

VOICE CLONING: A MULTI-SPEAKER TEXT-TO-SPEECH SYNTHESIS APPROACH BASED ON TRANSFER LEARNING

Abstract

Deep learning models are becoming predominant in many fields of machine learning. Text-to-Speech (TTS), the process of synthesizing artificial speech from text, is no exception. To this end, a deep neural network is usually trained using a corpus of several hours of recorded speech from a single speaker. Trying to produce the voice of a speaker other than the one learned is expensive and requires large effort since it is necessary to record a new dataset and retrain the model. This is the main reason why the TTS models are usually single speaker. The proposed approach has the goal to overcome these limitations trying to obtain a system which is able to model a multi-speaker acoustic space. This allows the generation of speech audio similar to the voice of different target speakers, even if they were not observed during the training phase.

Index Terms— text-to-speech, deep learning, multi-speaker speech synthesis, speaker embedding, transfer learning

1 Introduction

Text-to-Speech (TTS) synthesis, the process of generating natural speech from text, remains a challenging task despite decades of investigation. Nowadays there are several TTS systems able to get impressive results in terms of synthesis of natural voices very close to human ones. Unfortunately, many of these systems learn to synthesize text only with a single voice. The goal of this work is to build a TTS system which can generate in a data efficient manner natural speech for a wide variety of speakers, not necessarily seen during the training phase. The activity that allows the creation of this type of models is called Voice Cloning and has many applications, such as restoring the ability to communicate naturally to users who have lost their voice or customizing digital assistants such as Siri.

Over time, there has been a significant interest in end-to-end TTS models trained directly from text-audio pairs; Tacotron 2 [1] used WaveNet [2] as a vocoder to invert spectrograms generated by sequence-to-sequence with attention [3] model architecture that encodes text and decodes spectrograms, obtaining a naturalness close to the human one. It only supported a single speaker. Gibiansky et al. [4] proposed a multi-speaker variation of Tacotron able to learn a low-dimensional speaker embedding for each training speaker. Deep Voice 3 [5] introduced a fully convolutional encoder-decoder architecture which supports thousands of speakers from LibriSpeech [6]. However, these systems only support synthesis of voices seen during training since they learn a fixed set of speaker embeddings. Voiceloop [7] proposed a novel archi-

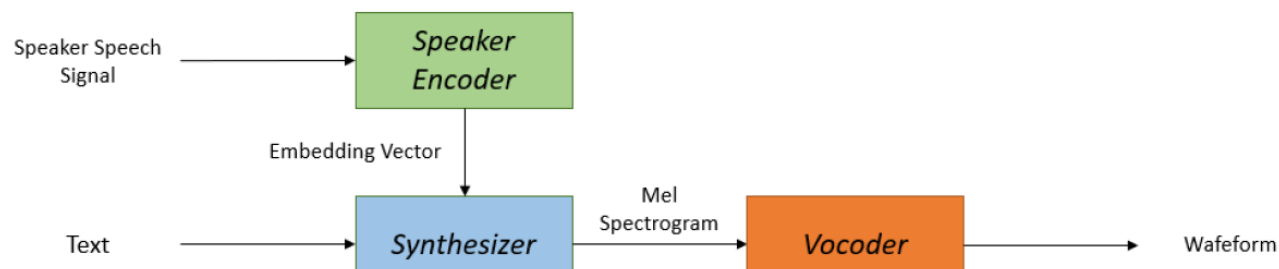
ture which can generate speech from voices unseen during training but requires tens of minutes of speech and transcripts of the target speaker. In recent extensions, only a few seconds of speech per speaker can be used to generate new speech in that speaker's voice. Nachmani et al. [8] for example, extended Voiceloop to utilize a target speaker encoding network to predict speaker embedding directly from a spectrogram. This network is jointly trained with the synthesis network to ensure that embeddings predicted from utterances by the same speaker are closer than embeddings computed from different speakers. Jia et al. [9] proposed a speaker encoder model similar to [8], except that they used a network independently-trained exploring transfer learning from a pre-trained speaker verification model towards the synthesis model.

This work is similar to [9] however introduces different architectures and uses a new transfer learning technique still based on a pre-trained speaker verification model but exploiting utterance embeddings rather than speaker embeddings. In addition, we use a different strategy to condition the speech synthesis with the voice of speakers not observed before and compared several neural architectures for the speaker encoder model. The paper is organized as follows: Section 2 describes the model architecture and its formal definition; Section 3 reports experiments and results done to evaluate the proposed solution; finally conclusions are reported in Section 4.

2 Model Architecture

Following [9], the proposed system consists of three components: a *speaker encoder*, which computes a fixed-dimensional embedding vector from a few seconds of reference speech of a target speaker; a *synthesizer*, which predicts a mel spectrogram from an input text and an embedding vector; a *neural vocoder*, which infers time-domain waveforms from the mel spectrograms generated by the synthesizer. At inference time, the speaker encoder takes as input a short reference utterance of the target speaker and generates, according to its internal learned speaker characteristics space, an embedding vector. The synthesizer takes as input a phoneme (or grapheme) sequence and generates a mel spectrogram, conditioned by the speaker encoder embedding vector. Finally the vocoder takes the output of the synthesizer and generates the speech waveform. This is illustrated in Figure 1.

2.1 Problem Definition






Fig. 1: High level overview of the three components of the system.

Consider a dataset of N speakers each of which has M utterances in the time-domain. Let's denote the j -th utterance of the i -th speaker as \mathbf{u}_{ij} while the feature extracted from the j -th utterance of the i -th speaker as \mathbf{x}_{ij} ($1 \leq i \leq N$ and $1 \leq j \leq M$). We chose as feature vector \mathbf{x}_{ij} the mel spectrogram.

The speaker encoder \mathcal{E} has the task to produce meaningful embedding vectors that should characterize the voices of the speakers. It computes the embedding vector \mathbf{e}_{ij} corresponding to the utterance \mathbf{u}_{ij} as:

$$\mathbf{e}_{ij} = \mathcal{E}(\mathbf{x}_{ij}; \mathbf{w}_{\mathcal{E}}) \quad (1)$$

where $\mathbf{w}_{\mathcal{E}}$ represents the encoder model parameters. Let's define it *utterance embedding*. In addition to defining embedding at the utterance level, we can also define the *speaker embedding*:

$$\mathbf{c}_i = \frac{1}{n} \sum_{j=1}^n \mathbf{e}_{ij} \quad (2)$$

In [9], the synthesizer \mathcal{S} predicts \mathbf{x}_{ij} given \mathbf{c}_i and \mathbf{t}_{ij} , the transcript of the utterance \mathbf{u}_{ij} :

$$\hat{\mathbf{x}}_{ij} = \mathcal{S}(\mathbf{c}_i, \mathbf{t}_{ij}; \mathbf{w}_{\mathcal{S}}) \quad (3)$$

where $\mathbf{w}_{\mathcal{S}}$ represents the synthesizer model parameters. In our approach, we propose to use the utterance embedding rather than the speaker embedding:

$$\hat{\mathbf{x}}_{ij} = \mathcal{S}(\mathbf{e}_{ij}, \mathbf{t}_{ij}; \mathbf{w}_{\mathcal{S}}) \quad (4)$$

We will motivate this choice in Paragraph 2.4.

Finally, the vocoder \mathcal{V} generates \mathbf{u}_{ij} given $\hat{\mathbf{x}}_{ij}$. So we have:

$$\hat{\mathbf{u}}_{ij} = \mathcal{V}(\hat{\mathbf{x}}_{ij}; \mathbf{w}_{\mathcal{V}}) \quad (5)$$

where $\mathbf{w}_{\mathcal{V}}$ represents the vocoder model parameters.

This system could be trained in an end-to-end mode trying to optimize the following objective function:

$$\min_{\mathbf{w}_{\mathcal{E}}, \mathbf{w}_{\mathcal{S}}, \mathbf{w}_{\mathcal{V}}} L_{\mathcal{V}}(\mathbf{u}_{ij}, \mathcal{V}(\mathcal{S}(\mathcal{E}(\mathbf{x}_{ij}; \mathbf{w}_{\mathcal{E}}), \mathbf{t}_{ij}; \mathbf{w}_{\mathcal{S}}); \mathbf{w}_{\mathcal{V}})) \quad (6)$$

where $L_{\mathcal{V}}$ is a loss function in the time-domain. However, it requires to train the three models using the same dataset, moreover, the convergence of the combined model could be hard to reach. To overcome this drawback, the synthesizer can be trained independently to directly predict the mel spectrogram \mathbf{x}_{ij} of a target utterance \mathbf{u}_{ij} trying to optimize the following objective function:

$$\min_{\mathbf{w}_{\mathcal{S}}} L_{\mathcal{S}}(\mathbf{x}_{ij}, \mathcal{S}(\mathbf{e}_{ij}, \mathbf{t}_{ij}; \mathbf{w}_{\mathcal{S}})) \quad (7)$$

where $L_{\mathcal{S}}$ is a loss function in the time-frequency domain. It is necessary to have a pre-trained speaker encoder model available to compute the utterance embedding \mathbf{e}_{ij} .

The vocoder can be trained either directly on the mel spectrograms predicted by the synthesizer or on the groundtruth mel spectrograms:

$$\min_{\mathbf{w}_{\mathcal{V}}} L_{\mathcal{V}}(\mathbf{u}_{ij}, \mathcal{V}(\mathbf{x}_{ij}; \mathbf{w}_{\mathcal{V}})) \text{ or } \min_{\mathbf{w}_{\mathcal{V}}} L_{\mathcal{V}}(\mathbf{u}_{ij}, \mathcal{V}(\hat{\mathbf{x}}_{ij}; \mathbf{w}_{\mathcal{V}})) \quad (8)$$

where $L_{\mathcal{V}}$ is a loss function in the time-domain. In the second case, a pre-trained synthesizer model is needed.

If the definition of the objective function was quite simple for both the synthesizer and the vocoder, unfortunately this is not the case for the speaker encoder. The encoder does not have labels to be trained on because its task is only to create the space of characteristics necessary to create the embedding vectors. The Generalized End-to-End (GE2E) [10] loss brings a solution to this problem and it allows the training of the speaker encoder independently. Consequently, we can define the following objective function:

$$\min_{\mathbf{w}_{\mathcal{E}}} L_{\mathcal{G}}(\mathbf{S}; w_{\mathcal{E}}) = \sum_{j,i} L(\mathbf{e}_{ji}) \quad (9)$$

where \mathbf{S} represents a similarity matrix and $L_{\mathcal{G}}$ is the GE2E loss function.

2.2 Speaker Encoder

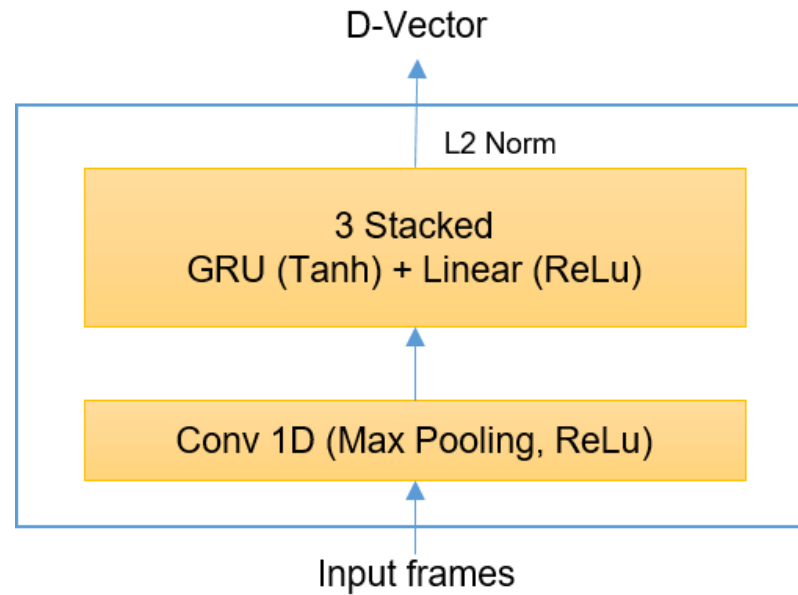


Fig. 2: Speaker encoder model architecture. Input is composed of a time sequence of dimension 40. The last linear layer takes the hidden state of the last GRU layer as input.

The speaker encoder must be able to produce an embedding vector that meaningfully represents speaker characteristics in the transformed space starting from a target speaker's utterance. Furthermore, the model should identify these characteristics using a short speech signal, regardless of its phonetic content and background noise. This can be achieved by training a neural network model on a text-independent speaker verification task that tries to optimize the GE2E loss so that embeddings of utterances from the same speaker have high cosine similarity, while those of utterances from different speakers are far apart in the embedding space.

The network maps a sequence of mel spectrogram frames to a fixed-dimensional embedding vector, known as d-vector [11, 12]. Input mel spectrograms are fed to a network consisting of one Conv1D [13] layer of 512 units followed by a stack

of 3 GRU [14] layers of 512 units, each followed by a linear projection of 256 dimension. Following [9], the final embedding dimension is 256 and it is created by L2-normalizing the output of the top layer at the final frame. This is shown in Figure 2. We noticed that this architecture was the best among the various tried and tested, as we will see in Section 3.

During the training phase, all the utterances are split into partial utterances that are 1.6 seconds long (160 frames). Also at inference time, the input utterance is split into segments of 1.6 seconds with 50% overlap and the model processes each segment individually. Following [9, 10], the final utterance-wise d-vector is generated by L2 normalizing the window-wise d-vectors and taking the element-wise average.

2.3 Synthesizer and Vocoder

The synthesizer component of the system is a sequence-to-sequence model with attention [1, 3] which is trained on pairs

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio, "Neural machine translation by jointly learning to align and translate," *CoRR*, vol. abs/1409.0473, 2015.

the network is trained in a
encoder to extract embed-
adopted vocoder compo-

nent of the system is a Pytorch github implementation¹

¹<https://github.com/fatchord/WaveRNN>

of the neural vocoder WaveRNN [15]. This model is not directly conditioned on the output of the speaker encoder but just on the input mel spectrogram. The multi-speaker vocoder is simply trained by using data from many speakers (see Section 3).

2.4 Transfer Learning Modality

The conditioning of the synthesizer via speaker encoder is the fundamental part that makes the system multi-speaker: the embedding vectors computed by the speaker encoder allow the conditioning of the mel spectrograms generated by the synthesizer so that they can incorporate the new speaker voice. In [9], the embedding vectors are speaker embeddings obtained by Equation 2. We used the utterance embeddings computed by Equation 1. In fact, at inference time only one utterance of the target speaker is fed to the speaker encoder which therefore produces a single utterance-level d-vector. Thus, in this case, it is not possible to create an embedding at the speaker level since the average operation cannot be applied. This implies that only utterance embeddings can be used during the inference phase. In addition, an average mechanism could cause some loss in terms of accuracy. This is due to larger variations in pitch and voice quality often occurring in utterances of the same speaker while utterances have lower intra-variation. Following [9], the embedding vectors computed by the speaker encoder are concatenated only with the synthesizer encoder output in order to condition the synthesis. However, we experimented with a new concatenation technique: first we passed the embedding through a single linear layer and then we applied the concatenation between the output of this layer and the synthesizer encoder one. The goal was to exploit the weights of the linear layer to make the embedding vector more meaningful,

since the layer was trained together with the synthesizer. We noticed that this method achieved good convergence of training and was about 75% times faster than the former vector concatenation.

3 Experiments and Results

We used different publicly available datasets to train and evaluate the components of the system. For the speaker encoder, different neural network architectures were tested. Each of them was trained using a combination of three public sets: LibriTTS [16] train-other and dev-other; VoxCeleb [17] dev and VoxCeleb2 [18] dev. In this way, we obtained a number of speakers equal to 8,381 and a number of utterances equal to 1,419,192, not necessarily all clean and noiseless. Furthermore, transcripts were not required. The models were trained using Adam [19] as optimizer with an initial learning rate equal to 0.001. Moreover, we experimented with different learning rate decay strategies.

During the evaluation phase, we used a combination of the corresponding test sets of the training ones, obtaining a number of speakers equal to 191 and a number of utterances equal to 45,132. Both training and test sets have been sampled at 16 kHz and input mel spectrograms were computed from 25ms STFT analysis windows with a 10ms step and passed through a 40-channel mel-scale filterbank.

We separately trained the synthesizer and the vocoder using the same training set given by the combination of the two “clean” sets of LibriTTS, obtaining a number of speakers equal to 1,151, a number of utterances equal to 149,736 and a total number of hours equal to 245,14 of 22.05 kHz audio. We trained the synthesizer using the L1 loss [20] and Adam as optimizer. Moreover, the input texts were converted into phoneme sequences and target mel spectrogram features are computed on 50 ms signal windows, shifted by 12.5 ms and passed through an 80-channel mel-scale filterbank. The vocoder was trained using groundtruth waveforms rather than the synthesizer outputs.

3.1 Baseline System

We choose as baseline for our work the Corentin Jemine’s real-time voice cloning system [21], a public re-implementation of the Google system [9] available on github²

²<https://github.com/CorentinJ/Real-Time-Voice-Cloning>

. This system is composed out of three components: a recurrent speaker encoder consisting of 3 LSTM [22] layers and a final linear layer, each of which has 256 units; a sequence-to-sequence with attention synthesizer based on [1] and Wave-RNN [15] as vocoder.

3.2 Speaker Encoder: Proposed System

To evaluate all the speaker encoder models and choose the best one, the Speaker Verification Equal Error Rate (SV-EER) was estimated by pairing each test utterance with each enrollment speaker. The models implemented are:

- *rec_conv network*: 5 Conv1D layers, 1 GRU layer and a final linear layer;
- *rec_conv_2 network*: 3 Conv1D layers, 2 GRU layers each followed by a linear projection layer;
- *gru network*: 3 GRU layers each followed by a linear projection layer;
- *advanced_gru network*: 1 Conv1D layer and 3 GRU layers each followed by a linear projection layer (Figure 2);
- *lstm network*: 1 Conv1D layer and 3 LSTM [22] layers each followed by a linear projection layer.

All layers have 512 units except the linear ones which have 256. Moreover, dropout rate of 0.2 was used after all the layers except before the first and after the last. All the models were trained using a batch size of 64 speakers and 10 utterances for each speaker. The results obtained are shown in Table 1.

Table 1: Speaker Verification Equal Error Rates.

Name	Step Time	Train Loss	SV-EER	LR Decay
rec_conv	0.33s	0.36	0.073	Reduce on Plateau
rec_conv_2	0.45s	0.49	0.075	Reduce on Plateau
gru	1.45s	0.33	0.054	Every 100,000 step
advanced_gru	0.86s	0.14	0.040	Exponential
lstm	1.08s	0.17	0.052	Exponential

We designed the advanced gru network trying to combine the advantages of convolutional and gru networks. In fact, looking at the table, this architecture was much faster than the gru network during training, and obtained the best SV-EER on the test set. Figure 3 illustrates the projection in a two-dimensional space of the utterance embeddings computed by the advanced gru network on the basis of 6 utterances extracted from 12 speakers of the test set. In Figure 4, the 12 speakers are 6 men and 6 women. The projections were made using UMAP [23]. Both the figures show that the model has created a space of internal features that is robust regarding the speakers, creating well-formed clusters of speakers based on their utterances and nicely separating male speakers from female ones.

The SV-EER obtained on the test set from the speaker encoder model of the proposed system is 0.040 vs the baseline one which is 0.049.

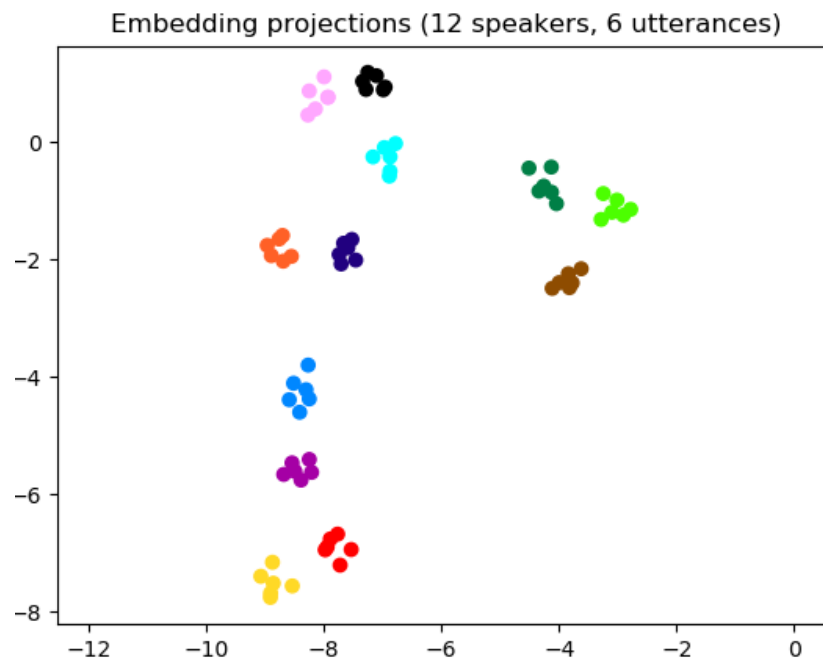
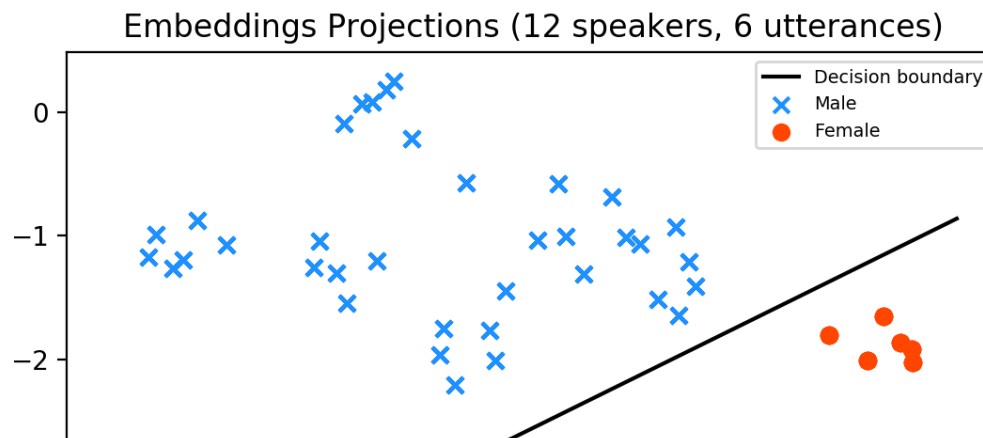


Fig. 3: Advanced Gru Network test utterance embeddings projection.



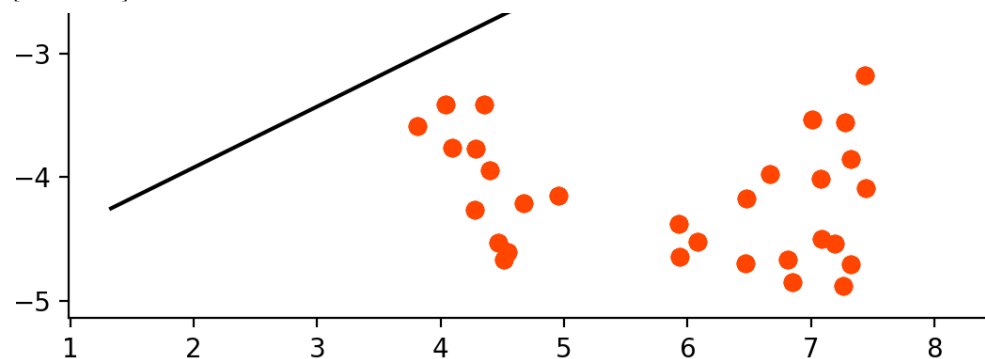
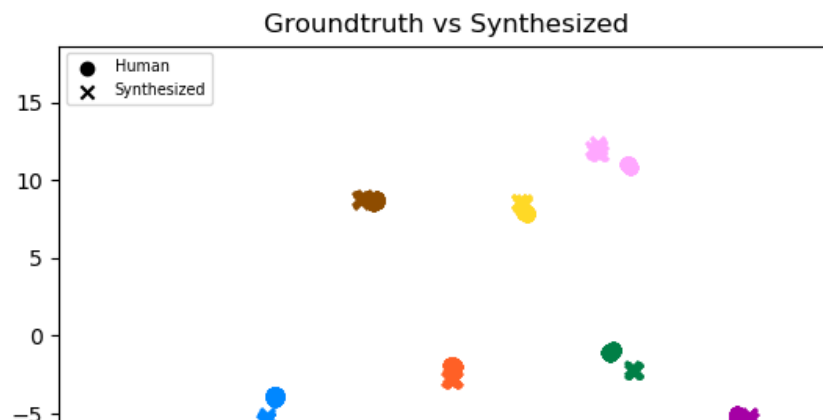


Fig. 4: Advanced Gru Network six utterances for six male vs six utterances for six female taken from the test set.

3.3 Similarity Evaluation

To assess how similar the waveforms generated by the system were from the original ones, we transformed the audio signals produced into utterance embeddings (using the speaker encoder advanced gru network) and then projected them in a two-dimensional space together with the utterance embeddings computed on the basis of the groundtruth audio. As test speakers, we randomly choose eight target speakers: four speakers (two male and two female) were extracted from the test-set-clean of LibriTTS [16], three (two male and one female) from VCTK [24] and finally a female proprietary voice. For each speaker we randomly extracted 10 utterances and compared them with the utterances generated by the system calculating the cosine similarity. The speakers averaged values of cosine similarity between the generated and groundtruth utterance embeddings range from 0.56 to 0.76. Figure 5 shows that synthesized utterances tend to lie close to real speech from the same speaker in the embedding space.



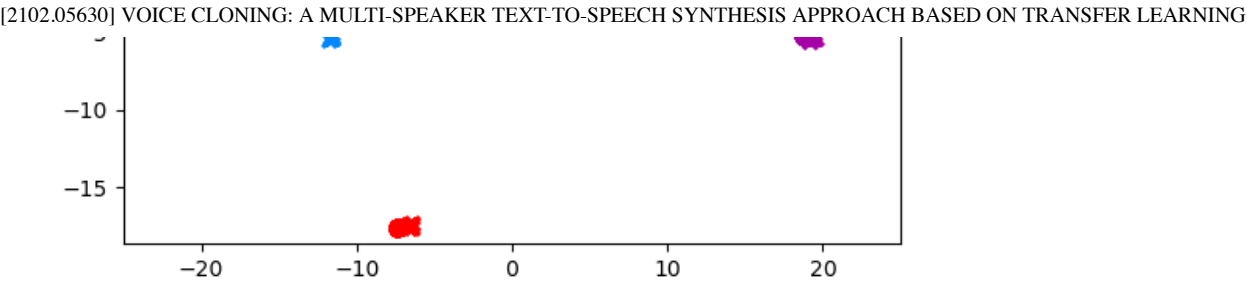


Fig. 5: Groundtruth utterance embeddings vs the corresponding generated ones of the 8 speakers chosen for testing.

3.4 Subjective Evaluation

Finally, we evaluated how the generated utterances were, subjectively speaking, similar in terms of speech timbre to the original ones. To do this, we gathered Mean Similarity Scores (MSS) based on a 5 points mean opinion score scale, where 1 stands for “very different” and 5 for “very similar”. Ten utterances of the proprietary female voice were cloned using both the proposed and the baseline system and then 12 subjects, most of them TTS experts, were asked to listen to the 20 samples, randomly mixed, and rate them. Participants were also provided with an original utterance as reference. The question asked was: “How do you rate the similarity of these samples with respect to the reference audio? Try to focus on vocal timbre and not on content, intonation or acoustic quality of the audio”. The results obtained are shown in Table 2. Although not conclusive, this experiment highlights a subjective evidence of the goodness of the proposed approach, despite the significant variance of both systems: this is largely due to the low number of test participants.

Table 2: MSS of the baseline and the proposed systems.

System	MSS
baseline	2.59 ± 1.03
proposed	3.17 ± 0.97

4 Conclusions

In this work, our goal was to build a Voice Cloning system which could generate natural speech for a variety of target speakers in a data efficient manner. Our system combines an independently trained speaker encoder network with a se-

quence-to-sequence with attention architecture and a neural vocoder model. Using a transfer learning technique from a speaker-discriminative encoder model based on utterance embeddings rather than speaker embeddings, the synthesizer and the vocoder are able to generate good quality speech also for speakers not observed before. Despite the experiments showed a reasonable similarity with real speech and improvements over the baseline, the proposed system does not fully reach human-level naturalness in contrast to the single speaker results from [1]. Additionally, the system is not able to reproduce the speaker prosody of the target audio. These are consequences of the additional difficulty of generating speech for a variety of speakers given significantly less data per speaker unlike when training a model on a single speaker.

5 Acknowledgements

The authors thank Roberto Esposito, Corentin Jemine, Quan Wang, Ignacio Lopez Moreno, Skjalg LepsÅy, Alessandro Garbo and JÃ¼rgen Van de Walle for their helpful discussions and feedback.

References

- [1] J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. Skerrv-Ryan, R. A. Saurous, Y. Agiomvrgiannakis, and Y. Wu, “Natural tts synthesis by conditioning wavenet on mel spectrogram predictions,” in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 4779–4783.
- [2] Aäron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew W. Senior, and Koray Kavukcuoglu, “Wavenet: A generative model for raw audio,” *CoRR*, vol. abs/1609.03499, 2016.
- [3] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio, “Neural machine translation by jointly learning to align and translate,” *CoRR*, vol. abs/1409.0473, 2015.
- [4]

- Andrew Gibiansky, Sercan Arik, Gregory Diamos, John Miller, Kainan Peng, Wei Ping, Jonathan Raiman, and Yanqi Zhou,
“Deep voice 2: Multi-speaker neural text-to-speech,”
in *Advances in Neural Information Processing Systems* 30, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., pp. 2962–2970. Curran Associates, Inc., 2017.
- [5] Wei Ping, Kainan Peng, Andrew Gibiansky, Sercan O. Arik, Ajay Kannan, Sharan Narang, Jonathan Raiman, and John Miller,
“Deep voice 3: 2000-speaker neural text-to-speech,”
in *International Conference on Learning Representations*, 2018.
- [6] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur,
“Librispeech: An asr corpus based on public domain audio books,”
in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 5206–5210.
- [7] Yaniv Taigman, Lior Wolf, Adam Polyak, and Eliya Nachmani,
“Voiceloop: Voice fitting and synthesis via a phonological loop,”
in *International Conference on Learning Representations*, 2018.
- [8] Eliya Nachmani, Adam Polyak, Yaniv Taigman, and Lior Wolf,
“Fitting new speakers based on a short untranscribed sample,”
CoRR, vol. abs/1802.06984, 2018.
- [9] Ye Jia, Yu Zhang, Ron J. Weiss, Quan Wang, Jonathan Shen, Fei Ren, Zhifeng Chen, Patrick Nguyen, Ruoming Pang, Ignacio Lopez-Moreno, and Yonghui Wu,

- “Transfer learning from speaker verification to multispeaker text-to-speech synthesis,”
CoRR, vol. abs/1806.04558, 2018.
- [10] Lipeng Wan, Qi shan Wang, Alan Papir, and Ignacio Lopez-Moreno,
“Generalized end-to-end loss for speaker verification,”
2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 4879–4883, 2018.
- [11] Georg Heigold, Ignacio Moreno, Samy Bengio, and Noam Shazeer,
“End-to-end text-dependent speaker verification,”
2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 5115–5119, 2016.
- [12] Ehsan Variani, Xin Lei, Erik McDermott, Ignacio Lopez Moreno, and Javier Gonzalez-Dominguez,
“Deep neural networks for small footprint text-dependent speaker verification,”
in *Proc. ICASSP*, 2014.
- [13] Serkan Kiranyaz, Onur Avci, Osama Abdeljaber, Turker Ince, Moncef Gabbouj, and Daniel J. Inman,
“1d convolutional neural networks and applications: A survey,”
ArXiv, vol. abs/1905.03554, 2019.
- [14] Junyoung Chung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio,
“Empirical evaluation of gated recurrent neural networks on sequence modeling,”
in *NIPS 2014 Workshop on Deep Learning, December 2014*, 2014.
- [15] Nal Kalchbrenner, Erich Elsen, Karen Simonyan, Seb Noury, Norman Casagrande, Edward Lockhart, Florian Stimberg, Aäron van den Oord, Sander Dieleman, and Koray Kavukcuoglu,
“Efficient neural audio synthesis,” in *ICML*, 2018.

- [16] Heiga Zen, Viet Dang, Rob Clark, Yu Zhang, Ron J. Weiss, Ye Jia, Zhifeng Chen, and Yonghui Wu, “Libritts: A corpus derived from librispeech for text-to-speech,” in *INTERSPEECH*, 2019.
- [17] Arsha Nagrani, Joon Son Chung, and Andrew Zisserman, “Voxceleb: A large-scale speaker identification dataset,” in *INTERSPEECH*, 2017.
- [18] Joon Son Chung, Arsha Nagrani, and Andrew Zisserman, “Voxceleb2: Deep speaker recognition,” in *INTERSPEECH*, 2018.
- [19] Diederik Kingma and Jimmy Ba, “Adam: A method for stochastic optimization,” *International Conference on Learning Representations*, 12 2014.
- [20] Katarzyna Janocha and Wojciech Czarnecki, “On loss functions for deep neural networks in classification,” *ArXiv*, vol. abs/1702.05659, 2017.
- [21] Corentin Jemine, “Master thesis: Automatic multispeaker voice cloning,” 2019, Unpublished master’s thesis, Université de Liège, Liège, Belgique.
- [22] Klaus Greff, Rupesh K. Srivastava, Jan Koutnik, Bas R. Steunebrink, and Jurgen Schmidhuber, “Lstm: A search space odyssey,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 10, pp. 2222–2232, Oct 2017.
- [23] Leland McInnes and John Healy, “Umap: Uniform manifold approximation and projection for dimension reduction,” *ArXiv*, vol. abs/1802.03426, 2018.

[24]

Christophe Veaux, Junichi Yamagishi, and Kirsten

MacDonald,

“Cstr vctk corpus: English multi-speaker corpus for cstr
voice cloning toolkit,” 2018.

[Feeling
lucky?](#)[Conversion
report \(OK\)](#)[Report
an issue](#)[View original
on arXiv](#)[Copyright](#)[Privacy Policy](#)[Generated on Mon Jan 3 15:30:38 2022 by LaTeXML !\[\]\(2b17f17ebbacc911bb0ff784ab641779_img.jpg\)](#)