**SPEECH SYNTHESIS USING DEEP LEARNING**

**S. B. Shah, A. Karn & S. K. Kushwaha**

**Computer Science and Engineering Department**

**Delhi Technological University**

1. **Abstract**

Speech synthesis is the production of speech artificially. The art of speech synthesis is applicable in many practical fields such as accessibility for visually impaired people, virtual assistants, music generation, and many more.

**Deep Learning** is a subdivision of machine learning that consists of algorithms inspired by the structure and function of the human brain called neural networks. Deep Learning has been said to make revolutionary advances in the machine learning and artificial intelligence field.

In this project, a **speech synthesizer using Deep Neural Networks** is implemented and studied. The program would extract some characteristics from a few samples of the speaker's voice, clone it and generate audio based on the given text. The performance of the program is determined by the naturalness of speech and the similarity between the speaker’s speech and the recreated speech.

1. **Introduction**

Deep learning is a subset of Machine Learning which aims to imitate the functioning of the human brain and learn from large amounts of data. Then, the same task is repeatedly performed so that the model learns from experience and improves its performance in subsequent attempts.

Deep learning is very popular in many sub-disciplines of machine learning. It is predominant in text-to-speech, which is the assistive technology that converts text to spoken word. A lot of research has been done to make these deep learning models more efficient and generate more natural-sounding speech. Some models even produce artificial speech that is almost indiscernible from human speech. Interestingly, since speech naturalness is judged based on subjective metrics, artificial speech that is “more natural than human speech” might be possible to produce. In fact, some even argue that it has already been achieved.

Speech generation and text-to-speech have a wide variety of applications and become increasingly popular every day. Properly and professionally recorded speech with correct pronunciation and sufficient variations in pitch would help the model produce more accurate speech with minimal background noise. However, not many datasets with properly recorded speech for models to use as input data exist currently. Training a relatively basic text-to-speech model such as Tacotron would require hundreds of hours of sample input speech.

Voice conversion is the process of converting a speech segment from one voice to another. On the other hand, voice cloning is the process of capturing a speaker’s voice and processing speed. To determine the similarity of generated voice, input reference samples needed can range from half an hour-long to only a few seconds long.

This study aims to work towards an efficient and potent voice cloning model. It should be able to clone voices with only a few seconds of reference input audio. The work of (J. Corentin et al, 2018) will be implemented along with a custom-made dataset.

1. **Related work**

Deep models that produce natural-sounding speech began appearing in 2016. Since then, a lot of research has been done in the field to make them better across many characteristics like efficiency, naturalness, etc.

This study was mainly based on the following paper:

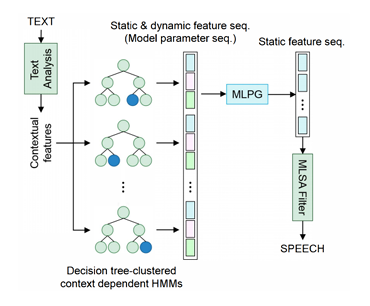
* **J. Corentin: Automatic Multispeaker Voice Cloning, 2019.**

Some of the other related works include:

* Sercan Arik, Gregory Diamos, Andrew Gibiansky, John Miller, Kainan Peng, Wei Ping, Jonathan Raiman, and Yanqi Zhou. Deep voice 2: Multi-speaker neural text-to-speech, 2017.
* J. S. Chung, A. Nagrani, and A. Zisserman. Voxceleb2: Deep speaker recognition. In INTERSPEECH, 2018.

1. **Research Methods**

Hidden Markov Model (HMM) based framework approach consists of clustering the linguistic features extracted from the input text with a decision tree and training an HMM per cluster. The HMMs are used to basically establish a distribution over spectrogram coefficients, their first and second derivatives, and a binary flag that has information about the voice-containing parts of the audio. Spectrogram coefficients are sampled from the distribution and fed to the MLPG vocoder using Maximum Likelihood Parameter Generation. The voice generated can be changed by training the HMMs on a speaker’s audio or tuning audio with interpolation techniques.

*Fig: J. Corentin: Automatic Multispeaker Voice Cloning, 2019, HMM-based TTS pipeline*

Improvements to this framework were later brought by feed-forward Deep Neural Networks (**DNN-2013**) which has better data efficiency as the training set is no longer fragmented in different clusters of contexts.

**RNNs(2014)** make natural acoustic models as they can learn a compact representation of complex and long-span functions. As RNNs (Recurrent Neural Network) are fit to generate temporally consistent series, the acoustic model can directly determine the static features, alleviating the need for dynamic features and MLPG.

**WaveNet(2016)** is a deep convolutional neural network that, for a raw audio waveform, models the distribution of a single sample conditionally to previous ones, which made a substantial breakthrough in TTS. By using this, audio can be generated by using an autoregressive manner while predicting samples. Wavenet holds many stacks of one-dimensional convolutions. These convolutions are dilated and have a dilation factor that increases exponentially with layer depth. This allows the strong nonlinearity and receptive field that is essential to model audio.

**Tacotron(2017)** is a sequence-to-sequence model that produces a spectrogram from a sequence of characters alone, further reducing the need for domain expertise.

1. **Analysis and Implementation**

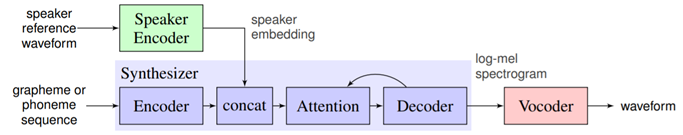
**5.1. Analysis**

The framework includes three stages as follows:

1. A speaker **encoder** that constructs an embedding from a short audio sample of a person. The embedding is a meaningful representation of the speaker's voice, so that similar voices in latent space are close together.

2. A **synthesizer** that, when conditioned on the embedding of a speaker, generates a spectrogram from the text. This model is the popular Tacotron 2 (Shen et al., 2017) real-time without WaveNet.

3. A **vocoder** that generates an audio waveform based on the spectrograms generated by the synthesizer.

*****Fig: The SV2TTS framework during inference. High-level view of the Tacotron architecture is represented by the blue blocks.*

For the datasets of the synthesizer and vocoder, transcripts are required and the quality of the generated audio can only be as good as that of the data. A large collection of many different speakers would be preferable to train the encoder, without any strong requirement on the noise level of the audios.

**5.1.1. Encoder**

This encoder is implemented using a Python library called PyTorch. The model is a 3-layer **Long Short Term Memory(LSTM)** with 768 hidden nodes followed by a projection layer of 256 units. This is a relatively small model but performs well. Before normalization, it also features a **Rectified Layer Unit(ReLU)** layer, making embeddings sparse and more interpretable.

The encoder is trained for a speaker verification task. A template for a person is created by deriving their speaker embedding, which is known as enrollment. The user then uses a short phrase to identify himself at runtime, and the system compares the embedding to the enrolled speaker embeddings. When similarity is higher than a given threshold, a user is distinguished.

The model computes the embeddings eij (1 ≤ i ≤ N, 1 ≤ j ≤ M ) of N speakers with M audio samples at training time. A speaker embedding ci is derived for each speaker:

The comparison of all embeddings eij against every speaker embedding ck (1 ≤ k ≤ N) in the batch results in a similarity matrix Sij,k against every speaker embedding ck (1 ≤ k ≤ N).

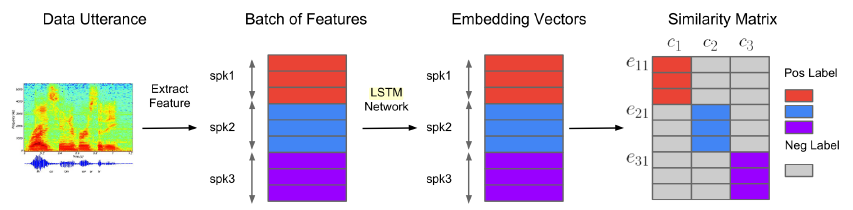


This measure is the scaled cosine similarity:

**

where w and b are learnable parameters.

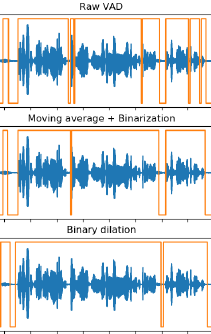
The cosine similarity of two vectors is simply their dot product from a computing perspective. Hence the rightmost hand side of equation 2. High similarity values are expected when the speakers are different, and low similarity values are expected when the speakers are different. The loss is the sum of row-wise softmax losses for optimization.

*Fig. J. Corentin: Automatic Multispeaker Voice Cloning, 2019, The construction of the similarity matrix at training time. (Wan et al., 2017).*

It should be noted that the centroid ci of the same speaker includes each utterance eij when computing the loss. A bias towards the speaker is created independently of the accuracy of the model, and the authors of SV2TTS say that this allows some room for trivial solutions. An utterance that is matched with its own speaker's embedding will be removed to prevent this .

When enrolling a speaker in a practical application, several utterances should be loaded from each user but not more than 10. As for the number of speakers, this number should not be too large since the complexity of the similarity matrix is O(N2M ). If this number is too high, the training time would become very large.

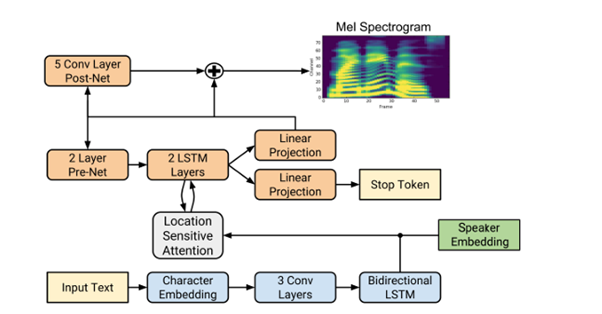
Voice Activity Detection(VAD) is done by using webrtcvad python module to avoid segments of audio that are silent when sampling partial and complete utterances. The audio is trimmed of the silent parts which exceed a maximum threshold of 0.2 seconds.



*Fig. J. Corentin: Automatic Multispeaker Voice Cloning, 2019, Silence is removed from top to bottom using VAD(Voice Automation Detection). The binary voice flag is represented by the orange line. Upper lines signify the parts with audio, while lower ones signify the quiet parts of audio.*

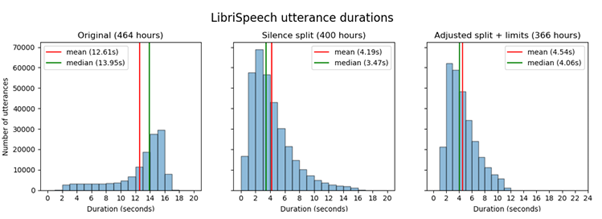
To train this model, the LibriSpeech dataset was used which contains two sections- “clean” and “other” where the clean sample section contains relatively reduced noise. According to (Wan et al., 2017), the number of speakers is strongly correlated with the good performance of the encoder as well as the entire framework. So, the speaker encoder should be trained for as many steps as feasible.

**5.1.2 Synthesizer**

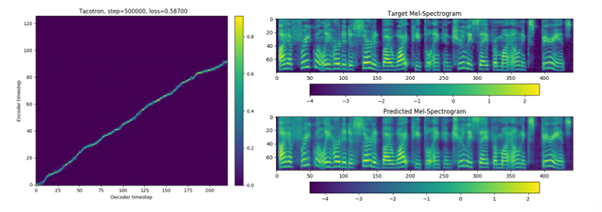
**Tacotron 2** is an open-source TensorFlow implementation of Tacotron 2 that has been stripped of Wavenet and SV2TTS's modifications. Tacotron is a recurrent sequence-to-sequence method that incorporates text to predict a mel spectrogram. It uses a location sensitive attention mechanism for its encoder-decoder system .The text sequence's characters are first incorporated as vectors. Convolutional layers are added afterward to increase the range of a single encoder frame. To produce the encoder output frames, these frames are passed through a bidirectional LSTM. This is where SV2TTS makes changes to the architecture: every frame emitted by the Tacotron encoder is concatenated with a speaker embedding. Previous decoder frames are concatenated onto the next decoder input frame when passed through a pre-net. The concatenated vector is projected into a single mel spectrogram frame after being passed through two LSTM layers. A residual post-net converts the entire sequence of frames into a mel spectrogram. *Fig: J. Corentin: Automatic Multispeaker Voice Cloning, 2019, The modified Tacotron architecture. The orange blocks represent the decoder, and the encoder is represented by the blue blocks.*

The synthesizer's target mel spectrograms have more characteristics than the speaker encoder's target mel spectrograms. They are made up of 80 channels and are based on a 50ms window with a 12.5ms step. In this approach, the input texts are not pronounced, and the characters are supplied as is. However, there are a few cleaning procedures: replacing abbreviations and numerals with their full textual form, forcing all characters to ASCII, normalizing whitespaces, and changing all characters to lowercase. Punctuation may be utilized, but it wasn't available in this database.

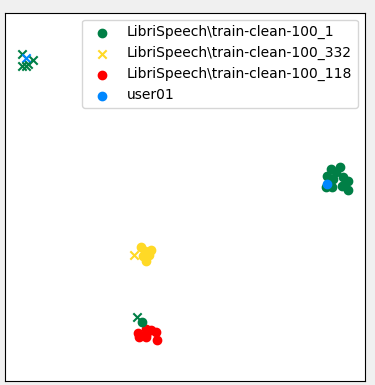
LibriSpeech was used because it offered the best voice cloning similarity on unseen speakers. Utterances on silences longer than 0.4 seconds were split with the audio aligned to the text. This aided the synthesizer's convergence, both because silences were removed from the target spectrogram and because the median time of the utterances in the dataset was reduced, as shorter sequences provide less room for timing errors. It was ensured that utterances are not shorter than 1.6 seconds, the duration of partial utterances used for training the encoder, and not longer than 11.25 seconds.

*Fig: J. Corentin: Automatic Multispeaker Voice Cloning, 2019, Distribution of length of utterances in the dataset*

Ground Truth Aligned (GTA) mode (also known as a teacher-forcing mode), where preceding frame of spectrogram is fed to the pre-net, is used to train the model. It is difficult to provide any quantitative assessment of the performance of the model.

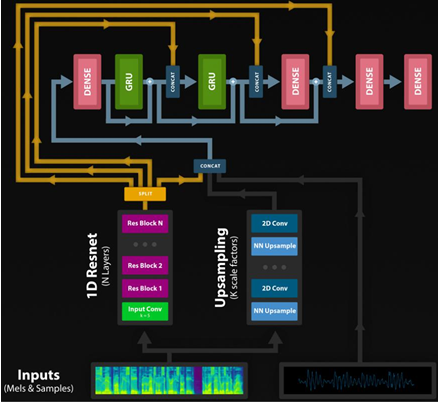
*Fig: J. Corentin: Automatic Multispeaker Voice Cloning, 2019, Tacotron producing correct outputs through informal listening tests*

The synthesizer is trained in several characteristics using Griffin-Lim (Griffin and Jae Lim, 1984) before training the vocoder. Griffin-Lim is an iterative approach that guesses the source audio signal of a spectrogram, not a machine learning model. The audio produced in this manner often retains little of the speaker's vocal qualities, but the speech is understandable. Even in the presence of complicated or convoluted terms, the synthesizer's speech accurately fits the text. However, the speech is occasionally strange, with pauses in unexpected places in the phrase or no pauses when they are anticipated. The limits that were imposed on the duration of utterances in the dataset (1.6s -11.25s) are likely also problematic.

By computing the embeddings of synthetic speech and projecting them using UMAP alongside ground truth embeddings, it is clear that some voice characteristics are lost with Griffin-Lim. The figure below shows an illustration. The synthesized embeddings clusters are similar to their respective ground truth embeddings clusters, as seen. The loss of emergent traits is also obvious; for example, the synthesized utterances for the pink, red, and two blue speakers show lower inter-cluster variance than their ground truth counterparts. The grey and purple speakers are also affected by this problem. Tacotron is frequently quicker than real-time. **F*ig: J. Corentin: Automatic Multispeaker Voice Cloning, 2019, UMAP projections of various speakers*

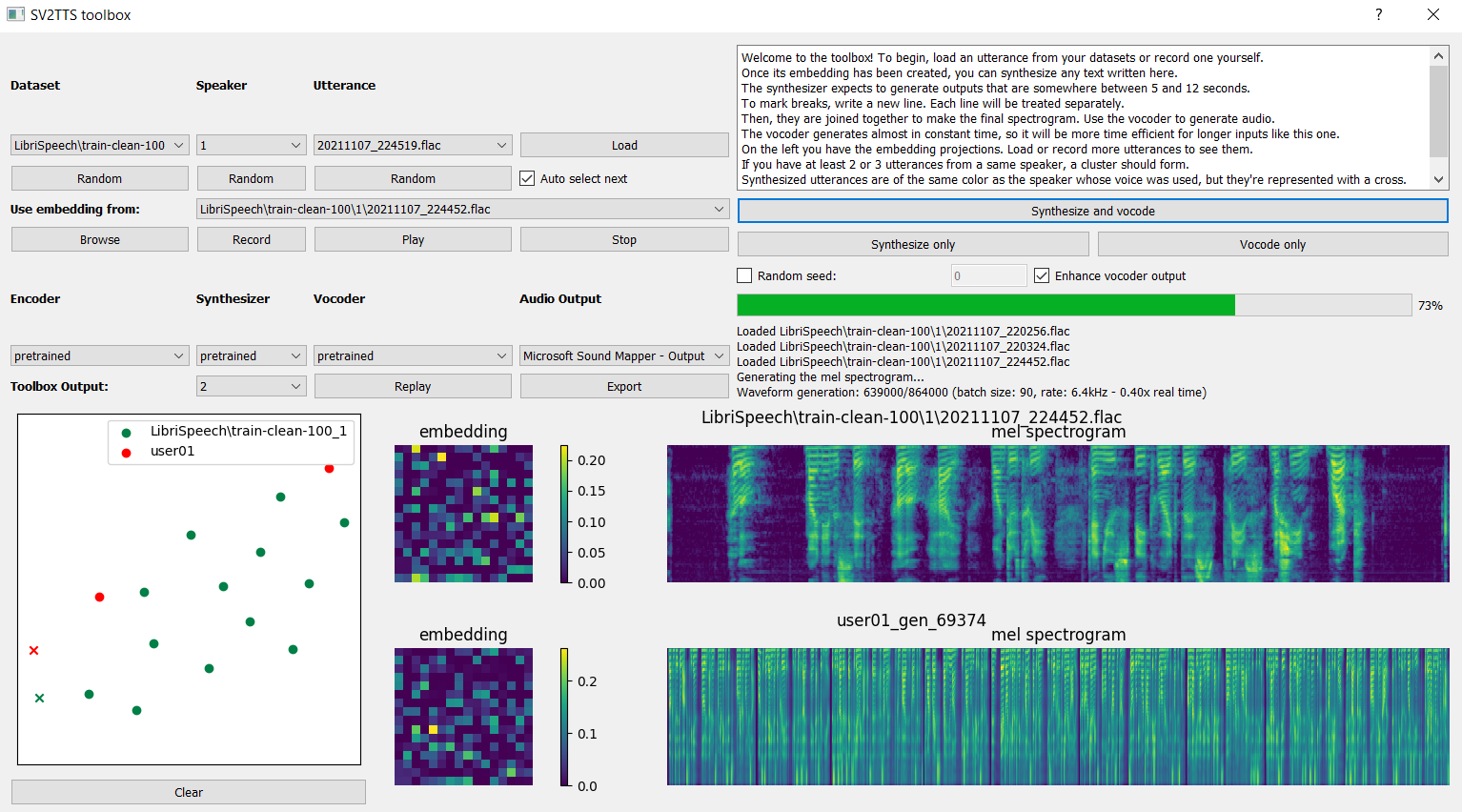
**5.1.3 Vocoder**

WaveNet has been at the heart of deep learning with audio since its release and remains state of the art when it comes to voice naturalness. It is known for being the slowest practical deep learning architecture at inference time. 8000 samples per second are generated by the Wavenet implemented by Google(Kalchbrenner et al., 2018, page 2). However, “vanilla” Wavenet only generates 172 samples per second. The difference is amazingly high. A mel spectrogram has the same number of segments as its related waveform. A Resnet-like model uses the spectrogram as input to generate features that will condition the layers throughout the transformation of the mel spectrogram to a waveform. Finally, two dense layers produce a distribution over discrete values that correspond to a 9-bit encoding of mu-law compounded audio.

*Fig: : J. Corentin: Automatic Multispeaker Voice Cloning, 2019, The alternative WaveRNN architecture.*

The vocoder frequently runs slower than real-time when dealing with short utterances. The number of folds in batched sampling has a significant impact on inference speed. On this setup, this threshold is 12.5 seconds; for utterances shorter than this threshold, the model will run slower than real-time on PyTorch. The authors of (Kalchbrenner et al. 2018) maintain that a small, dense WaveRNN will perform slower and less effectively than a a large, sparse one. Thus, can be inferred that a pruned vocoder would produce better results in terms of speed.

**5.2 Implementation**

****

*Fig. Framework Application*

This framework is implemented in an application with a simple user interface that helps users get a quick overview of every process going on currently in the system.

The user is allowed to record an utterance or select any utterance from a dataset in their disk. After this is done, the program computes its embedding and draws its corresponding spectrogram and heatmap. However, since embeddings are unidimensional vectors, the square shape has no structural meaning about the embedding values.

When the embedding has been computed, the selected voice is used to convert the text given by the user to speech. A complete spectrogram of the output is displayed after synthesis is completed. The embedding of synthesized utterance is also generated and is projected in the UMAP as well.

1. **Result and Discussion**

In this implementation, the voice recognition section works almost flawlessly whereas the speech synthesis works adequately well. Speech similarity is quite high, whereas speech naturalness can be improved. Overall, the results were very satisfactory and above expectations.

Since this framework was run on a computer with a relatively weak GPU along with time and computational constraints, it is highly possible that the results did not achieve their full potential. The model was trained for a million steps, whereas the original author trained it for 50 million steps and achieved slightly better results. With an even more powerful computer, the results would likely have been even closer to human levels by achieving even greater naturalness and similarity.

1. **Conclusion and Future Research Directions**

A framework for real-time voice cloning was studied and implemented. The results exceeded expectations but could be improved with more reference speech time.

Proper datasets for audio-based projects such as this are relatively scarce. The SV2TTS team has been working on a vocoder that could clone most voices, but not uncommon ones. In the future, new frameworks could be developed that account for this overlooked aspect as well.

The importance of research in text-to-speech increases day by day since many leading technology companies aim to adopt their own artificially intelligent chatbot or personal assistants. Developments in this field are being made all the time and future implementations could provide even better results.

1. **References**

Corentin Jemine. Real-time Voice Cloning, 2018 <https://matheo.uliege.be/bitstream/2268.2/6801/5/s123578Jemine2019.pdf>

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation

by jointly learning to align and translate. CoRR, abs/1409.0473, 2014. [http://arxiv.org/abs/1409.0473.](https://arxiv.org/pdf/1409.0473)

Jan Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, KyungHyun Cho, and Yoshua

Bengio. Attention-based models for speech recognition. CoRR, abs/1506.07503,

2015. <http://arxiv.org/abs/1506.07503.>

Sercan Arik, Gregory Diamos, Andrew Gibiansky, John Miller, Kainan Peng, Wei

Ping, Jonathan Raiman, and Yanqi Zhou. Deep voice 2: Multi-speaker neural text-to-

speech, 2017.

Sercan O. Arik, Jitong Chen, Kainan Peng, Wei Ping, and Yanqi Zhou. Neural voice

cloning with a few samples, 2018.

J. S. Chung, A. Nagrani, and A. Zisserman. Voxceleb2: Deep speaker recognition. In

INTERSPEECH, 2018.

D. Grin and Jae Lim. Signal estimation from modified short-time FourierSometransform.

IEEE Transactions on Acoustics, Speech, and Signal Processing, 32(2):236-243, April 1984. ISSN 0096-3518. doi: 10.1109/TASSP.1984.1164317.

Ze, Heiga, Andrew Senior, and Mike Schuster. "Statistical parametric speech synthesis using deep neural networks." 2013 IEEE international conference on acoustics, speech and signal processing. IEEE, 2013.

Ning, Yishuang, et al. "A review of deep learning based speech synthesis." Applied Sciences 9.19 (2019): 4050.

Tokuda, Keiichi, et al. "Speech synthesis based on hidden Markov models." Proceedings of the IEEE 101.5 (2013): 1234-1252.

Zen, Heiga, et al. "A hidden semi-Markov model-based speech synthesis system." IEICE transactions on information and systems 90.5 (2007): 825-834.