PRELIMINARY WORK ON SPEAKER
ADAPTATION FOR DNN-BASED SPEECH
SYNTHESIS

Blaise Potard       Petr Motlicek       David Imseng

Idiap-RR-02-2015

JANUARY 2015
Preliminary Work on Speaker Adaptation for DNN-Based Speech Synthesis

Blaise Potard, Petr Motlicek, David Imseng

Idiap Research Institute
Martigny, Switzerland

{bpotard,pmotlic,dimseng}@idiap.ch

Abstract

We investigate speaker adaptation in the context of deep neural network (DNN) based speech synthesis. More specifically, our current work focuses on the exploitation of auxiliary information such as gender, speaker identity or age during the DNN training process. The proposed technique is compared to standard acoustic feature transformations such as the feature based maximum likelihood linear regression (FMLLR) based speaker adaptation. Objective error measurements as well as perceptual experiments, performed on the WSJCAM0 database, suggest that the proposed method is superior to standard feature transformations.

Index Terms: DNN, TTS, speech synthesis, neural network, fmllr, speaker adaptation

1. Introduction

Deep neural networks (DNN) have recently emerged as one of the most common machine learning methods for speech-based applications, in particular for Automatic Speech Recognition (ASR). Paradoxically, even though neural networks were already applied for speech synthesis at the end of the 80s [1], and despite a high level of activity in parameter-based speech synthesis based on hidden Markov models (HMM) [2], there is very little in terms of published research applying DNNs to Text-to-Speech Synthesis (TTS). In particular, the authors found no article at all applying neural networks to speaker adaptive training for TTS.

One of the main advantages of using parametric synthesis methods, as opposed to e.g. unit selection, is their ability to modify the characteristics of an original voice by adapting it to another speaker, with very small amounts of data [3]. Extensive research has been conducted on this topic in the context of HMM-based synthesis: the most commonly used techniques are maximum a posteriori (MAP) estimation [4, 5] and maximum-likelihood linear regression (MLLR) [6, 7]. Speaker adaptation is fairly common in HMM-based TTS or ASR systems, and it is common as well in DNN-based ASR, by either using speaker adapted transformations on acoustic features – generally CMLLR transformations derived from ML-based optimization in HMM forced alignment [8] – or by retraining the neural network on acoustic data from the target speaker [9]. Speaker adaptation has however, to the best of our knowledge, not yet been applied on DNN-based TTS.

In this paper, we propose to transpose state-of-the-art speaker adaptation techniques as used in ASR, for DNN-based speech synthesis; and investigate the speaker adaptation schemes that provide the best performances on the WSJCAM0 database.

This paper is structured as follows: Section 2 presents the neural network framework used, while Section 3 presents generalities about speaker adaptation techniques. Section 4 presents our experimental framework, and finally Section 5 presents the evaluations we conducted.

2. DNN framework

Our DNN-based speech synthesis approach is inspired by Zen et al. [10].

2.1. Input features

A given text to be synthesized is converted in a sequence of input features. In our case, the input feature stream employed for DNN training is a direct mapping of a simplified version of the HTS “full” labels – which, on top of segmental information, contain rich contextual parameters such as lexical stress and position in syllable.

The labels were simplified in the following way: we used a triphone context instead of the default quinphone context, and the feature set was shrunk to only retain information up to the syllabic level. This is inspired by our previous work [11] which suggests that higher level information does not provide significant improvement for the synthesis of English. To obtain a direct mapping of input labels and acoustic features, the segment-based labels have to be converted to retain exactly one input label per acoustic frame. To differentiate the various frames from the original segment, we incorporate information relative to the segment duration, and to the frame position within the segment.

For speaker discriminative training, we simply need to add additional information to the input label. In practice, we incorporate either a single binary value for gender-based training, or a set of binary values (one binary value per speaker) for full speaker discriminative training.

Mapping to numeric values is relatively straightforward: for instance, phonemic information is converted to binary arrays – one binary value per phoneme type – while numeric values such as relative syllable position remain unmodified.

2.2. Output features

The output acoustic feature stream contain spectral (mel-cepstrum) and excitation parameters (pitch and aperiodic energy), as well as their time derivatives of first and second order. They are generated through publically available scripts from the EMME [12] project, using STRAIGHT [13] acoustic analysis. Each component is normalized per speaker.
2.3. DNN architecture
The goal of the DNN is to learn a mapping between the input features (labels) and output features (acoustic) on a train set. We used a slightly modified version of the Kaldi [14] toolkit for the DNN training.

In our experiments, the DNN comprises three hidden layers. Subsequent synthesis of the test set can then be performed using forward propagation of input features through the trained DNN.

Unlike the work of Zen et al., we did not apply any post-filtering on the acoustic features generated at the output of the DNN; the raw acoustic features are directly fed to the vocoder (i.e., their time derivatives are discarded).

3. Speaker adaptation techniques
Similarly to HMM-based synthesis techniques, we wish to create, using DNNs, “average voice” models from multiple speakers’ recordings – which requires compensating for the acoustic mismatch across speakers. Using DNN, two classes of methods can be used for reducing this acoustic mismatch: explicit transformations of the acoustic features provided to the DNN for training, e.g., using fMLLR transforms trained separately, or implicit transformations by providing information about speaker identity in the DNN input (such as gender, age, or physiology related factors such as vocal tract length). In this paper, we compare 3 implicit methods purely relying on DNN, to 2 explicit features transformations method using fMLLR transforms.

3.1. fMLLR transforms
Feature-space adaptation using feature-space MLLR (fMLLR), also known as constrained MLLR [15], was applied in our experiments. fMLLR was used through performing a Speaker Adaptive Training (SAT) of an HMM/GMM baseline ASR system. Resulting feature transforms were estimated in an unsupervised way from the training and test sets, for either each speaker, or for speakers grouped by gender. The acoustic features used were identical to the ones used for training the DNN.

3.2. Average voice models
The average voice approach is based on an idea of using data from multiple speakers to train HMM/GMM models in a SAT fashion [3].

For the fMLLR-based discriminative training, the input feature set was not modified (i.e., does not contain any speaker specific information).

For DNN speaker-based discriminative training, the input features were modified by adding either a binary value (for gender-based adaptation), or a set of binary values (for speaker-based adaptation).

As mentioned in Sect. 3.1, two different classes of fMLLR transformations were trained: the first trained per speaker (marked fmllr), and the second trained per gender (marked fmllrG).

In total, 5 DNNs were thus trained to create average voice models:

- **DNNG**: “baseline” DNN system trained without any speaker information.
- **DNNG*:**: DNN system trained with gender information.
- **DNNGid**: DNN system trained with speaker ID information.
- **D+F*:**: DNN system trained on fmllrG transformed acoustic features.
- **D+F**: DNN system trained on fmllr transformed acoustic features.

Each system can then be further adapted to the target speaker.

3.3. Adaptation to speaker
We used different adaptation techniques for D+F*, and DNN, systems.

For the DNN, systems, an adaptation pass was run by retraining the DNN using a small amount of data (40 sentences) from the target speaker. The final layer was not randomized. Cross validation and testing was run on the remaining 60 sentences.

For the D+F* systems, an additional layer initialized with the fmllr parameters for the target speaker was added to the network: indeed the fmllr transform for a given speaker is simply an affine transform, which can thus be transposed directly as a linear layer in the DNN. This was intended to investigate whether the DNN could “improve” the fmllr transform. Adapted DNN models will be marked with the suffix “a”, e.g. **DNNGa**.

4. Experimental setup
Experiments were performed on the WSJCAM0 database, which is commonly used in HMM-based speech synthesis.

4.1. Label/acoustic data
As part of the EMIME [12] project, considerable work has been performed on the WSJCAM0 [16] database. We therefore employed the same “full” labels as used in EMIME for training HTS systems (although we adjusted timing of the labels through a forced alignment with the Kaldi toolkit before using them). First, the labels were turned into numeric values: phonemic information was turned into binary arrays, values indicating the position in syllables, etc. were kept as numeric values. Note that some of the components of the “full” labels as used in HTS were discarded, as indicated in [11]. Then, the segments were expanded so that the input vectors have the same frame-rate as the acoustic data. Finally, the frame position information was added to the input vector: duration (in ms) of the original segment, and the relative position of the frame within the segment (normalized to be between 0 and 1). Overall, the numeric labels / input vectors comprised 176 components for DNN and both fmllr-based systems. DNNG and DNNG used respectively 177 and 268 components.

The acoustic data was extracted through the use of the EMIME feature extraction scripts, with the default settings: 40 mel-cepstrum coefficients, single value for log pitch frequency, and 21 parameters for aperiodic excitation; plus their respective first and second order time derivatives. The pitch component was turned into a continuous pitch by linearly interpolating in unvoiced segments, and adding a binary value indicating the voiced / unvoiced status. In total, the acoustic vectors comprised 189 components.

4.2. Neural network
In general, the neural networks we used were built with 3 hidden layers, with 700 nodes in each layer. Each layer comprised an affine component followed by a sigmoid function. The input data was normalized for each component to be of zero mean and unit variance. The output data was normalized to be of...
zero mean per speaker, then renormalized globally so that each component had values between 0.01 and 0.99. Note that apart from the number of nodes and layers, this setup is similar to the one described by Zen.

The training set of WSJCAM0 was split into two parts: 82 speakers were kept for DNN training, while 10 were used for the cross-validation.

Five average voice models were built for the five types of DNN systems, described in Sect. 3.2. Each of these system was then adapted to two speakers not present in the training set: male speaker “c2d” and female speaker “c2j”, using 40 sentences for training, and the remaining for cross-validation.

4.3. HMM models

For comparisons with state-of-the-art parametric systems, HMM-based synthesis models were built using the same database, using the HTS v.2.1 toolkit [17].

More specifically, an implementation from the EMIME project [12] was employed. We used traditional five-state left-to-right Hidden Semi-Markov Models (HSMM), with no-skip. The speech data used was sampled at 16 kHz.

As mentioned in Sect. 3.1, an average voice model was first built using SAT training using the WSJCAM0 [16] database. The model was then adapted to two test speakers, “c2d” and “c2j”.

The same 40 sentences from each speaker, previously applied in DNN adaptation, were also applied for training the HMM adapted models. As an adaptation, the constrained structural maximum a posteriori linear regression (CSMAPLR) [3] approach was used. We exploited STRAIGHT [18] for the analysis and synthesis phase of the HSMM-based speech synthesis.

5. Evaluation

Several kinds of evaluation measures were run to estimate the quality of speech synthesized using the DNNs: two different types of objectives measures, and one subjective listening test.

5.1. Objective measures

First, we applied Perceptual Evaluation of Speech Quality (PESQ) measure [19] to the whole test recordings related to our 2 reference speakers. PESQ results correspond to the average user perception of the speech sample under assessment PESQ - MOS (Mean Opinion Score: 1 - very annoying; 5 - imperceptible impairment).

Second, we used MCD (Mel Cepstral Distortion [20]) as a measure of synthesis error. Note that the distortion is normalized and lower values indicate smaller distortion.

Table 1 presents the PESQ (MOS)\(^1\) - as well as the MCD results. Scores obtained for the male (“c2d”) and female (“c2j”) speakers are indicated separately.

PESQ measure should be interpreted as a MOS regarding the similarity to the original. We can observe that all systems have relatively low PESQ scores. Note that even the scores obtained by the vocoder itself are relatively low (i.e., a good vocoder should obtain scores around 4.0\(^2\)). Further, several systems get decent PESQ scores for the male speaker: DNN\(_G\), DNN\(_{Ga}\), DNN\(_{Ba}\). HTS samples and both D+F systems get acceptable scores, although slightly lower.

Regarding the female speaker, all systems obtain much lower PESQ scores. The system that obtain the best scores is HTS, closely followed by the baseline DNN system adapted to the speaker (DNN\(_g\)).

Regarding the MCD, the best system surprisingly appears to be DNN\(_G\). Among systems adapted to the speaker, DNN\(_{Ga}\) seems to yield the best results overall, although HTS seems to perform slightly better for the male speaker.

5.2. Subjective Evaluation

To verify whether the previous objective judgments stand in front of a real human listener, we conducted a listening test opposing the best systems of each class: DNN-based system trained with gender information, opposed to fmllr\(_g\)-based DNN system; both systems were adapted to the speaker.

As the objective score differences were rather small, we conducted an ABX test – as opposed to a MOS test – in the hope of emphasizing the differences between the systems.

Two sentences for each target speakers were selected, and generated using each system. An ABX comparison was performed so that listeners were asked to decide which sample they preferred in terms of overall quality. For each pair of samples, they were allowed to choose between 5 preference levels: strong preference for sample A, small preference for sample A, no preference, small preference for sample B, strong preference for sample B.

As mentioned above, we only compared two systems: DNN\(_{Ga}\), and D+F\(_{Ga}\); the vocoded samples were used as reference. There were thus 3 systems in total, 3 pairs of systems, with 4 stimuli in each case. 13 listeners took part in the test. The order of the tests was randomized for each listener. Figure 1 presents the distribution of replies for each system pair.

The DNN systems appear to not be comparable with vocoded samples, as 100% of listeners marked a preference for vocoded speech. One possible explanation is that the DNN samples were much noisier than the vocoded samples, in particular, the “silent” parts of samples. Note that contrary to the usual HTS approach, or to Zen’s DNN approach, we did not perform any special treatment for silences, and remarkably, the breathing noise at the beginning or end of sentences were correctly

---

\(^1\)With the reference implementation of the PESQ tool, the performance of HTS samples was extremely bad; this appeared to be due to the aligner built in the tool, which seemed to be confused by HTS silence modeling. We disabled the alignment of audio files in “pseudomain.c” to obtain the presented results. Note that this procedure only increased the scores of HTS samples, and did not modify the score of other samples.

\(^2\)In particular, the speex voice coder gets better PESQ scores with bit-rates about 8kbps.
modeled.

When comparing the two DNN systems, it appears there is a slight preference for the pure DNN approach, which is consistent with the objective measures. This preference may not be generalizable however, since the listening and objective tests were performed on two speakers only.

6. Conclusions

While the subjective listening tests indicate that DNN based synthesis does not yet provide comparable quality with respect to the original speech, the objective measurements indicate that regardless of perceptual quality, the DNN can generate speech that is more realistic in terms of objective closeness to the original, than HTS samples. This indicates a potential suitability for applications such as very low bit-rate coding.

The adaptation technique that lead to the best results was the DNN-based approach using gender information, which was in all cases superior to the fmllr based approaches applied either per-gender or per-speaker. However, this may be due to the small amount of data available for each speaker in the given WSJCAM0 corpus.

Future work will investigate the use of audio smoothing, such as the Maximum-likelihood distribution optimization usually used in HTS, as well as the inclusion of additional information in the labels, such as the questions used in HTS models training. Finally, we will also focus on silence modelling, which seems to be partly responsible for strong noise in DNN-based synthesis.

7. References


Table 1: PESQ - MOS and MCD objective quality measures of 4 different speech synthesis techniques presented for 2 different reference speakers from the WSJCAM0 database. For each objective quality metric, the first line ("Average") indicates the results of the average voice models, while the second line ("Adapted") indicates the performance of the adapted voice models.