Waveform-Based Speaker Representations for 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

Speaker adaptation is a key aspect of building a range of speech processing systems, for example personalised speech synthesis. For deep-learning based approaches, the model parameters are hard to interpret, making speaker adaptation more challenging. One widely used method to address this problem is to extract a fixed length vector as speaker representation, and use this as an additional input to the task-specific model. This allows speaker-specific output to be generated, without modifying the model parameters. However, the speaker representation is often extracted in a task-independent fashion. This allows the same approach to be used for a range of tasks, but the extracted representation is unlikely to be optimal for the specific task of interest. Furthermore, the features from which the speaker representation is extracted are usually pre-defined, often a standard speech representation. This may limit the available information that can be used. In this paper, an integrated optimisation framework for building a task specific speaker representation, making use of all the available information, is proposed. Speech synthesis is used as the example task. The speaker representation is derived from raw waveform, incorporating text information via an attention mechanism. This paper evaluates and compares this framework with standard task-independent forms.

Index Terms: integrated, adaptation, speech synthesis, fixed-length speaker representation, attention mechanism, waveform, vocoder

1. Introduction

Speaker adaptation addresses the problem of adapting the behaviour of a model to generate speaker-specific outputs, for example, adaptive speech synthesis can generate speech of an arbitrary speaker's voice [1]. For deep-learning based models, due to the large number of parameters and connections which are not interpretable, adaptation is particularly challenging. One standard approach to address this issue is to extract a fixed length vector from a speaker's "enrolment" data as that speaker's representation, and this vector becomes additional input to the task-specific model. Together with the original input, this task-specific model could generate speaker-specific output, without modifying the model parameters [2][3]. This form of adaptation could be easily applied to a range of tasks, such as speech synthesis and speech recognition, and a range of task-specific models, such as feed-forward deep neural network (DNN), recurrent neural network (RNN), WaveNet [4] or Char2Wav [5].

It is necessary to optimise both the speaker representation extraction model and the task-specific model, and in this paper, two options of optimising the speaker representation extraction model parameters are compared. The standard approach involve two distinct stages. First, the speaker representation extraction model is optimised independently of the task of interest, using a very different optimisation criterion. For example, for Gaussian Mixture Model (GMM) based *i-vector* [6][7], the optimisation criterion is maximum a posteriori (MAP) of speaker-dependent GMMs; another example is DNN-based *d-vector* [8], which has the optimisation criterion of minimum speaker classification error. Next, after the speaker representation extraction model is optimised, speaker representation vectors are extracted, and the task-specific model is optimised given these extracted speaker representation vectors. The second approach is integrated optimisation which was investigated in our previous work [9], that the speaker representation extraction model shares the same optimisation criterion as the task-specific model. If gradient-descent based optimisation schemes are used, the gradient with respect to the speaker representation extraction model parameters can be propagated through speaker representation vectors.

The obvious advantage of the integrated approach is that both models are optimised with the same criterion, and thus the overall system would have better performance on this criterion. On the other hand, the integrated approach requires easy gradient propagation between the two models through the speaker representation vector, and therefore in this work, the speaker representation extraction model has a similar form with that of the DNN-based d-vector method [8], and optimisation of either model could be achieved by back-propagation conveniently.

The second improvement to the standard approach is the introduction of a text-dependent attention mechanism from our previous work [9], which could select the most representative parts from the enrolment data for speaker representation extraction. This would improve the simple averaging of the standard d-vector approach [8].

The last improvement and the novelty of this work is to extend the integrated framework to extract speaker representations from waveform directly. The standard approaches extract speaker representations from some pre-defined features, such as vocoder parameters. The standard vocoder, which extracts acoustic features from waveform, has been optimised for a very different criterion e.g. minimal reconstruction distortion or best perceptual quality. In this paper, a "neural vocoder" replaces the standard acoustic feature extraction step. It extracts speaker representations from all information in the raw waveform, and it is optimised with the same criterion as the task-specific model.

2. Speaker Adaptation

This paper considers the situation where the training data comprises data from multiple, S, speakers, and the test data is associated with an unknown test speaker. Thus the training data, $\mathcal{D}$, can be written as

$$\mathcal{D} = \{\mathcal{D}^{(1)}, \ldots, \mathcal{D}^{(\mathtt{S})}\} \tag{1}$$

where $\mathcal{D}^{(\mathtt{s})} = \{\boldsymbol{y}_{1:\mathrm{T}}^{(\mathtt{s})}, \boldsymbol{x}_{1:\mathrm{L}}^{(\mathtt{s})}\}$ is the data, T-length output sequence and L-length input sequence, associated with the s^{th} speaker. For the test speaker there is "enrolment" data, $\mathcal{D}^{\star}$, and input sequence $\boldsymbol{x}_{1:\mathrm{L}}^{\star}$. The goal is to predict the speaker-specific

output sequence, $p(\boldsymbol{y}^{\star}_{1:\mathrm{T}}|\boldsymbol{x}^{\star}_{1:\mathrm{L}}, \mathcal{D}^{\star}; \boldsymbol{\theta})$; $\boldsymbol{\theta}$ is the set of speaker-independent model parameters.

2.1. Fixed Length Vector Speaker Representation

All the work in this paper assume that the attributes of a speaker can be represented by some fixed length vector denoted as $\boldsymbol{\lambda}$. The general framework for generating the task-specific output sequence $\boldsymbol{y}^{\star}_{1:\mathrm{T}}$, can then be expressed as

$$p(\boldsymbol{y}^{\star}_{1:\mathrm{T}}|\boldsymbol{x}^{\star}_{1:\mathrm{L}}, \mathcal{D}^{\star}; \boldsymbol{\theta}_{\mathtt{a}}, \boldsymbol{\theta}_{\lambda}) = \int p(\boldsymbol{y}^{\star}_{1:\mathrm{T}}|\boldsymbol{x}^{\star}_{1:\mathrm{L}}, \boldsymbol{\lambda}; \boldsymbol{\theta}_{\mathtt{a}}) p(\boldsymbol{\lambda}|\mathcal{D}^{\star}; \boldsymbol{\theta}_{\lambda}) d\boldsymbol{\lambda} \tag{2}$$

where $\boldsymbol{\theta}_{\lambda}$ are the parameters associated with extracting the speaker representation $\boldsymbol{\lambda}$, and $\boldsymbol{\theta}_{\mathtt{a}}$ is the task-specific model parameters. $\boldsymbol{\theta} = \{\boldsymbol{\theta}_{\lambda}, \boldsymbol{\theta}_{\mathtt{a}}\}$.

For simplicity the work in this paper will assume that there is a unique estimate of the speaker representation for one speaker, given $\mathcal{D}^{\star}$ and $\boldsymbol{\theta}_{\lambda}$,

$$p(\boldsymbol{\lambda}|\mathcal{D}^{\star}; \boldsymbol{\theta}_{\lambda}) = \delta(\boldsymbol{\lambda} - \boldsymbol{\lambda}^{\star}); \quad \boldsymbol{\lambda}^{\star} = \boldsymbol{f}(\mathcal{D}^{\star}; \boldsymbol{\theta}_{\lambda}) \tag{3}$$

thus Eq.2 can be written as

$$p(\boldsymbol{y}^{\star}_{1:\mathrm{T}}|\boldsymbol{x}^{\star}_{1:\mathrm{L}}, \mathcal{D}^{\star}; \boldsymbol{\theta}_{\mathtt{a}}, \boldsymbol{\theta}_{\lambda}) = p(\boldsymbol{y}^{\star}_{1:\mathrm{T}}|\boldsymbol{x}^{\star}_{1:\mathrm{L}}, \boldsymbol{\lambda}^{\star}; \boldsymbol{\theta}_{\mathtt{a}}) \tag{4}$$

It is necessary to train the model parameters $\boldsymbol{\theta}_{\mathtt{a}}$ and $\boldsymbol{\theta}_{\lambda}$, and the training criterion[1] can be expressed as

$$\mathcal{L}(\boldsymbol{\theta}_{\mathtt{a}}, \boldsymbol{\theta}_{\lambda}) = \sum_{s=1}^{S} \log\left(p(\boldsymbol{y}^{(\mathtt{s})}_{1:\mathrm{T}}|\boldsymbol{x}^{(\mathtt{s})}_{1:\mathrm{L}}, \boldsymbol{\lambda}^{(\mathtt{s})}; \boldsymbol{\theta}_{\mathtt{a}})\right) \tag{5}$$

$$= \sum_{s=1}^{S} \log\left(p(\boldsymbol{y}^{(\mathtt{s})}_{1:\mathrm{T}}|\boldsymbol{x}^{(\mathtt{s})}_{1:\mathrm{L}}, \boldsymbol{f}(\mathcal{D}^{(\mathtt{s})}; \boldsymbol{\theta}_{\lambda}); \boldsymbol{\theta}_{\mathtt{a}})\right) \tag{6}$$

The optimisation of the task-specific model parameters, $\boldsymbol{\theta}_{\mathtt{a}}$, given the parameters of the speaker representation $\hat{\boldsymbol{\theta}}_{\lambda}$, and the set of estimated speaker representations $\{\hat{\boldsymbol{\lambda}}^{(1)}, \ldots, \hat{\boldsymbol{\lambda}}^{(\mathtt{s})}\}$ where $\hat{\boldsymbol{\lambda}}^{(\mathtt{s})} = \boldsymbol{f}(\mathcal{D}^{(\mathtt{s})}; \hat{\boldsymbol{\theta}}_{\lambda})$, can be written as

$$\hat{\boldsymbol{\theta}}_{\mathtt{a}}|\hat{\boldsymbol{\theta}}_{\lambda} = \arg\max_{\boldsymbol{\theta}_{\mathtt{a}}} \left\{ \sum_{s=1}^{S} \log\left(p(\boldsymbol{y}^{(\mathtt{s})}_{1:\mathrm{T}}|\boldsymbol{x}^{(\mathtt{s})}_{1:\mathrm{L}}, \hat{\boldsymbol{\lambda}}^{(\mathtt{s})}; \boldsymbol{\theta}_{\mathtt{a}})\right) \right\}$$

$$= \arg\max_{\boldsymbol{\theta}_{\mathtt{a}}} \left\{ \sum_{s=1}^{S} \mathcal{L}_{\mathtt{a}}(\boldsymbol{\theta}_{\mathtt{a}}, \hat{\boldsymbol{\lambda}}^{(\mathtt{s})}) \right\} \tag{7}$$

This form of optimisation has been used to find task-specific parameters. If gradient-descent based optimisation schemes are used then the gradient is required. This can be written as

$$\left.\frac{\partial \mathcal{L}(\boldsymbol{\theta}_{\mathtt{a}}, \boldsymbol{\theta}_{\lambda})}{\partial \boldsymbol{\theta}_{\mathtt{a}}}\right|_{\hat{\boldsymbol{\theta}}_{\lambda}} = \sum_{s=1}^{S} \left.\frac{\partial \mathcal{L}_{\mathtt{a}}(\boldsymbol{\theta}_{\mathtt{a}}, \boldsymbol{\lambda}^{(\mathtt{s})})}{\partial \boldsymbol{\theta}_{\mathtt{a}}}\right|_{\hat{\boldsymbol{\lambda}}^{(\mathtt{s})}} \tag{8}$$

2.2. Two-Stage or Integrated Optimisation

The main interest in this work is the form of the optimisation of the speaker representation extraction model parameters, $\boldsymbol{\theta}_{\lambda}$. There are two options that can be examined:

[1]Here the task-specific criterion is assumed to be the conditional maximum likelihood criterion. Other criteria can also be considered.

1. The standard approach to estimate the parameters of the speaker representation is to define a speaker representation criterion $\mathcal{L}_{\lambda}(\boldsymbol{\theta}_{\lambda})$, and then an estimate of the speaker representation $\hat{\boldsymbol{\theta}}_{\lambda}$ is found

$$\hat{\boldsymbol{\theta}}_{\lambda} = \arg\max_{\boldsymbol{\theta}_{\lambda}} \{\mathcal{L}_{\lambda}(\boldsymbol{\theta}_{\lambda})\} \tag{9}$$

One such example is GMM-based i-vector, where the training criterion is maximum a posteriori (MAP) of speaker-dependent GMMs [6][7]; Another example is DNN-based d-vector, where the training criterion is to minimise speaker classification error [8][10].

2. The approach adopted in this paper is to make the cost function associated with the speaker representation linked with the task-specific criterion. In this case

$$\hat{\boldsymbol{\theta}}_{\lambda}|\hat{\boldsymbol{\theta}}_{\mathtt{a}} = \arg\max_{\boldsymbol{\theta}_{\lambda}} \left\{ \mathcal{L}(\hat{\boldsymbol{\theta}}_{\mathtt{a}}, \boldsymbol{\theta}_{\lambda}) \right\} \tag{10}$$

Consider a gradient-descent based optimisation scheme, this requires computing

$$\left.\frac{\partial \mathcal{L}(\boldsymbol{\theta}_{\mathtt{a}}, \boldsymbol{\theta}_{\lambda})}{\partial \boldsymbol{\theta}_{\lambda}}\right|_{\hat{\boldsymbol{\theta}}_{\mathtt{a}}} = \sum_{s=1}^{S} \left(\frac{\partial \boldsymbol{\lambda}^{(\mathtt{s})}}{\partial \boldsymbol{\theta}_{\lambda}} \left.\frac{\partial \mathcal{L}_{\mathtt{a}}(\boldsymbol{\theta}_{\mathtt{a}}, \boldsymbol{\lambda}^{(\mathtt{s})})}{\partial \boldsymbol{\lambda}^{(\mathtt{s})}}\right|_{\hat{\boldsymbol{\theta}}_{\mathtt{a}}} \right) \tag{11}$$

The first approach described is a two-stage scheme, that first an estimate of the speaker representation $\hat{\boldsymbol{\theta}}_{\lambda}$ is found, then the task-specific model parameters $\hat{\boldsymbol{\theta}}_{\mathtt{a}}$ found. This approach suffers from a major mismatch between the training criteria, $\mathcal{L}_{\lambda}(\boldsymbol{\theta}_{\lambda})$ and $\mathcal{L}(\boldsymbol{\theta}_{\mathtt{a}}, \boldsymbol{\theta}_{\lambda})$, and therefore the set of estimated speaker representations, $\{\hat{\boldsymbol{\lambda}}^{(1)}, \ldots, \hat{\boldsymbol{\lambda}}^{(\mathtt{s})}\}$, are unlikely the optimal for the task of interest $\mathcal{L}(\boldsymbol{\theta}_{\mathtt{a}}, \boldsymbol{\theta}_{\lambda})$. On the contrary, the integrated approach would provide a set of estimated speaker representations specifically for the task of interest, and the performance of $\mathcal{L}(\boldsymbol{\theta}_{\mathtt{a}}, \boldsymbol{\theta}_{\lambda})$ would improve.

Nonetheless, the optimisation of $\boldsymbol{\theta}_{\lambda}$ in a two-stage scheme is usually faster and has smaller footprint, as it does not involve $\boldsymbol{\theta}_{\mathtt{a}}$ which is generally much larger. Therefore the two-stage scheme could provide an efficient and sensible initialisation of $\boldsymbol{\theta}_{\lambda}$ for the integrated approach.

2.3. Text-Dependent Speaker Representations

Many standard approaches extract speaker representations from acoustic features only, as it is believed that more speaker characteristic is encapsulated in the acoustic features than in the linguistic features. In the case of speech synthesis, the acoustic features are the output sequences $\boldsymbol{y}^{\star}_{1:\mathrm{T}}$. Also, many standard speaker representations, such as GMM-based i-vector and DNN-based d-vector, take some forms of averaging, in order to map a variable length vector sequence $\boldsymbol{y}^{\star}_{1:\mathrm{T}}$ to a fixed length vector $\boldsymbol{\lambda}^{\star}$. Therefore, if our $\boldsymbol{\theta}_{\lambda}$ takes a similar model architecture, Eq.3 can be modified as

$$\boldsymbol{\lambda}^{\star} = \frac{1}{T}\sum_{t=1}^{T} \boldsymbol{\lambda}^{\star}_{\mathtt{t}} = \frac{1}{T}\sum_{t=1}^{T} \boldsymbol{f}_{\mathtt{t}}(\boldsymbol{y}^{\star}_{\mathtt{t}}; \boldsymbol{\theta}_{\lambda}) \tag{12}$$

However, we believe that the rich linguistic features with broad contextual information could "guide" the speaker representation extraction process. More specifically, in the current form of averaging (Eq.12), we adopt the attention mechanism from our previous work [9],

$$\boldsymbol{\lambda}^{\star} = \sum_{t=1}^{T} \alpha^{\star}_{\mathtt{t}} \boldsymbol{\lambda}^{\star}_{\mathtt{t}} = \sum_{t=1}^{T} f_{\alpha}\left(\boldsymbol{x}^{\star}_{\mathtt{t}}, \boldsymbol{x}^{\star}_{1:\mathrm{T}}; \boldsymbol{\theta}_{\alpha}\right) \boldsymbol{f}_{\mathtt{t}}\left(\boldsymbol{y}^{\star}_{\mathtt{t}}; \tilde{\boldsymbol{\theta}}_{\lambda}\right) \tag{13}$$

whereas now $\boldsymbol{\theta}_\lambda = \{\boldsymbol{\theta}_\alpha, \tilde{\boldsymbol{\theta}}_\lambda\}$. Note that the attention mechanism has a sum-to-one constraint, therefore the weight of each sample is conditioned on all other samples. Also the linguistic features are state-aligned, thus have the same length T as the acoustic features. The gradient w.r.t. $\tilde{\boldsymbol{\theta}}_\lambda$ is:

$$\begin{aligned}\frac{\partial \mathcal{L}(\boldsymbol{\theta}_{\mathtt{a}}, \boldsymbol{\theta}_\lambda)}{\partial \tilde{\boldsymbol{\theta}}_\lambda}\bigg|_{\hat{\boldsymbol{\theta}}_{\mathtt{a}}, \hat{\boldsymbol{\theta}}_\alpha} &= \sum_{s=1}^{S} \left(\frac{\partial \boldsymbol{\lambda}^{(\mathtt{s})}}{\partial \tilde{\boldsymbol{\theta}}_\lambda}\bigg|_{\hat{\boldsymbol{\theta}}_\alpha} \frac{\partial \mathcal{L}_{\mathtt{a}}(\boldsymbol{\theta}_{\mathtt{a}}, \boldsymbol{\lambda}^{(\mathtt{s})})}{\partial \boldsymbol{\lambda}^{(\mathtt{s})}}\bigg|_{\hat{\boldsymbol{\theta}}_{\mathtt{a}}} \right) \\ &= \sum_{s=1}^{S} \left(\sum_{t=1}^{T} \left(\alpha_{\mathtt{t}}^{(\mathtt{s})} \frac{\partial \boldsymbol{\lambda}_{\mathtt{t}}^{(\mathtt{s})}}{\partial \tilde{\boldsymbol{\theta}}_\lambda}\bigg|_{\hat{\boldsymbol{\theta}}_\alpha} \right) \frac{\partial \mathcal{L}_{\mathtt{a}}(\boldsymbol{\theta}_{\mathtt{a}}, \boldsymbol{\lambda}^{(\mathtt{s})})}{\partial \boldsymbol{\lambda}^{(\mathtt{s})}}\bigg|_{\hat{\boldsymbol{\theta}}_{\mathtt{a}}} \right)\end{aligned} \tag{14}$$

The gradient w.r.t. $\boldsymbol{\theta}_\lambda$ in Eq.12 has a very similar expression, by replacing $\alpha_{\mathtt{t}}^{(\mathtt{s})}$ with $1/T$, thus denoted as "flat-attention".

The gradient w.r.t. $\boldsymbol{\theta}_\alpha$ is slightly different, due to sum-to-one constraint; therefore we use $\boldsymbol{\alpha}^{(\mathtt{s})} = [\alpha_1^{(\mathtt{s})}, \alpha_2^{(\mathtt{s})}, ..., \alpha_{\mathtt{T}}^{(\mathtt{s})}]'$ to denote the attention vector:

$$\begin{aligned}\frac{\partial \mathcal{L}(\boldsymbol{\theta}_{\mathtt{a}}, \boldsymbol{\theta}_\lambda)}{\partial \boldsymbol{\theta}_\alpha}\bigg|_{\hat{\boldsymbol{\theta}}_{\mathtt{a}}, \hat{\tilde{\boldsymbol{\theta}}}_\lambda} &= \sum_{s=1}^{S} \left(\frac{\partial \boldsymbol{\lambda}^{(\mathtt{s})}}{\partial \boldsymbol{\theta}_\alpha}\bigg|_{\hat{\tilde{\boldsymbol{\theta}}}_\lambda} \frac{\partial \mathcal{L}_{\mathtt{a}}(\boldsymbol{\theta}_{\mathtt{a}}, \boldsymbol{\lambda}^{(\mathtt{s})})}{\partial \boldsymbol{\lambda}^{(\mathtt{s})}}\bigg|_{\hat{\boldsymbol{\theta}}_{\mathtt{a}}} \right) \\ &= \sum_{s=1}^{S} \left(\frac{\partial \boldsymbol{\alpha}^{(\mathtt{s})}}{\partial \boldsymbol{\theta}_\alpha} \frac{\partial \boldsymbol{\lambda}^{(\mathtt{s})}}{\partial \boldsymbol{\alpha}^{(\mathtt{s})}}\bigg|_{\hat{\tilde{\boldsymbol{\theta}}}_\lambda} \frac{\partial \mathcal{L}_{\mathtt{a}}(\boldsymbol{\theta}_{\mathtt{a}}, \boldsymbol{\lambda}^{(\mathtt{s})})}{\partial \boldsymbol{\lambda}^{(\mathtt{s})}}\bigg|_{\hat{\boldsymbol{\theta}}_{\mathtt{a}}} \right)\end{aligned} \tag{15}$$

2.4. Waveform-Level Speaker Representation

Finally we investigate the form of $\boldsymbol{y}_{\mathtt{t}}^\star$. For completeness we start from Eq.13, and the same would apply to Eq.12 as well, by replacing $\alpha_{\mathtt{t}}^\star$ with $1/T$.

In Eq.13, the standard form of $\boldsymbol{y}_{\mathtt{t}}^\star$ is some combination of pre-defined acoustic features. For example, for DNN-based d-vector [8], each $\boldsymbol{y}_{\mathtt{t}}^\star$ is the stacking of filterbank energy features of several consecutive frames. However, the vocoder, which extracts acoustic features from waveform, has been optimised for a very different criterion e.g. minimal reconstruction distortion or best perceptual quality. Following the same argument in Section 2.2, it is possible to optimise this stage in the integrated framework as well, by replacing the vocoder with a "neural vocoder" with parameters $\boldsymbol{\theta}_{\mathtt{v}}$. Denote the sequence of waveform samples used to extract $\boldsymbol{y}_{\mathtt{t}}^\star$ as $o_{1:\tau}^\star$, Eq.13 can be rewritten as

$$\boldsymbol{\lambda}^\star = \sum_{t=1}^{T} \alpha_{\mathtt{t}}^\star \boldsymbol{f}_{\mathtt{t}}(\boldsymbol{y}_{\mathtt{t}}^\star; \boldsymbol{\theta}_\lambda) = \sum_{t=1}^{T} \alpha_{\mathtt{t}} \boldsymbol{f}_{\mathtt{t}}(\boldsymbol{f}_{\mathtt{v}}(o_{1:\tau}^\star; \boldsymbol{\theta}_{\mathtt{v}}); \tilde{\boldsymbol{\theta}}_\lambda) \tag{16}$$

and now $\boldsymbol{\theta}_\lambda = \{\boldsymbol{\theta}_{\mathtt{v}}, \tilde{\boldsymbol{\theta}}_\lambda, \boldsymbol{\theta}_\alpha\}$. The gradient of $\mathcal{L}(\boldsymbol{\theta}_{\mathtt{a}}, \boldsymbol{\theta}_\lambda)$ w.r.t. $\boldsymbol{\theta}_{\mathtt{v}}$ can be easily derived using chain-rule:

$$\begin{aligned}\frac{\partial \mathcal{L}(\boldsymbol{\theta}_{\mathtt{a}}, \boldsymbol{\theta}_\lambda)}{\partial \boldsymbol{\theta}_{\mathtt{v}}}\bigg|_{\hat{\boldsymbol{\theta}}_{\mathtt{a}}, \hat{\tilde{\boldsymbol{\theta}}}_\lambda, \hat{\boldsymbol{\theta}}_\alpha} &= \sum_{s=1}^{S} \left(\frac{\partial \boldsymbol{\lambda}^{(\mathtt{s})}}{\partial \boldsymbol{\theta}_{\mathtt{v}}}\bigg|_{\hat{\tilde{\boldsymbol{\theta}}}_\lambda, \hat{\boldsymbol{\theta}}_\alpha} \frac{\partial \mathcal{L}_{\mathtt{a}}(\boldsymbol{\theta}_{\mathtt{a}}, \boldsymbol{\lambda}^{(\mathtt{s})})}{\partial \boldsymbol{\lambda}^{(\mathtt{s})}}\bigg|_{\hat{\boldsymbol{\theta}}_{\mathtt{a}}} \right) \\ &= \sum_{s=1}^{S} \left(\sum_{t=1}^{T} \left(\alpha_{\mathtt{t}}^{(\mathtt{s})} \frac{\partial \boldsymbol{y}_{\mathtt{t}}^{(\mathtt{s})}}{\partial \boldsymbol{\theta}_{\mathtt{v}}} \frac{\partial \boldsymbol{\lambda}_{\mathtt{t}}^{(\mathtt{s})}}{\partial \boldsymbol{y}_{\mathtt{t}}^{(\mathtt{s})}}\bigg|_{\hat{\tilde{\boldsymbol{\theta}}}_\lambda} \right) \frac{\partial \mathcal{L}_{\mathtt{a}}(\boldsymbol{\theta}_{\mathtt{a}}, \boldsymbol{\lambda}^{(\mathtt{s})})}{\partial \boldsymbol{\lambda}^{(\mathtt{s})}}\bigg|_{\hat{\boldsymbol{\theta}}_{\mathtt{a}}} \right)\end{aligned} \tag{17}$$

3. Experiments

3.1. Data Configurations

In the experiments, the dataset is the Voice Bank corpus [11], which contains 209 speakers and on average $\sim$400 utterances for each speaker. First, 40 common held-out utterances are chosen and excluded from training. Next, 10 speakers each are chosen for validation and testing, and the 40 previously mentioned held-out utterances from these 20 speakers are used for validation and testing, respectively.

The vocoder parameters consist of 60D mel-cepstral coefficients (MCC), 25D mel Phase Distortion Deviation (PDD) (a measure of phase randomness) and linear-interpolated log F0 (total 86D). The vocoder parameters are extracted with PML vocoder [12], which does not use or depend on voicing decisions. The linguistic features consist of 592 binary linguistic features and 9 numerical features (total 601D).

3.2. Model Configurations

The d-vector-like speaker representation extraction model $\tilde{\boldsymbol{\theta}}_\lambda$ consists of 4 hidden ReLU maxout layers of 2 channels and 512 units each, similar to [8]. The output layer of d-vector baseline model is softmax. The input $\boldsymbol{y}_{\mathtt{t}}^{(\mathtt{s})}$, is the stacking of 40 consecutive frames of vocoder parameters (total 3440D).

The attention mechanism $\boldsymbol{\theta}_\alpha$ is a DNN with a normalising output layer on top of a $\tanh$ layer of size 16 and a sigmoid layer of size 1. The input $\boldsymbol{x}_{\mathtt{t}}^{(\mathtt{s})}$ is the stacking of 5 frames of linguistic features (total 3005D), which are spaced evenly in the corresponding 40 consecutive frames of vocoder parameters.

The "neural vocoder" $\boldsymbol{\theta}_{\mathtt{v}}$ is a DNN with 1 hidden ReLU maxout layer of 2 channels and 512 units, added to the input side of $\tilde{\boldsymbol{\theta}}_\lambda$. The input to $\boldsymbol{\theta}_{\mathtt{v}}$ is 3200 waveform samples at 16kHz, such that the corresponding duration is the same as 40 frames of vocoder parameters of 5ms each, and the input dimensions are similar as well.

The speech generation acoustic model $\boldsymbol{\theta}_{\mathtt{a}}$ has 7 hidden layers of size 2048 each, including 2 $\tanh$ layers, 3 LSTM layers, and another 2 $\tanh$ layers. The output layer is a linear layer. Residual connections [13] are applied to the $\tanh$ layers to ease the training with deeper acoustic model. Speaker representations $\boldsymbol{\lambda}^{(\mathtt{s})}$ are appended to every frame of the linguistic input and the hidden activations of all layers.

We use Merlin [14] and Tensorflow [15] to build the deep neural networks, and the optimiser is ADAM [16] with decaying learning rate.

3.3. Training and Evaluation Procedures

For the training of the baseline d-vector model, for those 189 training speakers, besides the 40 held-out utterances, another 40 utterances each are chosen for validation. After training, to generate a d-vector $\boldsymbol{\lambda}^{(\mathtt{s})}$ for any of the 209 speakers, all $\sim$360 utterances, including these 40 but excluding the 40 previously mentioned held-outs, are used.

For the training of the integrated frameworks, 3000 consecutive frames ($\sim$5 utterances) of enrolment data are used to estimate $\boldsymbol{\lambda}^{(\mathtt{s})}$ for optimising $\boldsymbol{\theta}_{\mathtt{a}}$. They are also used to estimate the gradient of $\mathcal{L}(\boldsymbol{\theta}_{\mathtt{a}}, \boldsymbol{\theta}_\lambda)$ with respect to all components of $\boldsymbol{\theta}_\lambda$. Those 3000 frames are sliced into 75 non-overlapping windows of 40 frames. At generation time, all $\sim$360 utterances of a speaker are sliced into non-overlapping windows of 40 frames to feed into $\boldsymbol{\theta}_\lambda$, and the outputs are averaged to estimate $\boldsymbol{\lambda}^{(\mathtt{s})}$.

For either two-stage or integrated optimisation, given the speaker representations $\{\hat{\boldsymbol{\lambda}}^{(1)}, \ldots, \hat{\boldsymbol{\lambda}}^{(\mathtt{s})}\}$ where $\hat{\boldsymbol{\lambda}}^{(\mathtt{s})} = \boldsymbol{f}(\mathcal{D}^{(\mathtt{s})}; \hat{\boldsymbol{\theta}}_\lambda)$, speaker-specific speech could be generated as the optimal output sequence in Eq. 4.

For the training of the integrated framework with "flat-attention", $\boldsymbol{\theta}_{\mathtt{a}}$, $\tilde{\boldsymbol{\theta}}_\lambda$, and $\boldsymbol{\theta}_{\mathtt{v}}$, are randomly initialised and jointly

optimised; for the training of the integrated framework with attention mechanism, $\boldsymbol{\theta}_{\mathrm{a}}$, $\boldsymbol{\theta}_{\lambda}$, and $\boldsymbol{\theta}_{\mathrm{v}}$ are initialised from a "flat-attention" model, and $\boldsymbol{\theta}_{\alpha}$ is pre-trained for 5 epochs before all system components are jointly optimised.

The objective evaluations include mel-cepstral distortion (MCD), PDD distortion (PDD), mean-square-error on fundamental frequency (F0), and correlation of F0 (F0-Corr). For subjective evaluations, we focus on the naturalness of speech and the similarity to the original speaker. Each listener taking the test assessed the 6 random utterances among ∼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 [17][18]. 36 listeners took the tests.

Next, we shall use "Two-Flat" to denote two-stage optimisation with flat-attention, "Int-Flat" to denote integrated optimisation with flat-attention, and "Int-Text" to denote integrated optimisation with text-dependent attention mechanism.

3.4. Integrated Optimisation

First, we compare two-stage optimisation and integrated optimisation. Since in our previous work, the same comparison has been made for extracting speaker representations from vocoder parameters [9], here the comparisons are based on extracting speaker representations from waveform.

Table 1: *Objective evaluations of different optimisation schemes*

feature	scheme	MCD	PDD	F0	F0-Corr
WAV	Two-Flat	1.373	1.106	33.284	0.761
WAV	Int-Flat	**1.176**	**1.024**	**30.824**	**0.810**

Table 2: *Naturalness and Speaker Similarity subjective evaluations of different optimisation schemes*

feature	scheme-1	Pref-1	No-pref	Pref-2	scheme-2
WAV	Two-Flat	34.8	21.4	**43.6**	Int-Flat
WAV	Two-Flat	25.5	17.3	**57.1**	Int-Flat

In Tables 1 and 2, integrated optimisation framework outperforms two-stage optimisation framework in every aspect of objective evaluation, as well as both naturalness and speaker similarity subjective evaluations. This confirms the hypothesis that task-specific speaker representations have better performance compared to task-independent speaker representations.

3.5. Text-Dependent Speaker Representation

Table 3: *Objective evaluations of different attention types*

feature	scheme	MCD	PDD	F0	F0-Corr
WAV	Int-Flat	1.176	1.024	30.824	0.810
WAV	Int-Text	**1.175**	**1.014**	**29.591**	**0.812**

Table 4: *Naturalness and Speaker Similarity subjective evaluations of different attention types*

feature	scheme-1	Pref-1	No-pref	Pref-2	scheme-2
WAV	Int-Flat	27.4	42.6	**29.9**	Int-Text
WAV	Int-Flat	24.2	48.4	**27.2**	Int-Text

Next, we compare flat-attention and text-dependent attention. Again, speaker representations are extracted from waveform. In Tables 3 and 4, text-dependent attention mechanism outperforms flat-attention model in every aspect of objective evaluation, as well as both naturalness and speaker similarity subjective evaluations. This confirms the hypothesis that the contextual linguistic features could provide rich information to improve the hitherto simple averaging process. However, the extent of improvement is less significant than the change from two-stage to integrated optimisation. One possible reason is that the attention mechanism is hard to train, and we expect better performance with further fine-tuning.

3.6. Waveform-Level Speaker Representation

Lastly, we compare speaker representations extracted from PML vocoder parameters and from waveform.

Table 5: *Objective evaluations of different features for speaker representation extraction*

feature	scheme	MCD	PDD	F0	F0-Corr
PML	Int-Text	**1.146**	**1.011**	30.583	0.799
WAV	Int-Text	1.175	1.014	**29.591**	**0.812**

Table 6: *Naturalness and Speaker Similarity subjective evaluations of different features for speaker representation extraction*

scheme	feat.-1	Pref-1	No-pref	Pref-2	feat.-2
Int-Text	PML	**52.4**	31.3	16.1	WAV
Int-Text	PML	**45.9**	27.3	26.7	WAV

In the objective evaluations, PML outperforms WAV in both MCD and PDD, while WAV outperforms PML in both measures related to F0. Therefore we expect that in subjective evaluations, PML would have better naturalness, and WAV would have better speaker similarity. However, in subjective evaluations, PML outperforms WAV in every comparison. One main reason is that this "neural vocoder" is too simple and too sensitive to the starting position of the 3200-sample sequence. We found that the cosine distance between $\boldsymbol{\lambda}_{\mathrm{t}}^{(\mathrm{s})}$ and $\boldsymbol{\lambda}_{\mathrm{t}+\tau}^{(\mathrm{s})}$ is much bigger than those extracted from PML parameters, for small shifts of τ by just a few samples at 16kHz. Therefore, this "neural vocoder" failed to handle the positions of the pitches of glottal cycles, while the vocoder parameter extraction process involves pitch-dependent windowing.

In future work, a more complex "neural vocoder" is required, for example an RNN-encoder to encapsulate the entire sequence, or a stacked-CNN with max-pooling to reduce positional sensitivity.

4. Conclusions

In this work, we extended our integrated training framework to extract speaker representations from waveform. We verified the improvements of the integrated optimisation and text-dependent attention mechanism under this new framework, and the novelty of this work is that the speaker representations are extracted directly from raw waveform, without a pre-defined vocoder.

Compared to extracting speaker representations from vocoder parameters, this new framework, with a rather simple "neural vocoder", improves F0 trajectories, even without receiving specific F0 information. It is promising to improve other objective and subjective measures in the future, with a more complex model.

5. 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] 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.

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

[6] 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.

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

[8] E. Variani, X. Lei, E. McDermott, I. L. Moreno, and J. Gonzalez-Dominguez, "Deep neural networks for small footprint text-dependent speaker verification," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 4052–4056.

[9] M. Wan, G. Degottex, and M. J. Gales, "Integrated speaker-adaptive speech synthesis."

[10] H. Lee, P. Pham, Y. Largman, and A. Y. Ng, "Unsupervised feature learning for audio classification using convolutional deep belief networks," in *Advances in neural information processing systems*, 2009, pp. 1096–1104.

[11] 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.

[12] G. Degottex, P. Lanchantin, and M. Gales, "A log domain pulse model for parametric speech synthesis," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 26, no. 1, pp. 57–70, 2018.

[13] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015. [Online]. Available: http://arxiv.org/abs/1512.03385

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

[15] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[16] 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

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

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