Integrated speaker-adaptive speech synthesis

Moquan Wan, Gilles Degottex, Mark J.F. Gales

Cambridge University Engineering Department, UK

mw545@cam.ac.uk, gad27@cam.ac.uk, mjfg@eng.cam.ac.uk

Abstract

Enabling speech synthesis systems to rapidly adapt to sound like a particular speaker is an essential attribute for building personalised systems. For deep-learning based approaches, this is difficult as these networks use a highly distributed representation. It is not simple to interpret the model parameters, which complicates the adaptation process. To address this problem, speaker characteristics can be encapsulated in fixed-length speaker-specific Identity Vectors (iVectors), which are appended to the input of the synthesis network. Altering the iVector changes the nature of the synthesised speech. The challenge is to derive an optimal iVector for each speaker that encodes all the speaker attributes required for the synthesis system. The standard approach involves two separate stages: estimation of the iVectors for the training data; and training the synthesis network. This paper proposes an integrated training scheme for speaker adaptive speech synthesis. For the iVector extraction, an attention based mechanism, which is a function of the context labels, is used to combine the data from the target speaker. This attention mechanism, as well as nature of the features being merged, are optimised at the same time as the synthesis network parameters. This should yield an iVector-like speaker representation that is optimal for use with the synthesis system. The system is evaluated on the Voice Bank corpus. The resulting system automatically provides a sensible attention sequence and shows improved performance from the standard approach.

Index Terms: speech synthesis, iVector, integrated, adaptation, attention mechanism

1. Introduction

Adaptive speech synthesis addresses the task of generating speech of an arbitrary speaker's voice [1], and for deep neural network acoustic models, this is a challenging task due to the large number of parameters and connections which are not interpretable and therefore hard to adapt. To address this issue, one standard approach is to encapsulate speaker characteristics in fixed-length speaker-specific Identity Vectors, or *iVectors*, and appending an iVector to the input of the acoustic model alters the nature of the output to represent a particular speaker, without adapting the entire acoustic model [2][3]. There are previous studies on other adaptation methods, such as output feature transformation and learning hidden unit contributions (LHUC) [4]. While they to some extent outperform iVector-based adaptation, they suffer from a large number of adaptation parameters, while the dimension of the iVector is significantly lower, which is critical for intepretation and rapid adaptation. In addition, iVector-based adaptation can be easily applied to a range of deep acoustic models, such as feed-forward deep neural network (DNN), recurrent neural network (RNN), WaveNet [5] or Char2Wav [6]. Therefore, in this work, the focus is to improve the iVector representation for better adaptive synthesis performance.

The traditional approach of iVector-based adaptation involves two distinct stages. First an auxiliary iVector extraction model is optimised and one iVector is estimated for each speaker, and then the iVectors are appended to corresponding inputs for acoustic model training. One of the simplest form of iVector is one-hot speaker code, and many have applied this form of iVectors for adaptation successfully [5][7][8]. The problem with this form is that the speaker codes do not represent any speaker characteristics, and the burden of representing speaker variations dictates a much powerful and expensive acoustic model. Also it is difficult to adapt to a new speaker, since the speaker code cannot be derived from new adaptation data. Another form of iVector is DNN-based *d-vector* [9]. In this framework, an auxiliary DNN is trained to classify each frame to the corresponding speaker, and the d-vector of a speaker is the average of the activations of the last hidden layer (or a bottle-neck hidden layer) across all frames from that speaker. The d-vector solves the problem of adapting to a new speaker, but it suffers a similar problem as speaker codes, that the target outputs of the auxiliary DNN do not represent any speaker characteristics, since they are simply one-hot vectors. Therefore, in this work, the choice of iVector is Gaussian Mixture Model (GMM) based *i-vector* [10][11]. In this framework, a Universal Background Model (UBM) is trained on the acoustic data of all speakers. Next, one GMM is trained for each speaker. The trainings of GMMs are unsupervised, and the mixture components are distributed based on acoustic characteristics, and they are more informative than the arbitrary one-hot speaker codes. Finally, each i-vector is estimated as a low-dimensional compact representation of the difference between the speaker-specific GMM and the UBM. The i-vectors span a space of speaker characteristics, and it is easy to interpolate or interpret in this speaker space. It is also easy to estimate a new i-vector for a new speaker, to train a new speaker-specific GMM, given adaptation data and UBM.

However, in all the above-mentioned approaches and many others, there are three major shortcomings:

1. iVector extraction process is independently trained first, followed by the training of the acoustic model for speech generation, and there are two separate training criteria that can result in two sub-optimal systems. The iVector representations are extracted for generic speaker adaptation tasks, and not all elements in an iVector are equally important for the specific task of adaptive speech synthesis, thus the iVectors can be less efficient.

2. iVector extraction process takes all frames of data as equally important, where actually a large portion of the data do not represent the distinctive characteristics of the speakers. Therefore, the iVectors may not effectively distinguish the speakers;

3. iVector extraction process makes use of the acoustic data only, since it is commonly believed that most of the speaker

characteristics are encapsulated in the acoustic data; however, the corresponding linguistic data (the contextual labels) are wasted, while they can provide additional information about speaker characteristics, or they can assist in weighing different frames of acoustic data in iVector extraction process.

Therefore, in this work, two improvements are proposed for these shortcomings, an integrated training framework for a unified training criteria, and an attention mechanism to automatically select the most speaker-representative parts of speech for iVector extraction. For the attention mechanism, the attentions are derived from the linguistic features. Studies have shown that speaker variability in vowels is more significant and consistent as compared to consonants [12], and we expect the attention mechanism to automatically emphasise more in the vowels.

2. iVector-based Adaptive Speech Synthesis

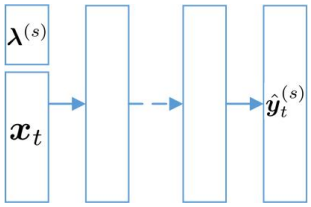

Figure 1: *iVector-based adaptive speech generation*

For iVector-based adaptive speech synthesis systems, at generation stage, a speaker-specific iVector is appended to the linguistic input of the acoustic model, in order to generate acoustic output of the corresponding speaker. For a frame, a mini-batch or an utterance, the speech generation process is as follows:

$$\hat{\boldsymbol{y}}_t^{(s)} = \mathcal{F}\left(\boldsymbol{x}_t, \boldsymbol{\lambda}^{(s)}, \boldsymbol{\theta}_A\right) \tag{1}$$

Here $\boldsymbol{x}_t$ is the linguistic input, $\boldsymbol{\lambda}^{(s)}$ is the iVector of speaker s, and $\hat{\boldsymbol{y}}_t^{(s)}$ is the predicted speaker-specific acoustic output. $\mathcal{F}(\cdot)$ represents the acoustic model with parameters $\boldsymbol{\theta}_A$. (Fig.1)

In this section, three different iVector extraction methods, their corresponding training processes and the dependencies are compared.

2.1. Independent iVector Extraction

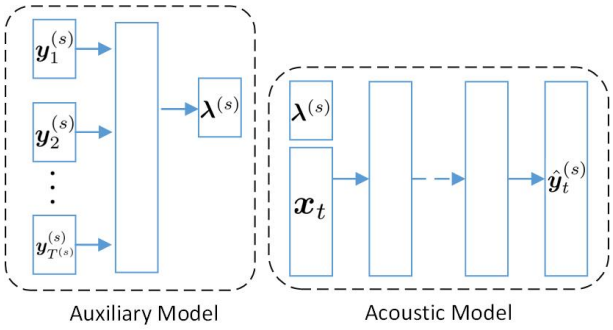

Figure 2: *Independent iVector extraction-based adaptive speech generation, with 2 distinct stages*

In previous studies [4][11], the iVector extraction model is trained independently first before the training of the acoustic model (left of Fig.2). Also at iVector extraction stage, the input of the extraction process includes only the acoustic data:

$$\boldsymbol{\lambda}^{(s)} = \mathcal{G}\left(\boldsymbol{Y}^{(s)}, \boldsymbol{\theta}_{Iy}\right) \tag{2}$$

Here $\boldsymbol{Y}^{(s)} = \{\boldsymbol{y}_{1:T^{(s)}}^{(s)}\}$ represents all acoustic data of speaker s, and $\boldsymbol{\lambda}^{(s)}$ is the iVector of speaker s. $\mathcal{G}(\cdot)$ represents an auxiliary model with parameters $\boldsymbol{\theta}_{Iy}$, and it is optimised completely before the training of the acoustic model, with only dependency on $\{\boldsymbol{Y}^{(s)}\}$.

After the auxiliary model is optimised and the iVectors extracted, the acoustic model $\boldsymbol{\theta}_A$ can be trained with the following gradient expression:

$$\frac{\partial \mathcal{E}}{\partial \boldsymbol{\theta}_A} = \frac{1}{\sum_{\forall s} T^{(s)}} \sum_{\forall s} \sum_{t=1}^{T^{(s)}} \frac{\partial \mathcal{E}_t^{(s)}}{\partial \boldsymbol{\theta}_A}\bigg|_{\boldsymbol{\lambda}^{(s)}, \boldsymbol{x}_t, \boldsymbol{y}_t^{(s)}} \tag{3}$$

Here $\mathcal{E}$ represents the overall cost function to minimise, and $\mathcal{E}_t^{(s)}$ is mini-batch cost. In our experiments, the training criteria is mean-square-error (MSE):

$$\begin{aligned} \mathcal{E} &= \frac{1}{\sum_{\forall s} T^{(s)}} \sum_{\forall s} \sum_{t=1}^{T^{(s)}} ||\hat{\boldsymbol{y}}_t^{(s)} - \boldsymbol{y}_t^{(s)}||^2 \\ &= \frac{1}{\sum_{\forall s} T^{(s)}} \sum_{\forall s} \sum_{t=1}^{T^{(s)}} \mathcal{E}_t^{(s)} \end{aligned} \tag{4}$$

For our baseline model, the auxiliary model is standard GMM-based i-vector extraction method [10][11], where the auxiliary model is trained to maximise the probability of all the data across different speakers, with the following constraint: for each speaker-dependent GMM, the mean supervector $\boldsymbol{\mu}^{(s)}$ of speaker s has the following expression:

$$\boldsymbol{\mu}^{(s)} \approx \boldsymbol{\mu}^{UBM} + \boldsymbol{T}\boldsymbol{\lambda}^{(s)},\ \boldsymbol{\lambda}^{(s)} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}) \tag{5}$$

where $\boldsymbol{\mu}^{UBM}$ is the mean supervector of speaker-independent UBM, and $\boldsymbol{T}$ is the speaker-independent total variability matrix. When the covariance matrices are constrained to identity matrices, these two parameters and the i-vectors are the model parameters to optimise. The optimisation process includes Expectation Maximisation (EM), which is not really compatible with stochastic gradient descent (SGD) for back-propagation, and thus not for the integrated frameworks with deep acoustic models.

2.2. Integrated iVector Extraction

The first improvement is to fine-tune and optimise the auxiliary model $\boldsymbol{\theta}_{Iy}$ alongside the training of the acoustic model $\boldsymbol{\theta}_A$, as the independently extracted iVectors is designed for generic speaker adaptation tasks, and they may not efficiently encapsulate speaker characteristics that are more relevant for speech synthesis tasks, specifically.

Therefore, while the iVector extraction still uses all the acoustic data (Eq.2), the training of $\boldsymbol{\theta}_{Iy}$ depends on the current state of $\boldsymbol{\theta}_A$ as well, and the following is the expression of the partial derivative:

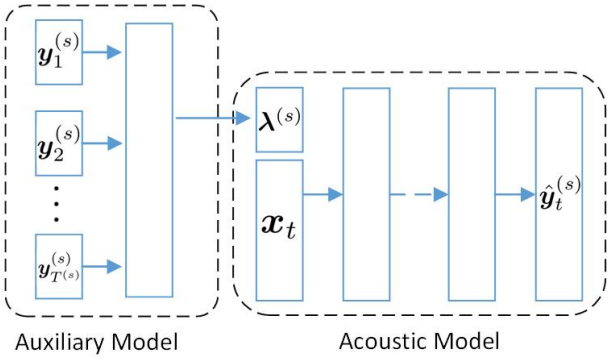

Figure 3: *Integrated iVector extraction-based adaptive speech generation*

$$\frac{\partial \mathcal{E}}{\partial \boldsymbol{\theta}_{Iy}} = \frac{1}{\sum_{\forall s} T^{(s)}} \sum_{\forall s} \sum_{t=1}^{T^{(s)}} \frac{\partial \mathcal{E}_t^{(s)}}{\partial \boldsymbol{\theta}_{Iy}} \Big|_{\boldsymbol{\theta}_A, \boldsymbol{x}_t, \boldsymbol{y}_t^{(s)}, \boldsymbol{Y}^{(s)}} \tag{6}$$

$$= \frac{1}{\sum_{\forall s} T^{(s)}} \sum_{\forall s} \left[\left(\frac{\partial \boldsymbol{\lambda}^{(s)}}{\partial \boldsymbol{\theta}_{Iy}} \Big|_{\boldsymbol{Y}^{(s)}} \right) \sum_{t=1}^{T^{(s)}} \left(\frac{\partial \mathcal{E}_t^{(s)}}{\partial \boldsymbol{\lambda}^{(s)}} \Big|_{\boldsymbol{\theta}_A, \boldsymbol{x}_t, \boldsymbol{y}_t^{(s)}} \right) \right] \tag{7}$$

The partial derivative w.r.t. $\boldsymbol{\theta}_{Iy}$ is propagated via $\boldsymbol{\lambda}^{(s)}$ (Fig.3), and for each mini-batch, the dependencies of this partial derivative include $\{\boldsymbol{Y}^{(s)}, \boldsymbol{\theta}_A, \boldsymbol{x}_t, \boldsymbol{y}_t^{(s)}\}$.

The expression of gradient of the acoustic model $\boldsymbol{\theta}_A$, given $\boldsymbol{\lambda}^{(s)}$, is the same as Eq.3.

For our integrated framework, the auxiliary model is a deep network, and both models can be jointly optimised using SGD and back-propagation. Also, since the number of frames / utterances from each speaker varies, an averaging layer is added on top to handle sequences with various lengths $T^{(s)}$, to find the mean of some output vectors:

$$\boldsymbol{\lambda}^{(s)} = \mathcal{G}\left(\boldsymbol{Y}^{(s)}, \boldsymbol{\theta}_{Iy}\right) = \mathcal{G}\left(\{\boldsymbol{y}_{1:T^{(s)}}^{(s)}\}, \boldsymbol{\theta}_{Iy}\right)$$

$$= \frac{1}{T^{(s)}} \sum_{\tau=1}^{T^{(s)}} \mathcal{G}_{Iy}\left(\boldsymbol{y}_\tau^{(s)}, \boldsymbol{\theta}_{Iy}\right) \tag{8}$$

In our work, the last layer of $\mathcal{G}_{Iy}(\cdot)$ is a linear layer, since the GMM-based i-vectors follow normal distribution and have no range limit.

2.3. Integrated iVector Extraction with Attention Mechanism

Next we introduce the attention mechanism, to automatically select the more representative parts of speech for iVector extraction, as a large portion of the acoustic data do not represent the distinctive characteristics of the speakers. The attentions are derived from the linguistic features i.e. the contextual labels, and the additional information could assist in the selection of the most representative parts of acoustic data. For instance, vowels are naturally more speaker-representative than consonants, and voiced regions are more representative than unvoiced regions. It is expected that these could be automatically derived from the linguistic data.

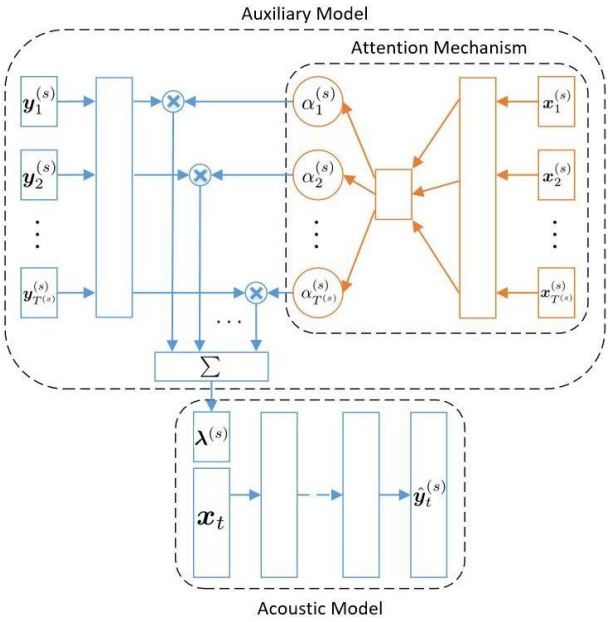

Figure 4: *Integrated iVector extraction-based adaptive speech generation, with attention mechanism (top right)*

Now, the iVector extraction uses all the acoustic data and all the linguistic data:

$$\boldsymbol{\lambda}^{(s)} = \mathcal{G}\left(\boldsymbol{Y}^{(s)}, \boldsymbol{X}^{(s)}, \boldsymbol{\theta}_I\right) \tag{9}$$

$$= \mathcal{G}\left(\boldsymbol{Y}^{(s)}, \boldsymbol{X}^{(s)}, \{\boldsymbol{\theta}_{Iy}, \boldsymbol{\theta}_{Ix}\}\right) \tag{10}$$

And we use $\boldsymbol{\theta}_{Iy}$ to denote the parameters of the auxiliary vector extraction part, with acoustic data $\boldsymbol{Y}^{(s)}$ as its input; and $\boldsymbol{\theta}_{Ix}$ to denote the parameters of the attention mechanism, with linguistic data $\boldsymbol{X}^{(s)}$ as its input, respectively.

The expression of partial derivative w.r.t. $\boldsymbol{\theta}_{Iy}$ is very similar to Eq.7, with the additional dependencies on the attention mechanism and its input:

$$\frac{\partial \mathcal{E}}{\partial \boldsymbol{\theta}_{Iy}} = \frac{1}{\sum_{\forall s} T^{(s)}} \sum_{\forall s} \sum_{t=1}^{T^{(s)}} \frac{\partial \mathcal{E}_t^{(s)}}{\partial \boldsymbol{\theta}_{Iy}} \Big|_{\boldsymbol{\theta}_A, \boldsymbol{x}_t, \boldsymbol{y}_t^{(s)}, \boldsymbol{Y}^{(s)}, \boldsymbol{X}^{(s)}, \boldsymbol{\theta}_{Ix}}$$

$$= \frac{1}{\sum_{\forall s} T^{(s)}} \sum_{\forall s} \left[\left(\frac{\partial \boldsymbol{\lambda}^{(s)}}{\partial \boldsymbol{\theta}_{Iy}} \Big|_{\boldsymbol{Y}^{(s)}, \boldsymbol{X}^{(s)}, \boldsymbol{\theta}_{Ix}} \right) \sum_{t=1}^{T^{(s)}} \left(\frac{\partial \mathcal{E}_t^{(s)}}{\partial \boldsymbol{\lambda}^{(s)}} \Big|_{\boldsymbol{\theta}_A, \boldsymbol{x}_t, \boldsymbol{y}_t^{(s)}} \right) \right] \tag{11}$$

For each mini-batch, the dependencies of this partial derivative include $\{\boldsymbol{Y}^{(s)}, \boldsymbol{X}^{(s)}, \boldsymbol{\theta}_{Ix}, \boldsymbol{\theta}_A, \boldsymbol{x}_t, \boldsymbol{y}_t^{(s)}\}$. The gradient expression of $\boldsymbol{\theta}_{Ix}$ is also very similar:

$$\frac{\partial \mathcal{E}}{\partial \boldsymbol{\theta}_{Ix}} = \frac{1}{\sum_{\forall s} T^{(s)}} \sum_{\forall s} \sum_{t=1}^{T^{(s)}} \frac{\partial \mathcal{E}_t^{(s)}}{\partial \boldsymbol{\theta}_{Ix}}\Big|_{\boldsymbol{\theta}_A, \boldsymbol{x}_t, \boldsymbol{y}_t^{(s)}, \boldsymbol{X}^{(s)}}$$
$$= \frac{1}{\sum_{\forall s} T^{(s)}} \sum_{\forall s} \left[\left(\frac{\partial \boldsymbol{\lambda}^{(s)}}{\partial \boldsymbol{\theta}_{Ix}}\Big|_{\boldsymbol{X}^{(s)}, \boldsymbol{Y}^{(s)}, \boldsymbol{\theta}_{Iy}} \right) \sum_{t=1}^{T^{(s)}} \left(\frac{\partial \mathcal{E}_t^{(s)}}{\partial \boldsymbol{\lambda}^{(s)}}\Big|_{\boldsymbol{\theta}_A, \boldsymbol{x}_t, \boldsymbol{y}_t^{(s)}} \right) \right] \quad (12)$$

For each mini-batch, the dependencies of this partial derivative include $\{\boldsymbol{Y}^{(s)}, \boldsymbol{X}^{(s)}, \boldsymbol{\theta}_{Iy}, \boldsymbol{\theta}_A, \boldsymbol{x}_t, \boldsymbol{y}_t^{(s)}\}$.

For our integrated framework, since the iVector takes the form of an average (Eq.8), the attention model can provide the weights for the averaging (Fig.4):

$$\boldsymbol{\lambda}^{(s)} = \sum_{\tau=1}^{T^{(s)}} \alpha_\tau^{(s)} \mathcal{G}_{Iy}\left(\boldsymbol{y}_\tau^{(s)}, \boldsymbol{\theta}_{Iy}\right) \quad (13)$$
$$\text{subject to } \alpha_\tau^{(s)} \in [0,1]; \sum_{\tau=1}^{T^{(s)}} \alpha_\tau^{(s)} = 1$$

To satisfy these constraints, a normalising layer is applied on top of the last layer of the attention extraction model:

$$\alpha_\tau^{(s)} = \frac{\mathcal{G}_{Ix}\left(\boldsymbol{x}_\tau^{(s)}, \boldsymbol{\theta}_{Ix}\right)}{\sum_{\tau'=1}^{T^{(s)}} \mathcal{G}_{Ix}\left(\boldsymbol{x}_{\tau'}^{(s)}, \boldsymbol{\theta}_{Ix}\right)} \quad (14)$$

In our work, the last layer of $\mathcal{G}_{Ix}(\cdot)$ is a sigmoid layer.

3. Training of the Models

This section discusses how to train the various model components in each framework, the dependencies in the training of each and the pre-trainings involved.

3.1. Independent iVector Extraction

The training of the independent iVector extraction auxiliary model depends only on $\{\boldsymbol{Y}^{(s)}\}$. After independent iVector extraction, since the only parameters to optimise are acoustic model parameters $\boldsymbol{\theta}_A$, training is quite straightforward with stochastic gradient descent (SGD) (Eq.3). The gradient dependencies include $\{\boldsymbol{\lambda}^{(s)}, \boldsymbol{x}_t, \boldsymbol{y}_t^{(s)}\}$.

3.2. Integrated iVector Extraction

The gradient dependencies of the acoustic model $\boldsymbol{\theta}_A$ include $\{\boldsymbol{\lambda}^{(s)}, \boldsymbol{x}_t, \boldsymbol{y}_t^{(s)}\}$, same as above. The gradient dependencies of the auxiliary model $\boldsymbol{\theta}_{Iy}$ include $\{\boldsymbol{Y}^{(s)}, \boldsymbol{\theta}_A, \boldsymbol{x}_t, \boldsymbol{y}_t^{(s)}\}$, and it is practically challenging and numerically unnecessary to load all acoustic data to form a gigantic $\boldsymbol{Y}^{(s)}$ for the estimation of $\boldsymbol{\lambda}^{(s)}$. Therefore, for each mini-batch of $\{\boldsymbol{x}_t, \boldsymbol{y}_t^{(s)}\}$, P utterances are randomly drawn for a noisy estimation of $\boldsymbol{\lambda}^{(s)}$ and gradients, and the process is listed in Algorithm 1.

For the initialisation of the model, we found that pre-training both model components, the acoustic model $\boldsymbol{\theta}_A$ and the auxiliary model $\boldsymbol{\theta}_{Iy}$, can largely improve the performance and convergence speed. The pre-training of $\boldsymbol{\theta}_A$ is simply the incompletely trained baseline model (e.g. the best validation score in the warm-up phase of training of baseline), and the pre-training of $\boldsymbol{\theta}_{Iy}$ has a cost function of the mean-square-error between the predicted iVectors and the independently extracted iVectors for the baseline model:

$$\mathcal{E}_{pre} = \frac{1}{\sum_{\forall s} T^{(s)}} \sum_{\forall s} \sum_{t=1}^{T^{(s)}} ||\boldsymbol{\lambda}_t^{(s)} - \boldsymbol{\lambda}'^{(s)}||^2 \quad (15)$$

$\boldsymbol{\lambda}_t^{(s)}$ is one instance of noisy estimate of iVector s using P utterances, and $\boldsymbol{\lambda}'^{(s)}$ is the independently extracted iVector for the baseline model.

Algorithm 1: Integrated training procedure

Initialization: random initialisation or load pre-trained model components;
for *each epoch* **do**
  for *each mini-batch (utterance) in training data,* **do**
    load linguistic and acoustic data $\left(\boldsymbol{x}_t^{(s)}, \boldsymbol{y}_t^{(s)}\right)$;
    identify the speaker s;
    draw P utterances of the same speaker;
      (exclude current and held-out utterances);
    if *attention mechanism applied* **then**
      concatenate P utterances $\Rightarrow \left(\boldsymbol{X}^{(s)}, \boldsymbol{Y}^{(s)}\right)$;
      update all model parameters, Eq.3, 12, 13;
    end
    else
      concatenate P utterances $\Rightarrow \left(\boldsymbol{Y}^{(s)}\right)$;
      update all model parameters, Eq.3, 7;
    end
  end
end

3.3. Integrated iVector Extraction with Attention Mechanism

The gradient dependencies of the acoustic model $\boldsymbol{\theta}_A$ include $\{\boldsymbol{\lambda}^{(s)}, \boldsymbol{x}_t, \boldsymbol{y}_t^{(s)}\}$, same as above. The gradient dependencies of the auxiliary model parameters $\{\boldsymbol{\theta}_{Iy}, \boldsymbol{\theta}_{Ix}\}$ include $\{\boldsymbol{Y}^{(s)}, \boldsymbol{X}^{(s)}, \boldsymbol{\theta}_A, \boldsymbol{x}_t, \boldsymbol{y}_t^{(s)}\}$, and the training process is same as before, Algorithm 1.

For the initialisation of the model, the pre-trained acoustic model $\boldsymbol{\theta}_A$ and the auxiliary vector extraction part $\boldsymbol{\theta}_{Iy}$ are the incompletely trained components (best validation score in the warm-up phase) from the training of the integrated framework above. The pre-training of $\boldsymbol{\theta}_{Ix}$ uses the same cost function of Eq.15, with bootstrapped $\boldsymbol{\theta}_{Iy}$.

3.4. Speech Generation for an Unseen Speaker

For the integrated frameworks, to generate speech for an unseen speaker, first, all (or a subset of) available utterances of speaker s are concatenated for the estimation of $\boldsymbol{\lambda}^{(s)}$ using the auxiliary model. Next, $\boldsymbol{\lambda}^{(s)}$ and the linguistic features of the target sentence are used for the generation of acoustic features (Eq.1).

For training process, training utterances (not held-out) are given to both the acoustic model and the auxiliary model. For validation process, previously unseen (held-out) utterances are given to the acoustic model, and previously seen (training) utterances are used for iVector estimation.

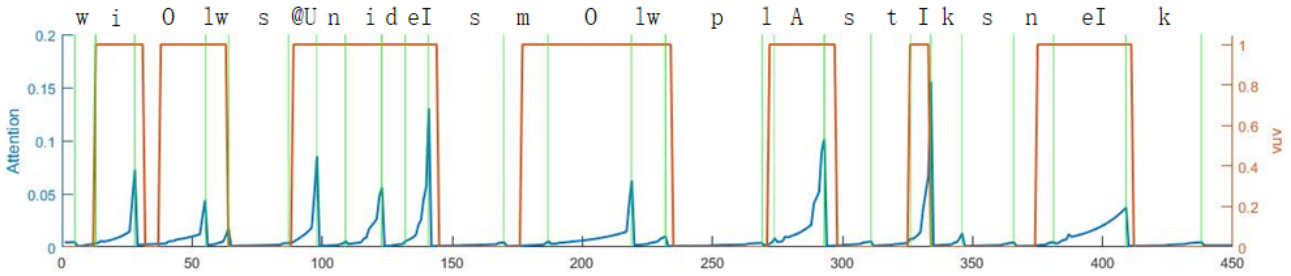

Figure 5: *phone sequence (black words and green boundaries), attention trajectory (blue) and vuv trajectory (red) for held-out utterance: "We also need a small plastic snake"*

4. Experiments

4.1. Configurations

In the experiments, 189 speakers from the Voice Bank corpus [13] are chosen for training, with on average $\sim$ 400 utterances for each speaker, excluding 40 common held-out utterances. 10 speakers each are chosen for validation and testing, and the 40 previously unseen utterances from these 20 speakers are used for validation and testing, respectively. Note that the estimations of the iVectors of the validation and testing speakers do not use these 40 unseen utterances, but the rest.

We denote the model with independently extracted iVector as "baseline", the model with integrated iVector extraction as "IIE", and the model with integrated iVector extraction and attention mechanism as "IIEA". The GMM-based baseline i-vectors are extracted using ALIZE toolkit [14].

The acoustic features for iVector extraction consists of 60D mel-cepstral coefficients (MCC), 25D band-aperiodicities (BAP) and linear-interpolated log F0, and their delta and delta-delta dynamic terms (total 258D). Voiced-unvoiced binaries (VUV) are not included for iVector extraction, but is included as the output of the acoustic model (total 259D). The acoustic features are extracted with STRAIGHT vocoder [15].

The linguistic input for both attention mechanism and the acoustic model consists of 592 binary linguistic features and 9 numerical features (total 601D). The iVectors are appended to every frame of the linguistic input.

We use Merlin [16], Theano [17] and Lasagne [18] to build the deep neural networks for the integrated adaptive speech generation frameworks. The speech generation acoustic model (bottom-right of Fig.4) $\boldsymbol{\theta}_A$ is a feed-forward DNN with a linear output layer of size 259 on top of 6 $\tanh$ layers of size 1536 each. The auxiliary model in integrated frameworks $\boldsymbol{\theta}_{Iy}$ is a DNN with a linear output layer on top of a $\tanh$ layer of the same size, either 32 or 128. The attention mechanism $\boldsymbol{\theta}_{Ix}$ is a DNN with a normalising output layer on top of a $\tanh$ layer of size 16 and a sigmoid layer of size 1. P is set to 20 for optimal speed and performance. The optimiser is ADAM [19] with decaying learning rate.

4.2. Interpret Attention

In our experiments, the attention tends to peak within the voiced regions (see Fig.5, where VUV=1). The attention also tends to peak within the vowels, which fits our hypothesis that these regions are more representative of speaker characteristics. On the other hand, the attention mechanism can be more selective than a simple VUV binary decision or a simple vowel/consonant binary decision, and it is optimised automatically with the same cost function on the final acoustic output. Therefore, we should expect IIEA to generate a more representative set of iVectors specifically for adaptive speech synthesis.

4.3. Objective Evaluations

Next we analyse in more details for iVector size 32 and 128.

Model	MCD (dB)	BAP (dB)	F0 RMSE	F0 CORR	VUV error
baseline	7.002	2.285	32.027	0.813	**6.007%**
IIE	**6.836**	2.256	30.817	0.821	6.028%
IIEA	6.852	**2.249**	**30.150**	**0.829**	6.016%

Table 1: *Objective Validation measures, iVector size 32*

In Table 1, IIE outperforms baseline in every aspect except VUV error rate; IIEA outperforms IIE in every aspect except Mel-cepstral Distortion (MCD). These suggest that both the integrated framework and the attention mechanism improve the iVector representation for better performance in adaptive speech synthesis.

Model	MCD (dB)	BAP (dB)	F0 RMSE	F0 CORR	VUV error
baseline	6.703	2.229	30.463	**0.836**	5.924%
IIE	**6.601**	**2.220**	30.017	0.831	**5.770%**
IIEA	6.615	2.222	**29.889**	0.833	5.799%

Table 2: *Objective Testing measures, iVector size 128*

In Table 2, IIE outperforms baseline in every aspect except F0 correlation; however, IIEA only outperforms IIE in F0 RMSE and F0 Correlation. The degradation is largely because the attention mechanism is difficult to train, and it is possible but maybe laborious to improve the performance of IIEA through more extensive hyper-parameter tuning. Nonetheless, as mentioned before, attention mechanism provides sensible and interpretable results (Fig.5), which are promising to progress further.

4.4. Subjective Evaluations

For subjective evaluations, we focus on two aspects, the naturalness of speech and the similarity to the original speaker. Each listener taking the test assessed the 3 pairs of model combinations for 8 random utterances among $\sim$400 held-out test utterances. For the similarity tests, the original utterance was given as the reference. Workers from Amazon Mechanical Turk were asked to take the test for a small reward [20][21].

For CMOS results, paired t-test is used to examine the significance between models, with solid brackets mean p-value$<$ 0.001, dashed brackets means p-value$<$ 0.01, and dotted brackets means p-value$<$ 0.05.

First we look at iVector size 128, since their objective results are generally better. In naturalness test (Fig.6), in terms of both CMOS and percentage preference, IIEA outperforms IIE, which outperforms baseline, as we have expected. Since the configuration of the acoustic model remains the same, this shows that both integrated training and attention mechanism improves the iVector representations of the speakers, for better performance in speech synthesis tasks, as compared to generic iVector representation.

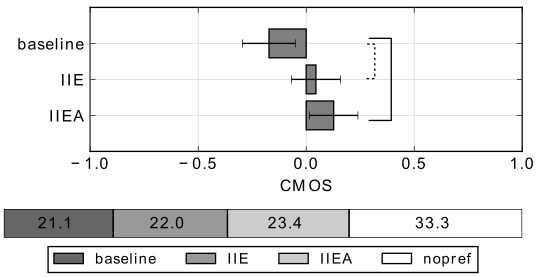

Figure 6: *naturalness preference test results, iVector size 128; 32 listeners took the test*

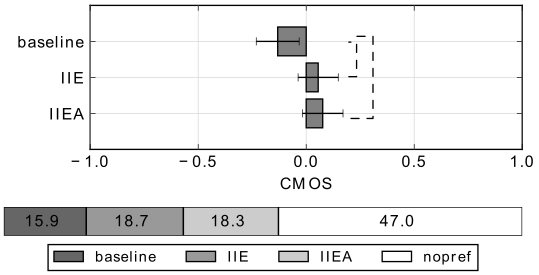

Figure 7: *similarity preference test results, iVector size 128; 33 listeners took the test*

In similarity test (Fig.7), in terms of CMOS, again IIEA outperforms IIE, which outperforms baseline. In terms of percentage preference, IIE outperforms IIEA, but both have much smaller ($<$19%) and very similar values ($\pm$0.4%), and percentage of no-preference is very high (47%), thus there is no sufficient evidence that IIEA has significant worse performance than IIE in similarity tests, while both outperform the baseline with relatively large improvements.

Next we look at iVector size 32, and interestingly, the trend of size 128 reversed, that this time IIE outperforms IIEA in naturalness (Fig.8), and IIEA outperforms IIE in similarity (Fig.9). Still, both IIE and IIEA outperforms the baseline in CMOS and percentage preference in all cases, and the difference between IIE and IIEA is not very significant, much less than the difference between either of them and the baseline.

Further analysis is required to fully understand the performance of the models in different configurations and tests, for an informed decision of which to choose given different tasks and priority objectives.

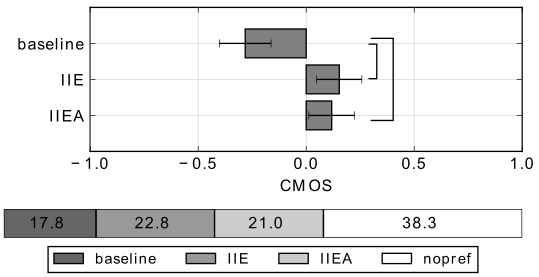

Figure 8: *naturalness preference test results, iVector size 32; 34 listeners took the test*

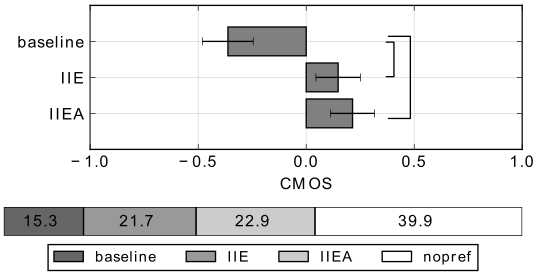

Figure 9: *similarity preference test results, iVector size 32; 31 listeners took the test*

5. Conclusions

In this work, we introduced integrated training framework and attention mechanism to improve the representativeness of iVector extraction for adaptive speech synthesis specifically, and we proposed some simple but effective auxiliary models for each components of the framework. They show improved performance, in both objective and subjective evaluations, compared to the standard approach GMM-based i-vector system, and the attention mechanism yields sensible and promising results.

In addition, our proposed integrated iVector extraction with attention mechanism can be applied to other forms of deep acoustic models for speaker adaptation as well. While in this work, the acoustic model is a simple feed-forward DNN, with contextual labels as inputs and vocoder parameters (and deltas) as outputs, it is very straightforward to apply our framework to an RNN acoustic model, to WaveNet [5] for waveform level synthesis by replacing the current one-hot vector, or to Char2Wav [6] for a complete end-to-end adaptive speech synthesis system.

In the future, we would like to investigate more complicated models, such as an RNN-based attention mechanism, or a WaveNet-like acoustic model. We would also like to investigate the effect of the amount of data for rapid adaptation, and the effect of iVector size for potential small-size iVector-based controllable synthesis.

6. Acknowledgements

The research for this paper was jointly supported by EPSRC International Doctoral Scholarship, reference number 10348827; St. John's College Internal Graduate Scholarship; the European Unions Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No 655764; and EPSRC grant EP/I031022/1 (Natural Speech Technology).

7. References

[1] K. Tokuda, H. Zen, and A. W. Black, "An HMM-based speech synthesis system applied to english," pp. 227–230, 2002.

[2] S. J. Prince and J. H. Elder, "Probabilistic linear discriminant analysis for inferences about identity," pp. 1–8, 2007.

[3] D. Garcia-Romero and C. Y. Espy-Wilson, "Analysis of i-vector Length Normalization in Speaker Recognition Systems." vol. 2011, pp. 249–252, 2011.

[4] Z. Wu, P. Swietojanski, C. Veaux, S. Renals, and S. King, "A study of speaker adaptation for DNN-based speech synthesis." pp. 879–883, 2015.

[5] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "WaveNet: A generative model for raw audio," *CoRR abs/1609.03499*, 2016.

[6] J. Sotelo, S. Mehri, K. Kumar, J. F. Santos, K. Kastner, A. Courville, and Y. Bengio, "Char2Wav: End-to-end speech synthesis," 2017.

[7] H.-T. Luong, S. Takaki, G. E. Henter, and J. Yamagishi, "Adapting and controlling DNN-based speech synthesis using input codes," in *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*. IEEE, 2017, pp. 4905–4909.

[8] N. Hojo, Y. Ijima, and H. Mizuno, "An Investigation of DNN-Based Speech Synthesis Using Speaker Codes." in *INTERSPEECH*, 2016, pp. 2278–2282.

[9] E. Variani, X. Lei, E. McDermott, I. L. Moreno, and J. Gonzalez-Dominguez, "Deep neural networks for small footprint text-dependent speaker verification," pp. 4052–4056, 2014.

[10] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788–798, 2011.

[11] P. Karanasou, Y. Wang, M. J. Gales, and P. C. Woodland, "Adaptation of deep neural network acoustic models using factorised i-vectors," 2014.

[12] C. Dromey and M. Sanders, "Intra-speaker variability in palatometric measures of consonant articulation," *Journal of communication disorders*, vol. 42, no. 6, pp. 397–407, 2009.

[13] C. Veaux, J. Yamagishi, and S. King, "The voice bank corpus: Design, collection and data analysis of a large regional accent speech database," pp. 1–4, 2013.

[14] A. Larcher, J.-F. Bonastre, B. G. Fauve, K.-A. Lee, C. Lévy, H. Li, J. S. Mason, and J.-Y. Parfait, "Alize 3.0-open source toolkit for state-of-the-art speaker recognition." pp. 2768–2772, 2013.

[15] H. Kawahara, I. Masuda-Katsuse, and A. De Cheveigne, "Restructuring speech representations using a pitch-adaptive time–frequency smoothing and an instantaneous-frequency-based F0 extraction: Possible role of a repetitive structure in sounds," *Speech communication*, vol. 27, no. 3, pp. 187–207, 1999.

[16] Z. Wu, O. Watts, and S. King, "Merlin: An open source neural network speech synthesis system," *Proc. SSW, Sunnyvale, USA*, 2016.

[17] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio, "Theano: A CPU and GPU math compiler in python," pp. 1–7, 2010.

[18] S. Dieleman, J. Schlter, C. Raffel, E. Olson, S. K. Snderby, D. Nouri *et al.*, "Lasagne: First release." Aug. 2015. [Online]. Available: http://dx.doi.org/10.5281/zenodo.27878

[19] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *CoRR*, vol. abs/1412.6980, 2014. [Online]. Available: http://arxiv.org/abs/1412.6980

[20] M. K. Wolters, K. B. Isaac, and S. Renals, "Evaluating speech synthesis intelligibility using Amazon Mechanical Turk," 2010.

[21] S. Buchholz and J. Latorre, "Crowdsourcing preference tests, and how to detect cheating," 2011.