n,i})}\right) \tag{13}$$

Using sigmoidal activation units for the encoding layer we have:

$$\frac{\partial CE(h_{t,i}, h_{n,i})}{\partial \theta_{\cdot,i}} = \frac{\partial a_{t,i}}{\partial \theta_{\cdot,i}} h_{t,i}(1 - h_{t,i}) \log\left(\frac{1 - h_{n,i}}{h_{n,i}}\right) + \frac{\partial a_{n,i}}{\partial \theta_{\cdot,i}} (h_{t,i} - h_{n,i}) \tag{14}$$

where $h_{t,i} = sigmoid(a_{t,i})$.

4. DNN-HMMs that use articulatory data

The previous section described approaches to learn the AAM. This section describes two strategies that exploit articulatory data for DNN-HMM acoustic modeling. Both strategies require that the AAM is learned first.

Please cite this article in press as: Badino, L., et al., Integrating articulatory data in deep neural network-based acoustic modeling. Comput. Speech Lang. (2015), http://dx.doi.org/10.1016/j.csl.2015.05.005

In a DNN-HMM phone recognition system each phone is modeled as a n-state ($n = 3$ in the present work) hidden Markov model which is typically context-independent (Mohamed et al., 2012). The phone state observation probabilities are approximated by scaling the phone state posteriors (here shortened to phone posteriors) computed by a DNN-based phone (state) classifier. Scaling consists in dividing by the phone state priors.

The two approaches that we implemented in order to exploit measured articulatory data only affect the DNN-based phone classification. The first approach is the well-known approach where AFs are recovered from speech acoustics through AAM and then appended to the observation vector of the DNN phone classifier (Zlokarnik, 1995; Wrench and Richmond, 2000; Badino et al., 2012). Reconstructed AFs are both used when training and testing the phone classifier.

Here we propose an alternative approach which we named AAM-based pretraining. In this approach the use of articulatory data is not direct. The DNN trained to learn the AAM is not used to recover the AFs appended to the observation vector but is instead used to initialize the parameters of the phone classifier DNN. Once a DNN is trained to learn the AAM (1) its topmost layer is removed; (2) a new layer, in which each node has a softmax activation function, is added on top of the net; (3) the net is finetuned to compute phone posteriors. The change of the topmost layer is necessary since both tasks (regression vs. classification) and targets (AFs vs. phone state posteriors) change.

The AAM-based pretraining substitutes the "standard" DBN-based initialization of the phone classifier DNN. The expected difference of the initialization values provided by the two pretraining strategies is due the backpropagation applied to learn the AAM (in the AAM-based pretraining).

In the AAM-based pretraining not only statistical properties of the acoustic domain (given by the DBN-based pretraining of the AAM DNN) but also acoustic–articulatory dependencies are used to drive the search for dependencies between acoustic features and phone classes. Similarly to the appended AFs approach the hypothesis is that phone classification can be improved by reverting the speech production process. Contrary to the appended AFs approach there is no explicit transformation of the acoustic space (into an acoustic + reconstructed articulatory space), however an implicit articulatory-driven transformation is carried out in the hidden layers of the phone classifier DNN.

5. Experimental setup

5.1. Datasets and data pre-processing

We used two British English datasets, the msak0 male voice of MOCHA-TIMIT (Wrench, 2000) and the single male voice mngu0 dataset (Richmond et al., 2011). Both consist of simultaneous recordings of speech and electromagnetic articulographic (EMA) data (plus, in the msak0 case, other types of articulatory data that we did not consider). msak0 is smaller than mngu0, with the first consisting of 460 utterances and the second of 1354 utterances.

EMA data are the x and y positions of upper incisor (UI) (except for the mngu0 corpus), lower incisor (LI), upper lip (UL), lower lip (LL), tongue tip (TT), tongue blade (TB) and tongue dorsum (TD).

Speech was segmented into 25 ms Hamming windows sampled every 10 ms, from which we extracted 20 mel-scaled filterbank coefficients (fbanks) plus their deltas and delta-deltas (for an overall vector of 60 fbanks). Contrary to Badino et al. (2012) we used fbanks as acoustic input for both AAM and phone posterior estimation.

We used 3-state monophones and state boundaries were computed using the HInit, HRest and HERest functions of HTK (Young et al., 1999).

Concerning the articulatory data, we used 42 AFs (36 in the mngu0 dataset) consisting of the x and y trajectories, plus their first and second derivatives. The EMA trajectories were first downsampled (to have a sample every 10 ms as for the acoustic coefficients) and smoothed using an elliptic lowpass filter with 20 Hz cutoff frequency. Then deltas and delta-deltas of the resulting trajectories were computed.

All acoustic and articulatory features were normalized to have 0 mean and unit variance.

To evaluate our systems in noisy speech conditions we corrupted the audio signal by adding 3 different kinds of noise: white Gaussian noise, noise in a cafeteria and noise produced by a subway train. The SNR ranged from 30 to 0 dB. The noisy audio data were generated following the same procedure as for the Aurora-2 database (Hirsch and Pearce, 2000) and using the FANT software (Hirsch, 2005). The SNR was calculated after filtering the clean audio and the noise with the G.712 characteristic. The speech energy was determined using the ITU recommendation P.56.

Please cite this article in press as: Badino, L., et al., Integrating articulatory data in deep neural network-based acoustic modeling. Comput. Speech Lang. (2015), http://dx.doi.org/10.1016/j.csl.2015.05.005

5.2. DNN and AE training

Most of the DNNs trained to learn the AAM received an input of 5 consecutive fbank vectors to reconstruct the AF vector corresponding to the central fbank vector. The 5-frame context is much smaller than that proposed in previous work (Zlokarnik, 1995; Wrench and Richmond, 2000). Since reconstructed AFs may convey information about the acoustic input on the DNN performing AAM, improved phone posterior estimation that uses reconstructed AFs may not be entirely due to articulatory information but also to an implicit larger acoustic context for phone posterior estimation. Our choice of a 5-frame context is a tradeoff between a reduced implicit context for phone posterior estimation and a context that guaranties good AF reconstruction. The number of nodes per hidden layer was manually set to be the same as the number of input nodes ($5 \times 60 = 300$). Since aDNN1 and jDNN1 only differ in terms of learning parameters initialization (i.e., pretraining strategy) they were compared on same-size networks. jDNN2 is the multi-task counterpart of jDNN1 thus resulting in a larger number of learning parameters due to a larger topmost hidden layer (with 600 nodes).

Concerning the DNNs that computed phone posteriors, the input covered a window of 9 consecutive frames where each frame consisted of 60 fbanks plus 42 or 36 AFs (when AFs were used). The DNN had 3 hidden layers with 1500 nodes per layer and 132 (= 44 phonemes $\times$ 3 states) softmax output units.[1]

The input to the deep AEs used to transform the articulatory domain consisted of one single vector of 42 (for msak0) or 36 (for mngu0) AFs. They had a 300-42-300 (or 300-36-300) structure meaning that the hidden layers had 300 nodes each, with the exception of the middle (encoding) layer that had as many nodes as the input and output layers. By constraining the AEs to extract as many features as the input features we ensure that possible improvements due to AE-extracted features were not due to dimensionality reduction effects. Contrary to Badino et al. (2012) the encoding nodes were binary nodes, with values ranging in the [0 1] interval, which were then normalized to have 0 mean and unit variance. In the denoising AE the input nodes were corrupted with Gaussian noise with 0 mean and 0.5 standard deviation. Concerning the segmental AEs, the input vector could be substituted, with probability $p = 0.33$, with a vector sharing the same phone state. In the Segmental Contractive AE λ was set to 0.3 (higher values of λ typically did not guarantee training convergence). Most of the AE's hyperparameters (e.g., Gaussian noise for DAE, p for SAE, etc) were validated by observing the AE reconstruction errors in the first fold validation of the MOCHA-TIMIT msak0 dataset.

The DBN-based pretraining of the DNNs, including AEs, was implemented using a recipe very similar to that proposed in Mohamed et al. (2012). DNNs were pretrained using stochastic gradient descent and a mini-batch size of 100 training cases. RBMs with Gaussian units were trained for 225 epochs and a 0.001 learning rate, RBMs with binary units only were trained for 75 epochs and a 0.1 learning rate. The weight cost was fixed to 0.0002 and the momentum switched from 0.5 to 0.9 after 5 epochs.

When applying AAM-based pretraining to the phone classifier DNN, the phone classifier DNN was not DBN-pretrained, the AAM DNN was used to initialize its learning parameters. In that case the AAM network needs to have the same structure of the phone classifier network, with the exception of the output layer. Thus, for the AAM-based pretraining case, we trained AAM nets that received an input of 9 acoustic vectors and had 1500 nodes per hidden layer. For AAM-based pretraining we only experimented with DNNs trained to reconstruct raw AFs.

Finally, concerning DNN finetuning, the DNNs were fine-tuned with conjugate gradient and batch size of 1000 training cases (which turned out to be the best compromise between DNN performance and training time). The number of training epochs was 100 for AAM and 50 for phone classification. Training epochs were validated on the first fold validation of the MOCHA-TIMIT msak0 dataset where we observed that: (i) after 40 epochs the phone classifier DNNs usually stopped improving (on both training and testing data); (ii) after 100 epochs the regression error reduction of the AAM DNNs was negligible.

When training aDNN1 and the phone classifier DNNs, the weights of the topmost layer were first updated for few epochs (e.g., 5), and then all the net parameters were updated. That was done to better preserve pretraining (Hinton et al., 2006).

To accelerate DNN pretraining and fine-tuning we used the GPUmat Matlab toolbox (GPUmat, 2014) running on Tesla S2050 Graphical Processing Units.

[1] 1500 was approximately the maximum number of nodes per hidden layer that our graphical processing units could sustain.

Please cite this article in press as: Badino, L., et al., Integrating articulatory data in deep neural network-based acoustic modeling. Comput. Speech Lang. (2015), http://dx.doi.org/10.1016/j.csl.2015.05.005

5.3. The DNN-HMM phone recognizer

In the DNN-HMM phone recognizer each phone was represented as a 3-state HMM whose observation probabilities can be approximated by the phone posteriors provided by the DNN-based phone classifier divided by state priors. However, scaling phone posteriors often increased PER, thus the PERs reported in the results section refer to the lowest PER between the scaled and the non-scaled case.

Speech was decoded by feeding the sequence of vectors of DNN estimated state posteriors into a Viterbi decoder. The probabilities of phone unigrams and bigrams used by the Viterbi decoder were computed on the speech training data only (as well as those of state bigrams). The probabilities of phone bigrams were computed using Good-Turing discounting, and back-off for missing bigrams.

Training and evaluation on the MOCHA-TIMIT msak0 voice (consisting of approximately 20 min of speech) was carried out by applying the same 5-fold cross-validation as in Wrench and Richmond (2000) and Badino et al. (2012), while the mngu0 dataset (consisting of approximately 1 h of speech) was divided as in (Richmond et al., 2011) into 1225 training utterances and 65 testing utterances (the validation utterances were excluded).

6. Results

We first show the performance of the proposed DNNs in the AAM task. Subsequently we try to understand what AE-transformed articulatory features represent. Finally we assess the utility of measured articulatory data in (i) speaker-dependent settings, (ii) in cross-speaker settings and (iii) in mismatched environment conditions where the phone recognizer is trained on clean speech and tested in different noisy conditions.

6.1. Articulatory feature reconstruction

The accuracy of the raw articulatory feature reconstruction of the three systems presented in Section 3.1 was evaluated as average root mean square error (*RMSE*) and average Pearson product moment correlation coefficient (r) between reconstructed and actual AFs. Table 1 shows *RMSE* and r values averaged over the full normalized raw articulatory feature set and over the articulatory position features only (excluding upper incisors, for comparison with previous work). *RMSE* on both normalized (0 mean and unit variance) and not normalized (with *RMSE* in millimeters) position features are shown. r was always computed after normalizing features to lie within the [0 1] range.

The 3-hidden layer aDNN1 significantly outperforms the 2-hidden layer aDNN1, e.g., the overall r is significantly larger ($p < 0.01$) according to a two-tailed t-test. That shows that increasing the number of hidden layers increases the reconstruction accuracy, consistently with Uria et al. (2011). The last rows show the reconstruction accuracy of aDNN1, jDNN1 and jDNN2, all having three hidden layers. Both aDNN1 and jDNN1 significantly outperform jDNN2. The poorer performance of jDNN2 might be due to the larger number of parameters that need to be trained. Despite the different pretraining strategies aDNN1 and jDNN1 show similar results. One possible explanation is that, for this particular task and, at least, for this particular DNN configuration, DBN-based pretraining does not seem to be particularly helpful. That is supported by the fact that a DNN whose learning parameters are randomly initialized and not pretrained performs as well as its pretrained counterpart (e.g., no significant RMSE and r differences between

Table 1
Articulatory feature reconstruction accuracy. Except for the last row, values are mean values from a 5-fold cross validation on msak0. 2H and 3H stand for 2-hidden-layer and 3-hidden-layer respectively. Upper teeth are excluded when only position features are considered.

DNN type	rAFs		rAFs (positions)		rAFs (positions)
	RMSE	r	*RMSE*	r	*RMSE* (mm)
2H aDNN1	0.696	0.689	0.605	0.790	1.523
3H aDNN1no pretraining	0.692	0.693	0.600	0.794	1.515
3H aDNN1	0.691	0.693	0.599	0.795	1.510
jDNN1	0.691	0.691	0.603	0.790	1.520
jDNN2	0.694	0.688	0.608	0.787	1.525
3H aDNN1 on mngu0	0.692	0.693	0.490	0.869	0.945

Please cite this article in press as: Badino, L., et al., Integrating articulatory data in deep neural network-based acoustic modeling. Comput. Speech Lang. (2015), http://dx.doi.org/10.1016/j.csl.2015.05.005

the 2nd and the 3rd system in Table 1 on the full articulatory feature set). The 3H aDNN1 compares favourably with almost all previous work (e.g., compare mngu0 results with that reported in Uria et al. (2011) and Uria et al. (2012)) despite using a smaller acoustic context. Performance differences between our DNNs and previous work DNNs may be mostly due to different training settings but it cannot be excluded that different articulatory feature preprocessing strategies might affect the performance.[2]

6.2. Autoencoder-transformed articulatory space

The goal of this section is to understand whether the articulatory features extracted by an AE are able to capture phonetically relevant combined movements/positions of different flesh points.

We carried out a qualitative analysis where we first trained a deep 300-42-300 SAE and then forced the encoding nodes to only get values 0 or 1. That required to set a threshold for each node. Setting a 0.5 threshold for each node would ignore the fact that some nodes tend to fire (i.e., tend to have values close to 1) much more frequently than others. Thus we computed each node threshold as the mean activation of that node.

For each encoding node we compared the positions (velocities and accelerations were not considered in this analysis) of each flesh point. We separated the input samples (each consisting of the 7 flesh point positions) that activated one specific encoding node (i.e., the node took value = 1) from those that did not activate that node. We first observed that the 0–1 difference of an encoding node was almost always associated with changed positions of more than one flesh point, confirming that an encoding node is able to encode the combined position (and movement) of several flesh points.

Additionally, we observed that different encoding nodes exhibited similar position changes. That may be partly due to the fact that the activity of some nodes may be more correlated to velocities and accelerations than to positions, but it can also be due to some co-adaptation of the encoding nodes. Such observation led us to experiment with sparse AEs (with an implementation very similar to, e.g., Le et al., 2011), where a penalty is introduced in the AE training error function to force the AE to simultaneously activate only few encoding nodes. Unfortunately the AFs extracted with sparse AEs turned out to be significantly less successful than features extracted with non-sparse AEs when used for phone recognition. That might be due to the fact that sparse AEs exhibited a larger reconstruction error on testing data which might cause their encoding nodes (i.e., the extracted articulatory features) to miss some important information for phone posterior estimation.

To find out whether the encoding nodes capture phonetic articulatory targets we computed the correlation between each encoding node and each phonetic label using symmetric uncertainty (SI), a normalized and symmetric version of mutual information ($SI(x, y) = 2(H(x) - H(x|y))/(H(x) + H(y))$, where x and y are two random variables and H is the entropy). We also computer SI between raw AFs and phonetic labels. AE-transformed AFs produced higher average SI than raw AFs.

Subsequently, we wanted to examine what phonetic-articulatory features are captured by the nodes that showed the largest correlation values. We selected the two nodes, within the top 5 most correlated nodes, that allowed the best visualization of some distinctive phonetic-articulatory gestures. Fig. 4 shows the average flesh point positions associated to the 0 vs. 1 values of the two nodes, which had a maximum correlation with phones /p/ (Fig. 4a) and /N/ (Fig. 4b) respectively. Comparing the two figures we see that in Fig. 4a changes of lip positions (which are critical for the production of /p/) are much more evident than in Fig. 4b, while changes of the tongue body and back (which are critical vocal tract parts for the production of /N/) are more evident in Fig. 4b.

6.3. Phone recognition

6.3.1. Speaker-dependent evaluation

Table 2 shows the frame-level phone classification error (flPCE) and PER of different phone recognition systems on msak0 averaged over 5-fold cross-validation splits.

[2] Differences with Canevari et al. (2012) are due to the fact that here we average r and $RMSE$ over AFs as in, e.g., Richmond et al. (2003), Uria et al. (2011) and Uria et al. (2012) while in Canevari et al. (2012) we computed the average reconstruction r and $RMSE$ of a frame of AFs.

Please cite this article in press as: Badino, L., et al., Integrating articulatory data in deep neural network-based acoustic modeling. Comput. Speech Lang. (2015), http://dx.doi.org/10.1016/j.csl.2015.05.005

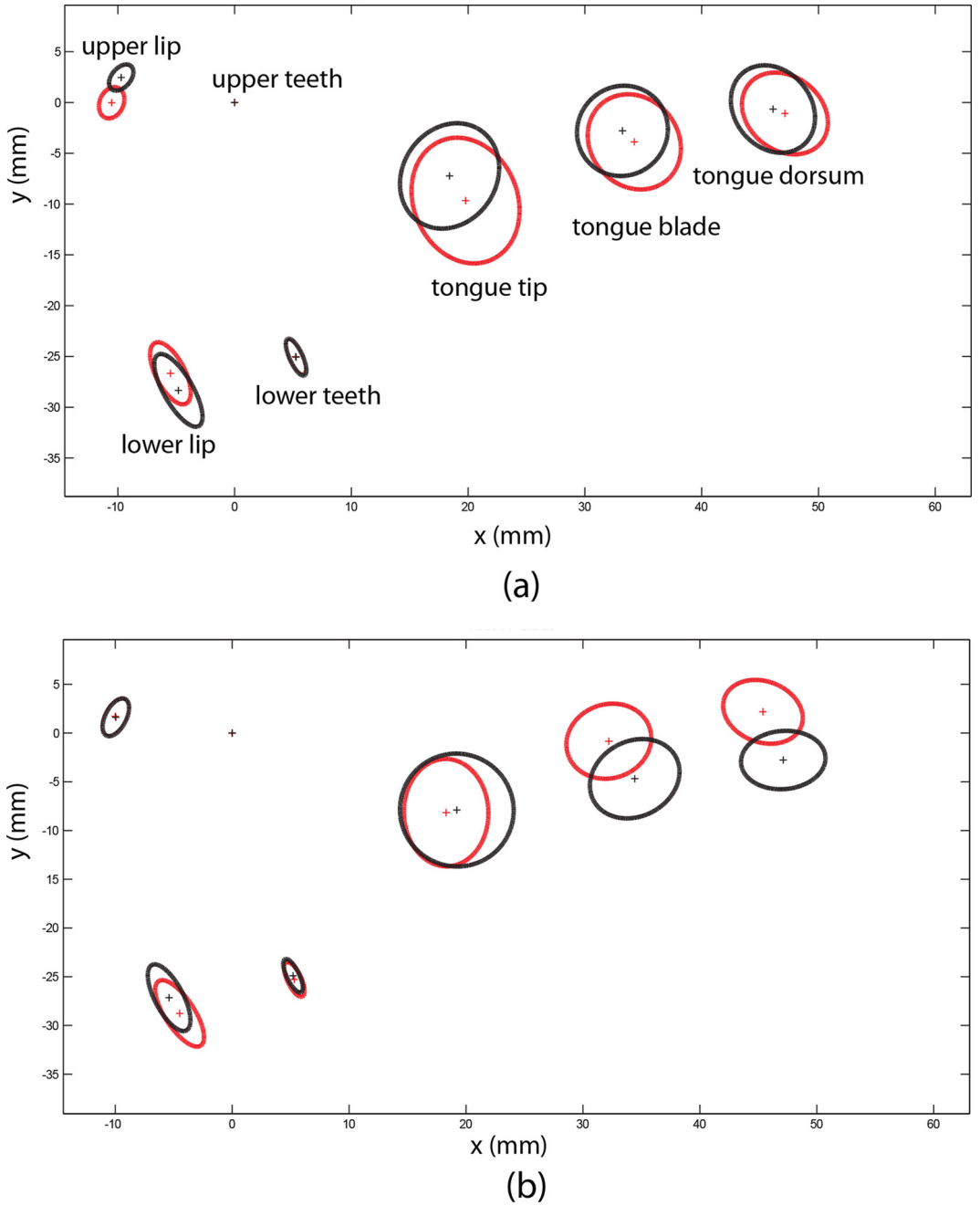

Fig. 4. Flesh point position changes associated to two encoding nodes. The figure shows the sagittal plane of a vocal tract. Cross and ellipse are mean and contour respectively of a Gaussian pdf that models the distribution of the positions of an articulator flesh point. Contours represent 50% of the probability mass. Red contours refer to encoding node value = 1, black contours to encoding node value = 0. (a) Position changes associated to an encoding node correlated with the bilabial plosive /p/. (b) Position changes associated to an encoding node correlated with the nasal velar /N/. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

The comparison between the first two acoustic systems shows the effect of the DBN-based pretraining on the phone classifier DNN. The first system is our (DBN-pretrained) acoustic baseline system, the second system is the non-pretrained acoustic baseline. We have also introduced a third acoustic system that takes as input both the fbanks vectors and their AE transformed version, i.e., bottleneck (BN) features extracted using an 300-42-300 AE as for AE-transformed AFs. This third system exactly matches the fbanks + AFs setup used for the "articulatory" systems.

Please cite this article in press as: Badino, L., et al., Integrating articulatory data in deep neural network-based acoustic modeling. Comput. Speech Lang. (2015), http://dx.doi.org/10.1016/j.csl.2015.05.005

Table 2
Speaker-dependent evaluation on msak0. flPCE and PER stand for frame-level phone classification error and phone error rate respectively. 2H and 3H stands for 2-hidden-layer and 3-hidden-layer respectively. BN stands for bottleneck acoustic features. rAFs are raw AFs while aAFs are AE-transformed AFs.

Feature set	AAM DNN	AE transformaton	flPCE (%)	PER
fbanks	–	–	32.0	30.0
fbanks (no pretraining)	–	–	33.4	31.2
fbanks + BN features	–	–	32.9	31.0
fbanks (AAM-based pretrain.)	3H aDNN1	–	31.3‡	29.2
fbanks + real rAFs	–	–	25.1‡	22.5‡
fbanks + rec. rAFs	2H aDNN1	–	29.3‡	28.4†
fbanks + rec. rAFs	3H aDNN1	–	29.2‡	28.2‡
fbanks + rec. rAFs	3H jDNN1	–	29.2‡	28.1‡
fbanks + rec. rAFs	3H jDNN2	–	29.3‡	28.3†
fbanks + rec. aAFs	2H aDNN1	AE	28.8‡,*	27.7‡
fbanks + rec. aAFs	2H aDNN1	DAE	28.6‡,*	27.5‡,*
fbanks + rec. aAFs	2H aDNN1	SCAE	28.9‡,*	27.7‡
fbanks + rec. aAFs	2H aDNN1	SAE	28.6‡,**	27.5‡

† Significantly better than the best acoustic baseline (first system in the table) with $p < 0.05$ according to a two-tailed t-test.
‡ Significantly better than the best acoustic baseline (first system in the table) with $p < 0.01$ according to a two-tailed t-test.
* Significantly better than the best performing system using rAF features with $p < 0.05$.
** Significantly better than the best performing system using rAF features with $p < 0.01$.

Similarly to Badino et al. (2012) and Canevari et al. (2012) results clearly show that appending reconstructed AFs significantly reduces PER (according to a two-tailed t-test) with a maximum reduction from 30.0 (best acoustic baseline) to 27.5. A perfect reconstruction of the raw AFs would produce a 25% relative PER reduction.

The DNN type and pretraining strategy for AAM do not have a significant impact on PER.

On the other hand the use of AE-transformed AFs further reduces flPCE and PER. The additional flPCE reduction is always significant, independently of the AE used, while the PER additional reduction turned out to be significant when a denoising AE was used.

Finally, the second strategy used to exploit articulatory data for phone classification, i.e., the AAM-based pretraining, produces higher PER than the appended AFs strategy but significantly outperforms the non-pretrained acoustic baseline (29.2 vs. 31.2 PER, 31.3 vs. 33.4 flPCE) and produces a significant smaller flPCE than the acoustic baseline (31.3 vs. 32.0 flPCA).

Some of the systems evaluated on msak0 where also trained and tested on mngu0 (Table 3). Results confirmed the utility of reconstructed AFs, with a maximum 10.1% relative PER reduction achieved with DAE-transformed AFs.

Table 3
Speaker-dependent evaluation on mngu0.

Feature set	AAM DNN	AE transformation	flPCE (%)	PER
fbanks	–	–	16.4	12.9
fbanks (no pretraining)	–	–	17.4	13.1
fbanks + real rAFs	–	–	12.5‡	10.7‡
fbanks + rec. rAFs	3H aDNN1	–	14.5‡	11.8†
fbanks + rec. aAFs	3H aDNN1	DAE	14.0‡,*	11.6‡

† Significantly better than the acoustic baseline (first system in the table) with $p < 0.05$ according to the bootstrap-based significance test proposed in Bisani and Ney (2004).
‡ Significantly better than the acoustic baseline (first system in the table) with $p < 0.01$ according to the bootstrap-based significance test proposed in Bisani and Ney (2004).
* Significantly better ($p < 0.05$) than the best performing system using rAF features.

Please cite this article in press as: Badino, L., et al., Integrating articulatory data in deep neural network-based acoustic modeling. Comput. Speech Lang. (2015), http://dx.doi.org/10.1016/j.csl.2015.05.005

Table 4
Effects on flPCE and PER of the acoustic context used for phone posterior estimation and AFs reconstruction.

Feature set	No. of acoustic vectors of AAM	flPCE (%)	PER
9 fbanks	–	32.0	30.0
13 fbanks	–	31.4	29.6
9 fbanks + rec. rAFs	5	29.2‡	28.2‡
9 fbanks + rec. rAFs	1	31.1‡	29.0‡

‡ Significantly better (with $p < 0.01$) than the system using 13 fbank input vectors.

Table 5
flPCE and PER of speaker-dependent phone recognizers trained on msak0 and tested on mngu0.

Feature set	AAM DNN	AE transformation	flPCE (%)	PER
fbanks	–	–	56.2	58.9
fbanks + rec. rAFs	2H aDNN1	–	54.6‡	58.0†
fbanks + rec. aAFs	2H aDNN1	DAE	55.1	58.3
fbanks + rec. aAFs	2H aDNN1	SAE	55.5	58.9

† Significantly better than the acoustic baseline with $p < 0.05$.
‡ Significantly better than the acoustic baseline with $p < 0.01$.

All flPCE and PER reductions were significant according to the bootstrap-based significance test of Bisani and Ney (2004).[3]

There might be a possibility that the PER reduction produced by the reconstructed AFs is due to the fact that their use implies an implicitly larger acoustic context. Each AF vector is reconstructed from a window of 5 acoustic vectors so it might contain information about a 5-vector acoustic context. That means that when the phone classifier uses 9 vectors of reconstructed AFs it might implicitly observe a 13 acoustic vector context (9 + 2 vectors due to the first AF vector + 2 vectors due to the last AF vector). To see if the utility of reconstructed AFs was mainly due to such enlarged acoustic context we considered a system that had 13 fbank vectors as input and no AFs, and one system that used AFs reconstructed from one single fbank vector (where the shorter acoustic context negatively affected AAM accuracy with, a r value drop from 0.693 to 0.608).

Results on msak0 (Table 4) clearly exclude the possibility that the PER reduction produced by the reconstructed AFs is mainly due to an implicitly larger acoustic context.

6.3.2. *Cross-speaker evaluation*

We carried out two kinds of cross-speaker evaluation. No speaker adaptation nor normalization were used. In the first evaluation some of the speaker-dependent systems trained on msak0 were tested on mngu0. When performing the AAM (learned on msak0) on mngu0 acoustics we try to recover the AFs of a speaker from other's speech acoustics. Table 5 shows that, despite such attempt produces poor reconstruction (Ghosh and Narayanan, 2011; Canevari et al., 2013b), reconstructed AFs can still reduce flPCE and PER. Note that since the phone sets of the two datasets are different we had to map the mngu0 phone set onto the msak0 phone set.

Considering the very limited availability of articulatory data, a more interesting question is whether measured articulatory information can be successfully combined with large acoustic-only datasets to improve acoustic modeling.

We addressed such question by testing our DNN-HMM systems on a "speaker portability" setting, a setting proposed in Arora and Livescu (2013) (where GMM-HMM systems where tested). In this setting we trained phone recognizers on the mngu0 dataset using articulatory information from mask0. Note that the two datasets not only have different speakers but also different utterances. We tested the two approaches described in Section 4. In the first approach an AAM DNN trained on msak0 was used to recover AFs from mngu0 speech acoustics. Subsequently the recovered AFs were appended to the mngu0 acoustic observation vectors thus creating the new acoustic-articulatory mngu0 dataset on which the phone classifier DNN was trained and tested.

[3] We could not apply the t-test because we did not use cross-validation on mngu0.

Please cite this article in press as: Badino, L., et al., Integrating articulatory data in deep neural network-based acoustic modeling. Comput. Speech Lang. (2015), http://dx.doi.org/10.1016/j.csl.2015.05.005

Table 6
flPCE and PER of phone recognizers trained on mngu0 using msak articulatory information. All bootstrap-based significance tests between the baseline (2nd system) and the phone recognizers that use articulatory information (last 3 systems) did not show any significant difference.

Features set	Pretraining	flPCE (%)	PER
mngu0 fbanks	No pretraining	17.4	13.1
mngu0 fbanks	DBN pretraing	16.4	12.9
mngu0 fbanks + rec. msak0 rAFs	DBN pretraing	16.5	13.5
mngu0 fbanks	msak0 AAM DNN with aDNN1	16.3	12.6
mngu0 fbanks	msak0 AAM DNN with jDNN1	16.1	12.8

The second approach is the AAM-based pretraining where the same AAM DNN was used to initialize the learning parameters of the phone classifier DNN trained on mngu0 acoustic data.

Table 6 shows that the AAM-based pretraining approach slightly outperforms the baseline both in terms of flPCE and PER. The appended AFs approach performs worse than the baseline and is significantly outperformed by the AAM-based pretraining approach (significant PER difference with $p = 0.014$).

6.3.3. Noise robustness

Fig. 5 shows the impact of reconstructed raw and AE transformed AFs in three different noisy conditions: additive Gaussian noise, cafeteria noise and subway noise. The DNNs performing AAM and phone posterior estimation were both trained on clean speech. In general AFs reduce PER when the SNR is larger or equal to a 10 dB SNR, but usually degrade performance when SNR is smaller. Even in noisy conditions the AE transformed AFs produce lower PER than the raw AFs.

With the aim of understanding the behaviour of AFs over clean and noisy conditions we trained AEs with 2 encoding nodes on 3 domains: fbanks, "fbanks + reconstructed raw AFs", and "fbanks + actual AFs" (which correspond to perfectly reconstructed AFs). Fig. 6 shows the 2-dimensional representation of 7 msak0 vowels over the 3 different

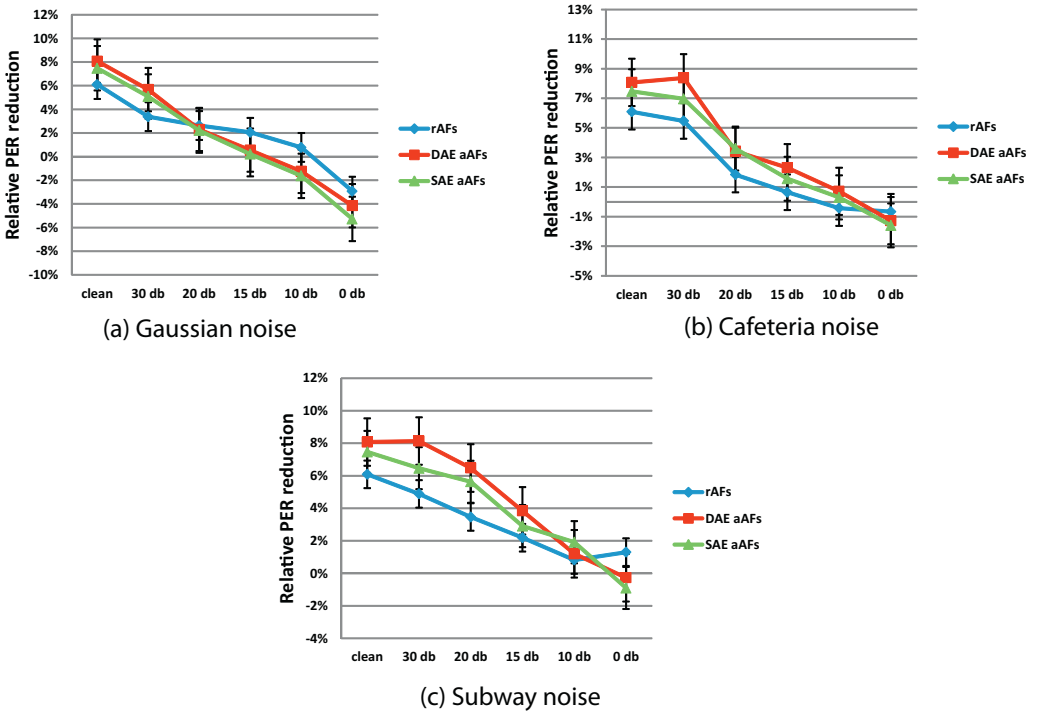

Fig. 5. Relative PER reduction with respect to the acoustic baseline produced by three different articulatory feature sets in three noisy conditions at different SNRs. Error bars show the standard error over the 5-fold cross validation splits.

Please cite this article in press as: Badino, L., et al., Integrating articulatory data in deep neural network-based acoustic modeling. Comput. Speech Lang. (2015), http://dx.doi.org/10.1016/j.csl.2015.05.005

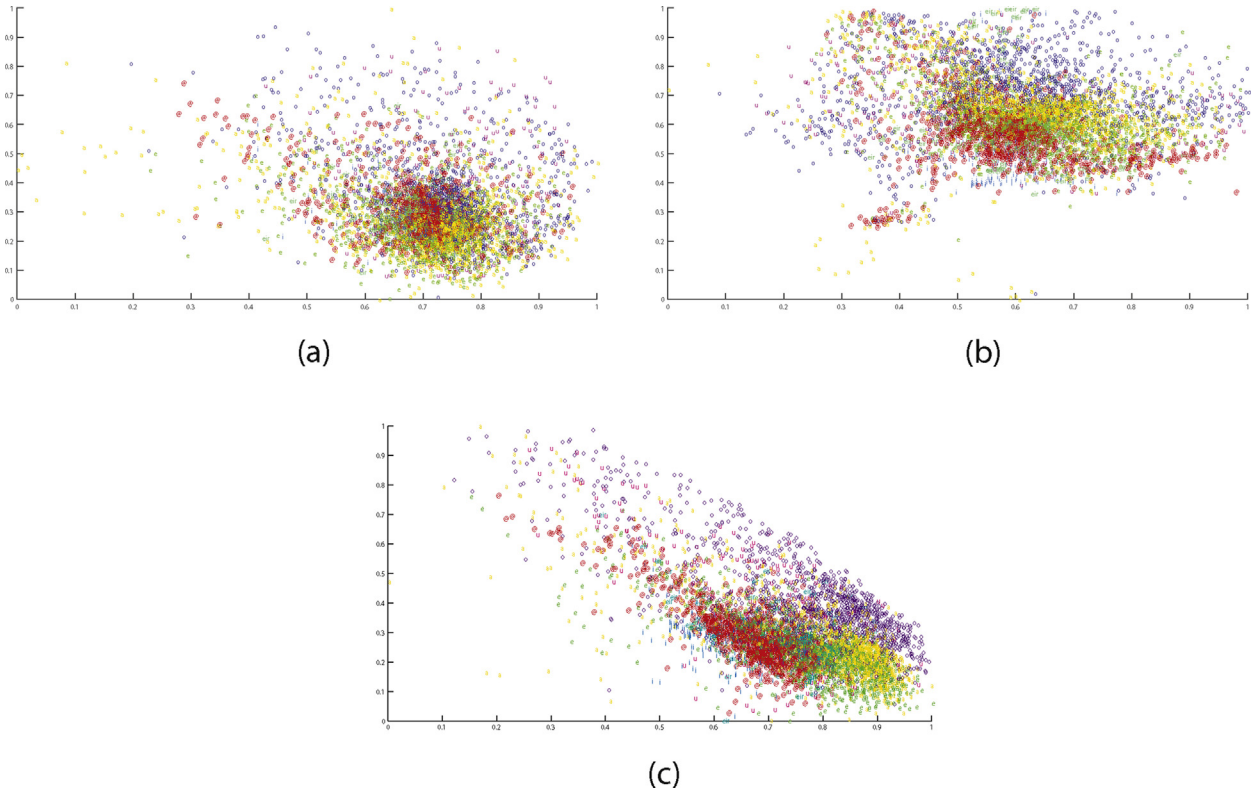

Fig. 6. A 2-dimensional representation of 7 vowels from the msak voice over 3 domains: (a) fbanks; (b) fbanks + reconstructed rAFs; (c) fbanks + actual rAFs. The 2 dimensional representations were obtained by training three 300-200-100-2-100-200-300 AEs on the 3 domains.

domains. In all 3 representations some vowels are not clearly distinguishable but it is easy to see that the vowel boundaries are much sharper in the "fbanks + actual AF" and in the "fbanks + reconstructed AF" 2-D domain than in the fbanks 2-D domain (where they largely overlap), and are sharper in the "fbanks + actual AFs" than in the "fbanks + reconstructed AFs" domain. We observed the same behaviour on all the other phone subsets that we examined (e.g., plosives) and over different AF reconstruction error degrees (up to a certain level of reconstruction error). The full phone set is not shown to allow a good visualization.

Fig. 6 shows that AFs help to better separate phones. The better the AF reconstruction the more evident the phone separation. When AF reconstruction accuracy decreases, because, e.g., of noise, the phone boundaries tend to blur. Up to a certain level of reconstruction error, reconstructed AFs help to better separate phones. Once reconstruction error goes above that point then reconstructed AFs presumably act as additional noise.

7. Discussion

The results we presented show that appending reconstructed AFs to the observation vector improves the phone recognition accuracy of a speaker-dependent DNN-HMM phone recognition system in both clean and noisy speech (Tables 2–5 and Fig. 5). Such results support the hypothesis that, although the extended observation space does not convey any additional information (because the AFs are recovered from acoustics through AAM), the recovered articulatory domain (combined with the acoustic domain) represents a transformation of the acoustic domain into a new speech-production constrained domain where phonetic-articulatory targets can be more easily discriminated.

The utility of the reconstructed features does not seem to only depend on the reconstruction accuracy but also on the strategy used to identify phones given the acoustic-articulatory observations. That is evident when comparing our results with that of Wrench and Richmond (2000), where reconstructed AFs did not improve over the acoustic baseline in a GMM-HMM phone recognition system trained and tested on exactly the same dataset as the one that we used, despite the acoustic baseline performed significantly worse than our best DNN-HMM baseline ($\simeq$37% vs. 30.0%) and

Please cite this article in press as: Badino, L., et al., Integrating articulatory data in deep neural network-based acoustic modeling. Comput. Speech Lang. (2015), http://dx.doi.org/10.1016/j.csl.2015.05.005

the acoustic window used for reconstruction covered a much larger acoustic context (20 vs. 5 acoustic frames and even 20 vs. 1 acoustic frame).

Improving the reconstruction accuracy, either by using a larger acoustic context or by using better strategies to learn the AAM, would likely increase the impact of AFs on recognition accuracy. That is supported by the large PER reduction achieved when using the actual AFs, which corresponds to perfect reconstruction. However, it might be possible that an almost perfect reconstruction turns out to be more difficult to achieve than a perfect phone classification based on acoustics only.

We experimented with different pretraing and training strategies to learn the AAM. Although in the present work we have proposed some novel strategies (Section 3.1), any attempt to use articulatory information in the DNN pretraining to drive the representation of the acoustic domain towards a speech production-constrained representation did not produce improvements over a simpler pretraining that only considers the acoustic domain (Table 1). However, in general, non-pretrained networks, whose learning parameters are randomly initialized before backpropagation, learn the AAM nearly as well as their pretrained counterparts. That suggests that DBN-based pretraining does not seem to be critical for the AAM task. Other directions seems more promising, e.g., weighting the reconstruction error depending on the relevance of an articulator position/movement for the production of a given sound (Canevari et al., 2013a) or using machine learning strategies like deep Mixture Density (Neural) Networks which can handle the non-uniqueness of AAM (Uria et al., 2012).

Not only the learning of the AAM but also the target of the AAM affects the utility of the AFs (Tables 2 and 3). The AE-based articulatory feature extraction that we proposed aims at facilitating the AAM learning. The AE-extracted features can represent dependencies between vocal tract points that correlate with phonetic targets (Fig. 4). Such features should have a more direct relation to speech acoustics than features representing the independent behaviour of each single point. To enforce such properties we have used denoising autoencoders (Vincent et al., 2010) and proposed "supervised" autoencoders that exploit the phonetic label information associated to each articulatory frame. The denoising autoencoder (and, in our view, the supervised autoencoders) may have the additional advantage of extracting AFs that are robust to the noise of the EMA measurements.

Our results show that the best speaker-dependent phone recognition accuracy rates are achieved when using AE-transformed AFs. PER reduction produced by transformed AFs is always larger than the PER produced by non-transformed AFs (Tables 2 and 3) and the difference can be statistically significant (Table 2). In terms of frame-level classification error the difference is always significant.

Such result, as well as the encouraging results obtained with Tract Variables (Mitra et al., 2012) and CCA derived AFs (Arora and Livescu, 2012, 2013, 2014), points out the importance of the representation of the articulatory domain. In future work we will compare CCA-derived features (including features derived from kernel CCA (Arora and Livescu, 2012) and deep CCA (Andrew et al., 2013)) with AAM reconstructed features (both raw and AE transformed) within DNN-HMM phone recognition systems. A main difference between the AAM and the CCA approach is that in our AAM approach only articulatory features are transformed (if autoencoders are used) while in the CCA approach both feature sets are (simultaneously) transformed. Additionally, in the AAM approach, features are transformed before learning the mapping from acoustic to articulatory features (which implicitly increases the correlation between them) while in the CCA approach features are transformed to explicitly maximize correlation. On the other end AE-based encoding allows mechanisms such as denoising. Inspired by Deep CCA we could experiment with an AAM-based approach were both domains are transformed by AEs, while, at the same time, an AAM is learned.

Finally, the utility of the AFs is also constrained by technological limitations in tracking the vocal tract behaviour. The EMA data used in this study as well as in many other studies can only provide partial information about the place of articulation but no information about the manner. If richer articulatory information were available, AFs would presumably produce a lower PER.

For the first time we tested the noise robustness of DNN-HMM acoustic models that integrate AFs (Fig. 5). Results show that appending AFs reduce PER in noisy speech conditions, confirming results of previous studies (e.g., of a dynamic Bayesian network ASR (Mitra et al., 2012), and of a feed-forward neural network binary phone classifier (Castellini et al., 2011)). However they also contradict the same previous studies when showing that the utility of the reconstructed AFs decreases with decreasing SNR. Such (disappointing) difference is probably due to the different strategies employed to exploit articulatory information. For example, in Castellini et al. (2011) the strategy is a decision fusion (rather than a feature fusion as in our case) where phone posteriors were computed by combining the posteriors of two phone classifiers, one trained on acoustic features only and one trained on reconstructed AFs only. For that

Please cite this article in press as: Badino, L., et al., Integrating articulatory data in deep neural network-based acoustic modeling. Comput. Speech Lang. (2015), http://dx.doi.org/10.1016/j.csl.2015.05.005

simple phone binary classification task (/p/ vs. /t/) all the critical articulatory information necessary to discriminate between the two phones was available. That raises the question on whether a decision fusion strategy can still be effective when some critical articulatory information is missing and calls for a comparison between decision fusion and feature fusion strategies in DNN-HMM acoustic modeling.

In general, appending AFs seems to reduce overfitting and thus increases the ability of the phone classifier DNN to generalize to unseen examples. In the first cross-speaker evaluation where we trained speaker-dependent systems on msak0 and tested them on mngu0, the use of reconstructed AFs always outperformed the baseline, although in this case raw AFs were more effective than AE-transformed AFs (Table 5).

Other studies have shown that measured articulatory data can improve speaker-independent neural network phone classifiers (Castellini et al., 2011; Canevari et al., 2013b), BN-HMM (Markov et al., 2006) and GMM-HMM (Arora and Livescu, 2013) phone recognizers. However the utility of measured articulatory data is largely limited by the fact that recording articulatory data is much more difficult than simply recording the audio of a speaker. That has largely constrained the size and number of the available corpora of articulatory data.

Considering such limited availability is it possible to successfully combine few measured articulatory information with large acoustic-only datasets to improve acoustic modeling? Or, once we have large acoustic training datasets that cover a large speech variability then speech articulatory data from small datasets like MOCHA-TIMIT or the Wisconsin X-rays dataset cannot be helpful anymore?

Similarly to Arora and Livescu (2013) we tested our recognition systems in a "speaker portability" setting. We built phone recognizers that were trained on the acoustic-only data of mngu0 and at the same time used an AAM DNN trained on msak0. The AAM DNN was either used to reconstruct msak0 AFs from mngu0 acoustics or to pretrain the phone classifier DNN then finetuned on mngu0 acoustics (AAM-based pretraining). Although mngu0, contrary to, e.g., TIMIT, is a single-speaker dataset, it is three times larger than msak0 and the phone recognition accuracy of its DNN-HMM acoustic baseline is much higher than a DNN-HMM system trained on TIMIT. That makes improvements over the baseline particularly challenging.

While the reconstructed AFs approach failed to perform as well as the acoustic baseline, the AAM-based pretraining approach slightly outperformed it (Table 6).

The 2.3% PER reduction produced by the AAM-based pretraining is not statistically significant. However, the AAM-based pretraining consistently outperforms the baseline in both the same-speaker case (AAM and phone classifier both trained on msak0, Table 2) and the cross-speaker case (Table 6). Although these results should be considered preliminary we consider AAM-based pretraining as a promising alternative approach for ASR systems that use measured articulatory data. In future work we will test the utility of AAM-based pretraining on large multi-speaker acoustic datasets which may require to learn a speaker-independent AAM (Ghosh and Narayanan, 2011; Hueber et al., 2012).

7.1. Conclusion

We aim at improving DNN-HMM acoustic models that use measured articulatory data. Such acoustic models require that an acoustic-to-articulatory mapping is learned first. In order to improve the accuracy of the AAM we have proposed variants to DNN pretraining and training strategies proposed in Badino et al. (2012). Additionally we have "facilitated" the learning of the AAM by transforming the articulatory space on which speech acoustics are mapped. Such transformation was carried out by using deep autoencoders. Autoencoders extract new articulatory features that can capture inter-articulator coordinated movements (Fig. 4). Compared to the independent movements of each articulator, which represent the original articulatory space, the inter-articulator coordinated movements should have a more direct relation to speech sounds. To enforce such property we have used denoising autoencoders (Vincent et al., 2010) and proposed for the first time "supervised" autoencoders that exploit the phonetic label information associated to each articulatory frame.

Measured articulatory information was integrated in our DNN-HMM phonetic recognition systems following two different approaches. In the first approach ("recovered AFs" approach), articulatory features (either 'raw' or autoencoder-transformed) are recovered through AAM and then appended to the observation vector of the DNN that computes phone state posteriors. The second approach ("AAM based pretraining") is a novel approach where the AAM DNN is not used to recover articulatory features but to initialize the learning parameters of the DNN that computes phone state posteriors.

Please cite this article in press as: Badino, L., et al., Integrating articulatory data in deep neural network-based acoustic modeling. Comput. Speech Lang. (2015), http://dx.doi.org/10.1016/j.csl.2015.05.005

Evaluations on the MOCHA-TIMIT msak0 dataset (Table 2) and the mngu0 dataset (Table 3) show that the "recovered AFs" approach significantly reduces PER, with a maximum relative 10.1% PER reduction achieved with autoencoder-transformed articulatory features.

The "recovered AFs" approach was also tested in different noisy conditions (Fig. 5). The recovered AFs increase noise-robustness but such increase reduces with decreasing SNR. That is an unexpected result that partly contradicts related previous work (e.g., Mitra et al., 2012; Castellini et al., 2011).

Recovered AFs are successful even in the case where a speaker-dependent system is tested on a different voice (Table 5). However they are not useful and even worsen performance in a cross-speaker setting, named "speaker portability", where we tested whether it is possible to successfully use measured articulatory information from one speaker to improve a recognition system trained on the only acoustic data of a different speaker (Table 6). Such "speaker portability" setting addresses the question whether the little available measured articulatory information can be combined with large acoustic-only datasets to improve acoustic modeling.

In such setting "AAM based pretraining" performs significantly better than the "recovered AFs" approach and outperforms the acoustics-only baseline with a relative 2.3% PER reduction (Table 6). Although such improvement is not statistically significant, the fact that the AAM-based pretraining consistently outperforms the baseline in both the same-speaker case (AAM and phone classifier both trained on the same dataset, Table 2) and the cross-speaker case makes it a promising alternative approach for ASR systems that use measured articulatory data.

Acknowledgements

The authors acknowledge the support of the European Commission project POETICON++ (grant agreement 288382). The authors would like to thank the anonymous reviewers for their valuable comments and Marco Jacono and Alessandro Bruchi for their support on GPUs and CUDA.

References

Andrew, G., Arora, R., Bilmes, J., Livescu, K., 2013. Deep canonical correlation analysis. In: Proceedings of the 30th International Conference on Machine Learning (ICML).

Arora, R., Livescu, K., 2012. Kernel CCA for multi-view learning of acoustic features using articulatory measurements. In: Symposium on Machine Learning in Speech and Language Processing (MLSLP).

Arora, R., Livescu, K., 2013. Multi-view CCA-based acoustic features for phonetic recognition across speakers and domains. In: Proceedings of the International Conference on Acoustics, Speech and Signal Processing, Vancouver, Canada.

Arora, R., Livescu, K., 2014. Multi-view learning with supervision for transformed bottleneck features. In: Proceedings of the International Conference on Acoustics, Speech and Signal Processing, Florence, Italy.

Badino, L., Canevari, C., Fadiga, L., Metta, G., 2012. Deep-level acoustic-to-articulatory mapping for DBN-HMM based phone recognition. In: Proceedings of IEEE Spoken Language Technology Workshop, Miami, Florida, Erratum available at http://www.rbcs.iit.it/online/badino_et_al_slt2012_erratum.pdf

Bartoli, E., D'Ausilio, A., Berry, J., Badino, L., Bever, T., Fadiga, L., 2013. Listener–speaker perceived distance predicts the degree of motor contribution to speech perception. Cereb. Cortex, http://dx.doi.org/10.1093/cercor/bht257.

Bharadwaj, S., Arora, R., Livescu, K., Hasegawa-Johnson, M., 2012. Multi-view acoustic feature learning using articulatory measurements. In: International Workshop on Statistical Machine Learning for Speech Processing, Kyoto, Japan.

Bisani, M., Ney, H., 2004. Bootstrap estimates for confidence intervals in ASR performance evaluation. In: Proceedings of the International Conference on Acoustics, Speech and Signal Processing, Montreal, Canada.

Browman, C.P., Goldstein, L., 1992. Articulatory phonology: an overview. Phonetica 49 (3–4), 155–180.

Canevari, C., Badino, L., Fadiga, L., Metta, G., 2012. Cross-corpus and cross-linguistic evaluation of a speaker-dependent DNN-HMM ASR system using EMA data. In: Workshop on Speech Production for Automatic Speech Recognition, Lyon, France.

Canevari, C., Badino, L., Fadiga, L., Metta, G., 2013a. Relevance-weighted reconstruction of articulatory features in deep neural network- based acoustic-to-articulatory mapping. In: Proceedings of Interspeech, Lyon, France.

Canevari, C., Badino, L., D'Ausilio, A., Fadiga, L.L., Metta, G., 2013b. Modeling speech imitation and ecological learning of auditory-motor maps. Front. Psychol., http://dx.doi.org/10.3389/fpsyg.2013.00364, 2013.

Castellini, C., Badino, L., Metta, G., Sandini, G., Tavella, M., Grimaldi, M., Fadiga, L., 2011. The use of phonetic motor invariants can improve automatic phoneme discrimination. PLoS ONE 6 (9), e24055, http://dx.doi.org/10.1371/journal.pone.0024055.

Chomsky, N., Halle, M., 1968. The Sound Pattern of English. Harper and Row, New York.

Dahl, G.E., Yu, D., Deng, L., Acero, A., 2012. Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. IEEE Trans. Audio Speech Lang. Process. 20 (1), 30–42.

DAusilio, A., Pulvermller, F., Salmas, P., Bufalari, I., Begliomini, C., Fadiga, L., 2009. The motor somatotopy of speech perception. Curr. Biol. 19, 381–385.

Galantucci, B., Fowler, C.A., Turvey, M.T., 2006. The motor theory of speech perception reviewed. Psychonom. Bull. Rev. 13, 361–377.
Ghosh, P.K., Narayanan, S.S., 2011. A subject-independent acoustic-to-articulatory inversion. In: Proceedings of ICASSP, Prague.
GPUmat 3.0. Available at http://sourceforge.net/projects/gpumat/ (accessed 04.06.2014).
Graves, A., Mohamed, A., Hinton, G.E., 2013. Speech recognition with deep recurrent neural networks. In: Proceedings of ICASSP, Vancouver, Canada.
Grimaldi, M., Gili Fivela, B., Sigona, F., Tavella, M., Fitzpatrick, P., Craighero, L., Fadiga, L., Sandini, G., Metta, G., 2008. New technologies for simultaneous acquisition of speech articulatory data: 3D articulograph, ultrasound and electroglottograph. In: Proceedings LangTech, Rome, Italy.
Hinton, G.E., 2002. Training products of experts by minimizing contrastive divergence. Neural Comput. 14, 1771–1800.
Hinton, G.E., Sejnowski, T.J., 1986. Learning and relearning in Boltzmann machines. In: Rumelhart, D.E., McClelland, J.L. (Eds.), Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Vol. 1: Foundations. MIT Press, Cambridge, MA, pp. 282–317.
Hinton, G.E., Salakhutdinov, R.R., 2006. Reducing the dimensionality of data with neural networks. Science 313 (5786), 504–507.
Hinton, G.E., Osindero, S., Teh, Y., 2006. A fast learning algorithm for deep belief nets. Neural Comput. 18, 1527–1554.
Hirsch, H.G., 2005. Fant – Filtering and Noise Adding Too, Available at http://dnt.kr.hsnr.de/download.html (Accessed 30.03.014).
Hirsch, H.G., Pearce, D., 2000. The aurora experimental framework for the performance evaluation of speech recognition systems under noisy conditions. In: Proceedings of ISCA ITRW ASR.
Hueber, T., BenYoussef, A., Bailly, G., Badin, P., Elisei, F., 2012. Cross-speaker acoustic-to-articulatory inversion using phone-based trajectory HMM for pronunciation training. In: Proceedings of Interspeech, Portland, USA.
Jakobson, R., Fant, G., Halle, M., 1952. Preliminaries to Speech Analysis: The Distinctive Features and Their Correlates. MIT Press, Cambridge, MA.
King, S., Frankel, J., Livescu, K., McDermott, E., Richmond, K., Wester, M., 2007. Speech production knowledge in automatic speech recognition. J. Acoust. Soc. Am. 121 (2), 723–742.
Le, Q.V., Ngiam, J., Coates, A., Lahiri, A., Prochnow, B., Ng, A.Y., 2011. On optimization methods for deep learning. In: Proceedings of the International Conference on Machine Learning, Bellevue, Washington, USA.
Liberman, A.M., Cooper, F.S., Shankweiler, D.P., Studdert-Kennedy, M., 1967. Perception of the speech code. Psychol. Rev. 74, 431–461.
Lindblom, B., Lubker, J., Gay, T., 1979. Formant frequencies of some fixed-mandible vowels and a model of speech motor programming by predictive simulation. J. Phonet. 7, 146–161.
Maddieson, I., 1997. Phonetic universals. In: Hardcastle, W.J., Laver, J. (Eds.), The Handbook of Phonetic Sciences. Blackwell Publishers, Oxford, pp. 619–639 (Chapter 20).
Markov, K., Dang, J., Nakamura, S., 2006. Integration of articulatory and spectrum features based on the hybrid HMM/BN modeling framework. Speech Commun. 48, 161–175.
Mitra, V., Nam, H., Espy-Wilson, C., Saltzman, E., Goldstein, L., 2010. Retrieving tract variables from acoustics: a comparison of different machine learning strategies. IEEE J. Sel. Top. Signal Process. 4 (6), 1027–1045.
Mitra, V., Nam, H., Espy-Wilson, C., Saltzman, E., Goldstein, L., 2011. Speech inversion: benefits of tract variables over pellet trajectories. In: Proceedings of the International Conference on Acoustics, Speech and Signal Processing, Prague, Czech Republic.
Mitra, V., Nam, H., Espy-Wilson, C., Saltzman, E., Goldstein, L., 2012. Recognizing articulatory gestures from speech for robust speech recognition. J. Acoust. Soc. Am. 131 (3), 2270–2287.
Mohamed, A., Dahl, G.E., Hinton, G.E., 2012. Acoustic Modeling using Deep Belief Networks. IEEE Trans. on Audio, Speech, and Language Processing 20 (1), 14–22.
Narayanan, S., Nayak, K., Lee, S., Sethy, A., Byrd, D., 2004. An approach to real-time magnetic resonance imaging for speech production. J. Acoust. Soc. Am. 115, 1771–1776.
Ngiam, J., Nam, M., Lee, J., Khosla, H., Kim, A., Ng, A.Y., 2011. Multimodal deep learning. In: Proceedings of the International Conference on Machine Learning, Bellevue, Washington, USA.
Pearce, D., Hirsch, H.G., 2000. The aurora experimental framework for the performance evaluation of speech recognition systems under noisy conditions. In: Proceedings of the International Conference on Spoken Language Processing, Beijing, China.
Qin, C., Carreira-Perpiñán, M.A., 2007. An empirical investigation of the nonuniqueness in the acoustic-to-articulatory mapping. In: Proceedings of Interspeech, Antwerp, Belgium.
Richmond, K., 2006. A trajectory mixture density network for the acoustic-articulatory inversion mapping. In: Proceedings of Interspeech, Pittsburgh, USA.
Richmond, K., King, S., Taylor, P., 2003. Modelling the uncertainty in recovering articulation from acoustics. Comput. Speech Lang. 17 (2), 153–172.
Richmond, K., Hoole, P., King, S., 2011. Announcing the electromagnetic articulography (Day 1) subset of the mngu0 articulatory corpus. In: Proceedings of Interspeech, Florence, Italy.
Roweiss, S., 1999. Data Driven Production Models for Speech Processing. California Institute of Technology, Pasadena, CA (Ph.D. thesis).
Schaaf, T., Metze, F., 2010. Analysis of gender normalization using MLP and VTLN features. In: Proceedings of Interspeech, Makuhari, Japan.
Seltzer, M., Yu, D., Wan, Y., 2013. An investigation of deep neural networks for noise robust speech recognition. In: Proceedings of the International Conference on Acoustics, Speech and Signal Processing, Vancouver, British Columbia, Canada.
Smolensky, P., 1986. Information processing in dynamical systems: Foundations of harmony theory. In: Rumelhart, D.E., McClelland, J.L. (Eds.), Parallel Distributed Processing, Vol. 1. MIT Press, Cambridge, pp. 194–281 (Chapter 6).
Stephenson, T., Bourlard, H., Bengio, S., Morris, A., 2000. Automatic speech recognition using dynamic Bayesian networks with both acoustic and articulatory variables. In: Proceedings of the International Conference on Spoken Language Processing, Beijing, China.
Toda, T., Black, A., Tokuda, K., 2007. Statistical mapping between articulatory movements and acoustic spectrum using a Gaussian mixture model. Speech Commun. 50 (3), 215–222.

Please cite this article in press as: Badino, L., et al., Integrating articulatory data in deep neural network-based acoustic modeling. Comput. Speech Lang. (2015), http://dx.doi.org/10.1016/j.csl.2015.05.005

Uria, B., Renals, S., Richmond, K., 2011. A deep neural network for acoustic-articulatory speech inversion. In: Proceedings of NIPS Workshop on Deep Learning and Unsupervised Feature Learning.

Uria, B., Murray, I., Renals, S., Richmond, K., 2012. Deep architectures for articulatory inversion. In: Proceedings of Interspeech, Portland, OR, USA.

Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., Manzagol, P.A., 2010. Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion. J. Mach. Learn. Res. 11, 3371–3408.

Welling, M., Rosen-Zvi, M., Hinton, G.E., 2005. Exponential family harmoniums with an application to information retrieval. In: Advances in Neural Information Processing Systems.

Westbury, J.R., 1994. X-ray Microbeam Speech Production Database User's Handbook. Waisman Center on Mental Retardation and Human Development. University of Wisconsin, Madison, WI, USA, version 1.0 edition.

Wrench, A.A., 2000. Multi-channel/multi-speaker articulatory database for continuous speech recognition research. Phonus 5, 1–13.

Wrench, A.A., Richmond, K., 2000. Continuous speech recognition using articulatory data. In: Proceedings of the International Conference on Spoken Language Processing, Beijing, China.

Young, S., Kershaw, D., Odell, J., Ollason, D., Valtchev, V., Woodland, P., 1999. The HTK Book. Entropic, Cambridge, UK.

Yu, H., Finke, M., Waibel, A., 1999. Progress in automatic meeting transcription. In: Proceedings of Eurospeech, Budapest, Hungary.

Zlokarnik, I., 1995. Adding articulatory features to acoustic features for automatic speech recognition. J. Acoust. Soc. Am. 97 (5), 3246.

Leonardo Badino is a postdoctoral researcher at the Italian Institute of Technology (IIT), Genova, Italy. He received a M.Sc. degree in Electronic Engineering from the University of Genova in 2000 and a Ph.D. in Computer Science from the University of Edinburgh, UK. From 2001 to 2006 he worked as software engineer and project manager at Loquendo, a speech technology company. During his Ph.D. he worked on prosodic prominence detection and generation for text-to-speech synthesis. He is currently working on speech production for ASR, limited resources ASR and computational analysis of non-verbal sensorimotor communication.

Claudia Canevari received a M.SC. degree in Bioengineering and a Ph.D. in "Robotics, Cognition and Interaction Technologies" from the University of Genova, in 2010 and 2014, respectively. Currently she holds a research fellow position at the Robotics, Brain and Cognitive Sciences Department of the Italian Institute of Technology. Her research focuses on measured articulatory data for ASR and involves acoustic-to-articulatory mapping strategies, articulatory signal processing, and creation of new corpora of articulatory data.

Luciano Fadiga, M.D., Ph.D. in Neuroscience. Professor of Physiology at the University of Ferrara and Senior Researcher at the Italian Institute of Technology. He has a long experience in electrophysiology in monkeys (single neurons recordings) and humans (transcranial magnetic stimulation, study of spinal excitability and brain imaging). His current researches are focused on the mirror mechanisms for speech/language understanding and on the realization of brain-computer interfaces specifically designed for human use.

Giorgio Metta is director of the iCub Facility department at the Istituto Italiano di Tecnologia (IIT) where he coordinates the development of the iCub robotic platform/project. He holds an M.Sc. cum laude (1994) and Ph.D. (2000) in electronic engineering both from the University of Genoa. From 2001 to 2002 he was postdoctoral associate at the MIT AI-Lab. His research activities are in the fields of biologically motivated and humanoid robotics and, in particular, in developing humanoid robots that can adapt and learn from experience. He has been working as principal investigator and research scientist in about a dozen EU projects.

Please cite this article in press as: Badino, L., et al., Integrating articulatory data in deep neural network-based acoustic modeling. Comput. Speech Lang. (2015), http://dx.doi.org/10.1016/j.csl.2015.05.005