UNIVERSITY OF TARTU
Institute of Computer Science
Computer Science Curriculum

Hain Zuppur

Creating a Voice Conversion Model for Estonian

Bachelor's Thesis (9 ECTS)

Supervisor: Liisa Rätsep

Tartu 2021

Abstract

Voice conversion has a variety of uses, like enhancement of impaired speech or entertainment purposes. The main challenge in voice conversion is extracting speaker-independent linguistic features from speech. To date, one of the most promising solutions is the Cotatron model. Estonian has some high-quality speech synthesis models, but there are no voice conversion models for Estonian. This thesis aims to take the Cotatron model and train it using Estonian Text-to-Speech datasets to produce a voice conversion model for the Estonian language.

Keywords: neural networks, voice conversion, speech technology, Cotatron, speaker imitation.

CERCS: P176 Artificial intelligenc

Kokkuvõte

Hääle teisendamisel on palju erinevaid kasutusi: abistavas tehnoloogias, inimese ja arvuti interaktsioonis ja meelelahutuses. Peamine probleem ühe inimese kõne teise inimese kõneks muutmisel on rääkijast sõltumatute keeleliste tunnuste eraldamine kõnest. Parimat tulemust on hetkel näidanud hääleteisenduse mudel Cotatron. Eesti keele jaoks on loodud häid kõnesünteesi mudeleid, aga pole ühtegi hääle teisendamise mudelit. Selle töö eesmärk on treenida Cotatroni mudelit kasutades eestikeelseid kõnesünteesi andmestikke, et luua häälteisenduse mudel eesti keele jaoks.

Võtmesõnad: tehisnärvivõrgud, hääleteisendus, kõne tehnoloogia, Cotatron, kõneleja imiteerimine.

CERCS: P176 Tehisintellekt

Contents

1. Introduction 4
2. Background 5
2.1 Artificial neural networks 5
2.2 Voice conversion 6
2.3 Spectrograms 7
2.4 Vocoding 9
2.5 Tacotron 2 and Multi-speaker Tacotron 2 10
2.6 Cotatron 11
2.7 Blow 13
2.8 AutoVC 14
2.9 FragmentVC 15
3. Experiments 16
3.1 Choosing the model 16
3.2 Implementation Details 17
3.3 Dataset 18
3.4 Training 19
4. Results 20
4.1 Survey setup 20
4.2 Survey results 21
4.3 Results compared to the original English model 23
5. Conclusion 24
6. References 25
Appendix 28
I. Estonian Cotatron implementation and audio samples 28
II. Estonian Cotatron audio samples 28
Licence 29

1. Introduction

There are many uses for voice conversion: assistive technology, human-computer interaction, and entertainment. The most advanced speech conversion models are based on deep neural networks [1]. Voice conversion is the conversion of one person's voice into another person's voice. Developments in voice conversion make it increasingly possible to use this technology in many different applications where it has not been previously possible to use voice conversion due to errors or artificial voice. Voice conversion has uses from entertainment to personalized communication aids for the speech impaired. To turn one person's voice into another person's voice, the model must encode speaker-independent features of a given sound clip and synthesize sound with the target speaker's voice representation [1]. One of the best models is the Cotatron, which uses textual transcription in addition to speech. As a result, Cotatron is better able to distinguish speech-independent features from speech, and synthesized speech is more natural and more similar to the voice of the target speaker than previous methods [1]. Cotatron does not synthesize speech but creates a mel spectrogram from which sound is created using MelGAN, which converts the mel spectrogram into an audio file.

The aim of this thesis was to give an overview of exciting voice conversion models, explain how they work, and retrain one of them using an Estonian dataset to produce a voice conversion model for the Estonian language. Additionally, the thesis aims to evaluate the performance of the Estonian voice conversion model.

2. Background

This chapter gives an overview of artificial neural networks, Cotatron, Tacotron 2, multi-speaker Tacotron 2, and MelGAN. These are the technologies the Cotatron is based on. Cotatron architecture is based mainly on multi-speaker Tacotron 2, and to convert the generated mel spectrogram to audio, it uses MelGAN. Also, we give a short overview of other voice conversion models: Blow, autoVC, and FragmentVC.

2.1 Artificial neural networks

Sarat Kumar Sarvepalli [2] has written that artificial neural networks are computing systems vaguely inspired by the biological neural networks that constitute animal brains. The artificial neural networks consist of connected neurons that compute an output value using an objective function from their input values. What makes artificial neural networks unique is that their creator does not set the parameters inside the network; instead, the neural network is trained using thousands or even millions of examples, and the network changes its parameters to perform its task better. During the training process, each time the neural network makes a wrong choice, a feedback mechanism finds what neuron made a mistake and tunes its parameters to make the neuron produce the desired output. Sarvepalli describes that by doing that for a sufficient amount of time and with many training samples, the model learns to generalize and pick up hidden patterns unknown to humans.

A typical architecture used for natural language processing tasks, like machine translation and voice synthesis, is Sequence-to-sequence Recurrent neural network. The architecture is based on [3]; this network is also called encoder-decoder; the architecture is shown in Figure *1*. This type of network takes an input sequence and transforms it into an output sequence. Ilya Sutskever *et al*. [3] found that this kind of sequence to sequence model used for machine translation outperforms other SMT-based systems. The sequence-to-sequence models consist of two parts, an encoder and a decoder. The encoder takes the input sequence and outputs a context embedding. After that, the decoder takes the context embedding and starts to generate the output element by element. Every element is generated by taking the highest probability prediction from previous predictions and the context embedding. During training, if the prediction of one of the elements is wrong, then the subsequent predictions will also be wrong, and this causes a compounding of errors. This causes the training process to be longer and harder. One standard solution for this problem is giving the ground truth instead of the previous

predictions, which means the wrong predictions from previous elements would not affect the current elements prediction. This kind of method is called teacher forcing. [4]

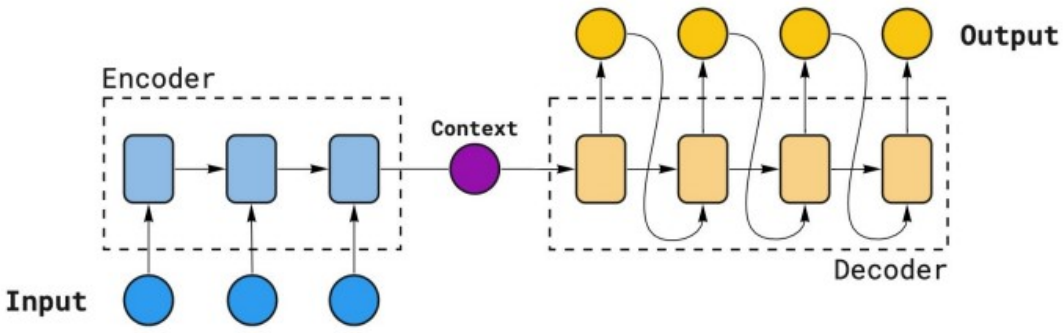

Figure 1. Sequence-to-sequence architecture [4].

While the Sequence-to-sequence works better than other recurrent neural networks, they still struggle whit longer inputs because the context embedding becomes large.

Dzmitry Bahdanau *et al*. [5] Found that using attention alongside context embedding works much better on longer sequences. In models that use attention, the encoders output the context embedding and an encoding sequence that is the same length as the input sequence. Then at each decoding timestep, an alignment is calculated between the current decoder input and the input sequence. The alignment shows how relevant each element of the input sequence is to the current output element. Then the alignment is used to calculate timestep-specific context embedding. This kind of method helps the decoder focus more on the relevant parts of the context embedding for the current timestep, thus allowing the context embedding to be longer while not losing accuracy.

2.2 Voice conversion

Berrak Sisman *et al.* [6] explain that voice conversion is a technology of modifying a speaker's voice, so it sounds like someone else is speaking while preserving the linguistic information (contents of speech and intonation). Speech consists of three factors: linguistic factors, supra-segmental factors, and segmental factors. Linguistic factors are the information given in the speech, the sentence structure, word choices, and individual's distinctive and unique use of language. The supra-segmental factors also called prosodic characteristics of a speech, describe the stress and tone of the speech. The segmental factors determine the fundamental frequencies

of vowels and syllables. Berrak Sisman *et al.* [6] describe that we keep the linguistic features the same but change the supra-segmental factors and segmental factors to the target speakers for voice conversion.

In their overview of voice conversion, Berrak Sisman *et al.* [6] says that older voice conversion systems used statistical modeling for conversion. However, they have a significant disadvantage; they require parallel data. In a parallel dataset, there are the same sentences pronounced by multiple people. These kinds of datasets are hard to obtain because we need to record them specially and can not use existing data. Modern models do not use parallel data, so datasets can be easily obtained because we can use exciting data like news readings.

Modern approaches to voice conversion use neutral networks; all of the different models described in this thesis are neural networks-based. Like in many other fields, deep learning techniques have provided much better results in voice conversion than other approaches. In Voice Conversion Challenge 2020 [7], all of the submitted models were deep learning based.

2.3 Spectrograms

Modeling raw audio is complex; in his thesis, Oleh Matsuk [4] explained that typical audio recording used for voice conversion is sampled at 22 050 Hz, which means one-second-long audio recording contains 22 050 discrete samples, which means typical senates will be over 100 000 samples. That is why many models use spectrograms as a more compact audio representation.

Raw audio consists of a complicated waveform; by looking at the waveform, it is hard to understand what info it contains. That is why signal processing spectrograms are often used. Spectrograms are a representation of audio where by using some function, it is calculated which frequencies the waveform consists of. When making the spectrogram, the audio is split into frames of a specified length. In Figure *2*, there is audio as a waveform and a spectrogram. On both graphs, the x-axes represent time, but the y-axis is different for both of them. In the waveform, y-axes represent amplitude, and in the spectrogram, it represents frequencies. On the spectrogram, the lighter the cell is, the more of that frequency is in that spectrogram frame.

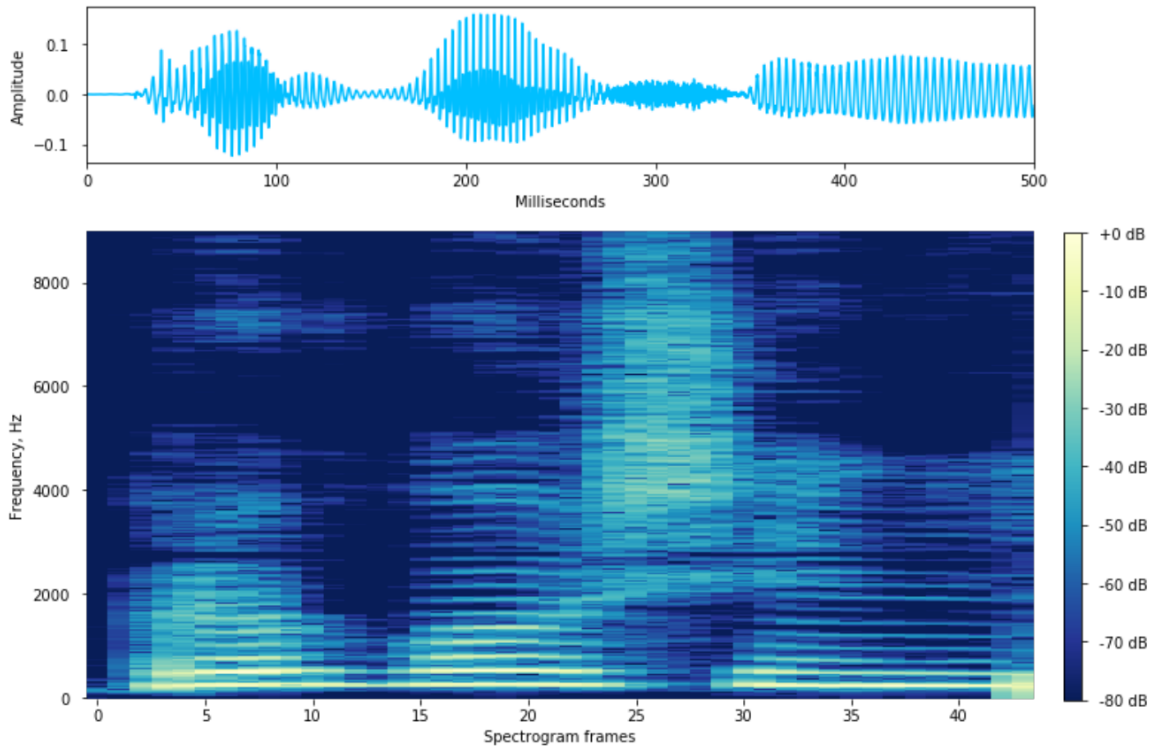

Figure 2. Spectrogram decomposition of a waveform [4].

Matsuk [4] describes that humans perceive changes in frequency more at lower frequencies than higher frequencies. This means human hearing is logarithmic. Spectrograms are often in mel scale to capture this aspect of human hearing. The frequencies in the spectrogram have been transformed to match the sensitivity of human hearing.

Matsuk [4] explains that many neural network-based models like The Tacotron 2 generate a mel spectrogram instead of raw audio. As mentioned above, generating raw audio is very difficult for neural-network-based models because of the high resolution. The first step in turning raw audio into a mel spectrogram is to split the audio into slices using some window function (Hann window function is often used). Then, each slice is performed a short-time Fourier transform operation (STFT), which finds what fundamental frequencies and how much the current slice consists of. By doing that, we obtain a spectrogram, and then we need to convert it to *mel scale*. Most voice synthesis and conversion models use audio files sampled at 22 050 Hz, so when we use mel spectrogram, with 1024 samples per frame with a hop length of 256, then we will have 86 spectrogram frames per second. If we use an 80-channel mel scale,

then when we can convert a one-second audio clip (sequence of 22 050 elements) into a mel spectrogram, that is an 86×80 spectrogram matrix. Matsuk explains that by doing this, we achieve a data compression factor of around 3.2 and reduce the sequence length by a factor of 256.

2.4 Vocoding

Constructing raw audio from mel spectrograms is difficult; the spectrogram has much less information in it than raw audio. There are different ways of converting mel spectrograms into audio. Traditionally for that task have been used algorithm-based systems like Griffin-Lim Algorithm (GLA) [8]. Wang *et al*. [9] have described algorithm-based systems are not good for filling the gaps of information in the spectrogram because using those methods introduces distinct robotic artefacts in the converted audio. In recent years, neural network-based approaches have become more popular because, with sufficient training data, those models can convert the mel spectrogram into raw audio much better without introducing artefacts.

Previous state-of-the-art models like WaveNet [10] are autoregressive neural-network-based, which means future values are predicted based on past values. These models produce much better results in converting mel spectrograms into raw audio than algorithm-based models like Griffin-Lim Algorithm (GLA) [8]. Kundan Kumar *et al.* [11] writes that the main drawback of these types of models is that they are autoregressive. For longer mel spectrograms, they are quite slow because the different timesteps cannot be run in parallel, and because of this, they are not suitable for real-time use.

In recent years, non-autoregressive models have become more popular, like Parallel WaveNet [12], because the different time steps can be run in parallel, making these models much faster. Aaron van den Oord *et al*. [12] found that the Parallel WaveNet produces the same quality conversion as WaveNet [10]. Kundan Kuma *et al*. [11] describe that because these models can run in parallel, they can better take advantage of modern hardware, which provides a significant speed increase. The MelGAN [11] authors also point out the main drawback of these kinds of systems; they take up lots of memory, and because of that, pose some hardware limitations on their uses.

MelGAN [11] authors describe it as a non-autoregressive feed-forward convolutional architecture to perform audio waveform generation in a generative adversarial network (GAN) setup. MelGAN converts mel spectrogram to raw audio and shows promising results when used for a text-to-speech synthesis model. MelGAN is a Non-autoregressive model, which means it is highly parallelizable and Can take advantage of modern hardware used for deep learning. Authors suggest that because of this, it is much faster than autoregressive neural-networks-based models like WaveNet.

2.5 Tacotron 2 and Multi-speaker Tacotron 2

According to the Tacotron 2 [13] research paper, it is a neural network-based text-to-speech synthesis model. It is based on a recurrent sequence-to-sequence feature prediction network that maps character embedding's to mel Spectrograms. Using the WaveNet model it turns the generated mel spectrograms into raw audio.

The multispeaker Tacotron 2 [14] builds upon the Tacotron 2 [13]. While the Tacotron 2 could only synthesize one speaker's voice, the Multi-speaker Tacotron 2 can synthesize multiple voices. The authors [14] describe the model architecture as the basic Tacotron 2 model combined with an additional speaker encoder. The speaker encoder is trained separately from the Tacotron 2. The architecture of the multi-speaker Tacotron 2 is shown in Figure *3*.

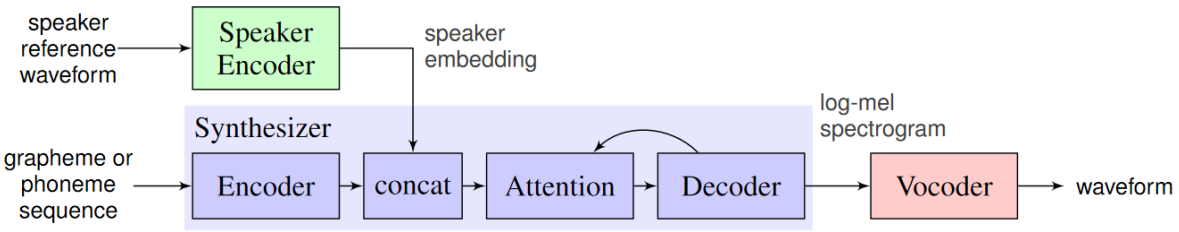

Figure 3. Multi-speaker Tacotron 2 model overview [14].

Matsuk [4] describes the main difference between the text-to-speech model and the multi-speaker text-to-speech model is control over speaker identity (voice of the speaker) and prosody (patterns of stress and intonation). By controlling the speaker's identity and prosody, we can change the voice of the text-to-speech model.

Ye Jia *et al.* [14] explain that the speaker encoder is used to generate a representation of the target speaker that captures the characteristics of different speakers. The speaker encoder maps the target speaker's mel spectrogram frames to a fixed-dimensional embedding vector. The authors explain that this network is trained using generalized end-to-end speaker verification loss (GE2E). The same speaker utterances have high cosine similarity, and different speaker utterances have low cosine similarity.

The Multi-speaker Tacotron 2 Authors [14] describe that the synthesizer is also modified to include the target speaker embedding. As shown in Figure *3*, this embedding is used when constructing the converted mel spectrogram. The target speaker embedding contains information about the target speaker's voice; this embedding is combined in the synthesizer with the synthesizer's encoder's output.

2.6 Cotatron

According to the Cotatron [1] research paper, it is a speech encoder that converts one speaker's voice to another. Cotatron is trained with a text-to-speech dataset. The voice conversion system is used to reconstruct converted speech with features extracted by Cotatron from the source speaker and the target speaker embedding. According to the source, Cotatron architecture is based upon Multi-speaker Tacotron 2.

Cotatron [1] authors describe how to convert source speakers' voice recording to sound like target speaker voice is shown in Figure *4*; the process goes as follows: the source audio file is converted into a mel spectrogram. The mel spectrogram and its transcription are fed into Cotatron to extract the Cotatron feature *L*. The transcription is used to help Cotatron to extract speaker-independent linguistic features from source speaker audio because the transcription contains no information about the speaker's voice. This feature *L* contains information about

the rhythm of the speech and what was said without the information about the source speaker's voice. The Authors suggest that the Cotatron feature alone is not enough to reconstruct the audio because the Cotatron feature does not capture all of the required features, like intonation.

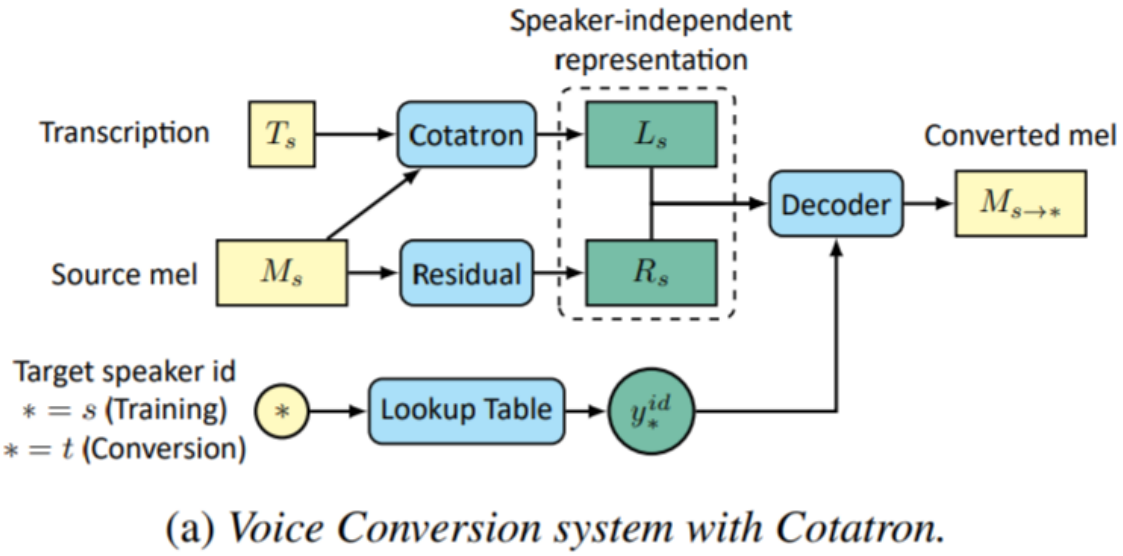

Figure 4. Cotatron model architecture. L is cotatron feature, and R is a residual feature [1].

Authors [1] say that to aid the decoder with reconstruction, the mel spectrogram of the source audio is also given to the residual encoder to get the residual feature *R*. The residual feature *R* is obtained by passing the frames of the mel spectrogram through 6 layered convolutional neural networks that output is a single channel that is instance normalized to avoid speaker-dependent information ending up in the residual feature. As the last step, the output is smoothed by convolution with a Hann function of window size 21. Cotatron authors suggest that the residual features need to be narrow enough to provide a bottleneck for the model; otherwise, the model learns to cheat during the training and encodes speaker features.

Cotatron [1] Authors say that the cotatron feature *L* and residual feature *R* are given to the vocal conversion decoder (VC Decoder). As a first step, the given cotatron feature and residual feature are concatenated channel-wise. Then the target speaker embedding y^{id} is obtained from the lookup table, and the concatenated cotatron feature *L* and residual feature *R* is conditioned

with the target speaker embedding. The result of this is decoded into a mel spectrogram that can be turned into raw audio using different approaches; the Cotatron authors used MelGAN.

2.7 Blow

The authors [15] describe Blow as a flow-based single-scale hyperconditioned flow that features a many-block structure with shared embedding and performs conversion in a forward-backward manner. Blow is inspired by the image generation model Glow [16].

Blow is a flow-based generative model. Blows authors [15] describe that these kinds of models learn a bijective mapping from input samples $x \in X$ to latent representations $z \in Z$. These kinds of mapping are called normalizing flows. Authors describe that they are composed of a sequence of invertible transformations.

According to its authors [15], Blow uses a single-scale structure because when using a multi-scale structure, speaker identity traits were only present at a coarser level of representation. When using a single-scale structure and carrying the same input dimensionality across all blocks, gradients flowed without issues.

Authors [15] use many blocks structure to overcome the limitation of using raw audio as input. Typical flow-based models use for input one block with input sizes of 32×32 and 256×256; when using these for raw audio, the input would be 256×1 which corresponds to 16ms of audio sampled at 16 kHz. That is too short of use for voice conversion because phenome durations are between 50 – 180 ms. Blow authors explain that to mitigate this, Blow uses eight blocks with 12 flows each; every block has two times squeeze operations, so for all eight blocks, the total squeezing factor is 2^8.

Glow [16] based models perform the manipulation of the data in the z space that means input x is mapped to z, and then additional modification is done to z. Blow authors [15] propose a forward-backward conversion, where the input x is converted into z using the source speaker

identifier. The result is speaker identity free representation of the audio. Then the *z* is converted back to *x* with the target speaker identifier to get the converted audio.

2.8 AutoVC

Authors [17] describe autoVC as a many-to-many voice conversion model that uses a simple autoencoder framework consisting of three modules: a content encoder (produces a content embedding from speech), a speaker encoder, and a decoder. autoVC was one of the first models to perform zero-shot voice conversion (convert unseen source speakers voice to unseen target speaker voice) with moderate success. autoVC takes the input audio as mel spectrograms and outputs the converted audio as mel spectrogram.

AutoVC Authors [17] describe the goal of the speaker encoder as creating the same embedding for all of a speaker's voice clips and creating a different embedding for different speakers. Typical many-to-many voice conversion models use one-hot encoding or a lookup table for encoding the speaker identity, but autoVC needs to encode unseen speakers. The author's solution was to use a 256x1 vector as a speaker embedding; that way, it is possible to generate new embeddings after training using the speaker encoder.

Qian *et al.* [17] explain that the content encoder takes the mel spectrogram concatenated with the speaker embedding as an input. Each time step of the input is fed into convolutional layers and then into Long short-term memory (LSTM) layers. The LSTM layers are downsampled to create a bottleneck during training and force the model to learn. Authors describe the downsampling as dimension reduction along the temporal axis.

According to the autoVC authors [17], the decoder takes the speaker and content embeddings as input and then upsamples them to the original temporal resolution. Then the upsampled embeddings are concatenated and fed into convolutional layers and then LSTM layers (similarly to the content encoder). The authors describe that in the network, the channel dimensions will go down from 512 to 80, and the output will be 80 channel mel spectrogram.

To convert source speakers' voice to target speaker using autoVC, we need to generate source speaker embedding and target speaker embedding by converting an audio clip of them speaking

into mel spectrogram and feeding it into speaker encoder. Then source speaker's audio is converted to a mel spectrogram and concatenated with source speakers embedding and fed into the content encoder. The encoded content is concatenated with target speakers embedding and is then fed into the decoder that produces a converted mel spectrogram. To get raw audio from the mel spectrogram, the autoVC authors [17] used the WaveNet vocoder [10].

2.9 FragmentVC

FragmentVC [18] is an encoder-decoder architecture-based any-to-any voice conversion model. FragmentVC differs from Cotatron and autoVC by using Wav2Vec 2.0 [19] latent phonetic structure of the raw audio of the source speaker instead of mel spectrograms for the encoder input.

According to Yist Y. Lin *et al.* [18], the source encoder uses pre-trained Wav2Vec 2.0 [19] to extract features from the source speaker audio. The Wav2Vec 2.0 outputs a 768-dimensional feature vector that is then converted to 512-dimension by a two-layer neural network.

The FragmentVC authors [18] describe the process of obtaining target speaker encoding as converting the target speaker audio into mel spectrograms and then feeding them into the target encoder that produces the target encoding. Multiple target speaker audio files can be given to the target encoder to improve the conversion quality.

Yist Y. Lin *et al.* [18] explain that the decoder consists of a stack of extractors and smoothers. Their output is linearly projected, and using Tacotron 2 PostNet [13] is converted into a mel spectrogram of the converted audio.

The source speaker audio needs to be encoded with Wav2Vec 2.0 to convert source speaker audio recording into target speakers voice and then getting the source embedding by feeding the Wav2Vec output to the encoder. Then the target speaker embedding is obtained by converting one or multiple sound files into a mel spectrogram and feeding them into the target encoder. The source embedding and target embedding is given to the decoder that produces the converted mel spectrogram. The mel spectrogram can be turned into raw audio by any means.

3. Experiments

This chapter gives an overview, why Cotatron was chosen for training, details about the implementation, and a description of the training process.

3.1 Choosing the model

Multiple different factors were considered for choosing the model, training time, data needed, and quality and documentation of the published code. Ultimately Cotatron was chosen to be retrained with Estonian data.

Cotatron was chosen for multiple reasons; the conversion quality was better than older models, like Blow. The available Estonian data already contained transcription, so it did not pose any disadvantage for Cotatron. In the published Cotatron audio samples[1], authors also provide an audio sample from the Korean Cotatron model (information about different language implementations are not published), which proved the model's viability for other languages. The published code[2] has good and understandable documentation, and the code is well commented.

The Blow was not chosen for retraining with Estonian data because of long training times, over 13 days, and voice conversion quality. Cotatron authors [1] demonstrated that with the same dataset, Cotatron voice conversion was more natural sounding, audio converted using Blow sounded more robotic. Authors [1] found in their study that Blow outperformed Cotatron with similarity, but because of the robotic voice, the overall quality was lower. After training Cotatron, Blow was considered to be also trained with Estonian data. The pre-processing and training process was straightforward, but it could not be trained due to time limitations.

The autoVC conversion quality is promising, but it was not chosen for multiple reasons. The published code[3] does not contain everything for training the model, for example, saving the model, and the documentation is not as good as others. The second reason was that from its GitHub issues, it seems many people have had trouble recreating the results.

[1] Cotatron audio samples https://mindslab-ai.github.io/cotatron/
[2] Cotatron - Official PyTorch Implementation https://github.com/mindslab-ai/cotatron
[3] AutoVC published code https://github.com/auspicious3000/autovc

The FragmentVC model was planned to be trained alongside Cotatron to compare results, but due to technical difficulties getting it to work and not having enough time to fix the problems and train the model, it was not trained. FragmentVC has good documentation of the published code[4] , and the code is understandable.

3.2 Implementation Details

For this thesis, the PyTorch[5] implementation of the Cotatron was used. The existing implementation contained the code used for pre-processing, training the Cotatron and Vocal Conversion Decoder. The original code was made for the English language, so it had to be modified to contain the unique letter "ö", "ä"," ü" and "õ" that are not present in the English alphabet. The Cotatron pre-processing framework removes all of the unknown symbols from the transcription, only leaving the Estonian alphabet letters and punctuation; the numeric characters are intentionally left out of scope due to spelling of numbers in Estonian is non-trivial. The transcription is converted to a lower case to simplify the input. All of the allowed characters are assigned a numeric id because the model only allows numeric inputs, so the input transcription is converted into a sequence of integers. The modified Cotatron models link can be found in Appendix I.

The audios are converted into 80-channel mel spectrograms using a sampling rate of 22.05 kHz using Short-time Fourier transform (STFT) with a window size of 1024, hop size 256 using Hann window, with minimum frequency 70 Hz and maximum 8000 Hz.

For converting mel spectrograms into raw audio, pre-trained MelGAN[6] provided by Cotatron authors was used. The MelGAN was trained on the English dataset. The model was not retrained because converting Estonian speech into mel spectrogram and then back to raw audio using the provided MelGAN model works very well and did not introduce any artefacts.

[4] FragmentVC published code https://github.com/yistLin/FragmentVC
[5] Cotatron - Official PyTorch Implementation https://github.com/mindslab-ai/cotatron
[6] Pre-trained MelGAN https://drive.google.com/uc?id=1sYSFFxL0JFUzP1TdhjuFfOEPByQGryxt

3.3 Dataset

For training the Cotatron and its vocal conversion decoder, different text-to-speech datasets were used, four of them were studio-recorded, and one was crowdsourced. The studio recorded datasets were: The News sentence corpus, The Institute of the Estonian Language Literature corpora, The Institute of the Estonian Language single sentences corpus, and The ERR radio news corpus. The crowdsourced dataset was Common Voice.

For the studio-recorded datasets, the Kõnekorpus [20] was used because it contained all of them, and the metadata was processed the same way. It contains 13 different speakers and contains about 118.5 hours of studio-recorded speech and 69 561 different sentences. It is combined from 4 different corpora:

- The News sentence corpus (*Uudislausete kõnekorpus)* [21] consists of 65.9 hours of audio from 4 different speakers and, in total 36 000 sentences. The audio is from students reading news in a studio setting.
- The Institute of the Estonian Language Literature corpora *(Ilukirjanduskorpused*) [22] consists of 26.5 hours of audio from 2 different speakers and 16 000 sentences. The audio is about literature.
- The Institute of the Estonian Language single sentences corpus *(EKI üksiklausete korpused)* [23] consists of 11.3 hours of audio from 4 different speakers and 6545 sentences. The audio is single sentences.
- The ERR radio news corpus *(ERRi raadiouudiste korpus)* [24] consists of 14.8 hours of audio from 7 different speakers and 11 016 sentences. The audio is from the news presenter reading the news.

Common voice [25] is a crowdsourced speech recognition corpus; in time of writing, it contained 538 different speakers and in total contained 27 hours of data, of which 19 hours are validated. The data in this dataset is crowdsourced; volunteers can read in different sentences on their computer or mobile device. For that reason, this dataset is noisier, and the recordings have more background noise than Kõnekorpus [20]. The dataset is also validated by volunteers who check if the recordings ate understandably and contain the right text. The dataset is made for speech recognition, but the structure is the same as in text-to-speech datasets, which suits this thesis purpose.

Before training the Cotatron and its vocal conversion decoder, the dataset was resampled to be in 22050 Hz. The Common voice dataset was also converted from MP3 to wav because the models only accept audio files in wav format. From the Common Voice dataset, only the validated split was used to ensure the dataset's quality. Then all lines longer than 10 seconds were removed from the dataset; this removed about 12 % of the Kõnekorpus dataset and 5% of the Common Voice dataset; the percentages are calculated from sentences count. Removing longer sentences was done to allow more efficient batching during training. Without removing long sentences, Cotatron batch size needed to be 16, but whit the removal of long sentences, it was able to be trained with batch size 64 witch mimics the original Cotatron training.

In total, the combined dataset had 495 different speakers and had 71 434 sentences totaling 111 hours of speech and an average sentence length of 5.3 seconds. For training, all of the metadata files needed to be combined. Kõnekorpus had separate metadata files for each speaker, where there was sentence transcription and a path to the corresponding wav file. The common voice had one metadata file containing sentence transcription, speaker id, the path to the corresponding MP3, and other information not relevant for the current thesis, like speaker age and gender. The metadata files were combined, and missing info was added like speaker id for Kõnekorpus metadata files. In Common Voice, the .mp3 in the path were replaced with .wav, the metadata format was “Path_to_wav|transcription|speaker_id”.

3.4 Training

Cotatron model and Cotatron vocal decoder were trained on the University of Tartu High-Performance Computing Centre (HPC) [26][26] using Tesla V100 GPUs. Cotatron was trained with a batch size of 64, using Adam optimizer [27] with a learning rate of 3×10^{-4} using weight decay of 1×10^{-6} and teacher force of 0.5. In total, Cotatron was trained for 92 epochs, and it took six days. The training was stopped when loss converged, and the model produced stable alignment.

The Cotatron vocal decoder was trained with a batch size of 128 using Adam optimizer [27] with a learning rate of 3×10^{-4} using weight decay of 1×10^{-6}. The training took nine days and 150 epochs. The training was stopped when loss converged, and the model produced stable alignment.

4. Results

The nuances of human speech make an automatic evaluation for voice conversion difficult. There is no ground truth in the given dataset about how a sentence would sound if another speaker said it. Because of this, voice conversion systems are evaluated by subjective measures like Mean opinion score (MOS), where different samples of converted speech are evaluated by people on a scale of one to five, where one is bad, and five is excellent. Usual metrics for that are evaluated are naturalness and similarity. The retrained model was evaluated by conducting a survey.

4.1 Survey setup

For analyzing the results of the voice conversion quality, a survey was conducted. For the survey audio sample generation, speakers were split into five groups by the amount of training data available for the speaker:

- Top five speakers by training data, training samples count between 10 000 and 5000 samples, all of the speakers were from the *kõnekorpus* [20].
- The twelve remaining speakers from *kõnekorpus*, training samples, count between 2000 and 1000 samples.
- The top thirteen speakers by training data in the common voice [25] corpus, training samples count between 600 and 60 samples.
- Speakers with little data, training samples count between 50 and 30 samples
- Speakers with very little data, training samples count between 30 and 1 samples

For each group, twenty-five samples were generated with randomly selected source speech from the whole training set. In total, one hundred and twenty-five samples were generated. Initial tests showed that to rate twenty-five samples would take about fifteen to twenty minutes. According to a study conducted by Melanie Revilla *et al.* [28], the maximum length of a web survey should be twenty minutes; after that, people might not complete the survey. Because of this, having survey participants rate all one hundred and twenty-five samples would not have been feasible because it would take over an hour and a half to complete, and participants would be affected by decision fatigue. For that reason, the samples were randomly shuffled and split into five groups. Each survey participant was randomly assigned one of the five groups and had to rate twenty-five samples.

In the survey, participants could see the source speech transcription and listen to the source speaker audio clip, a random clip from the target speaker, and the converted audio. After listening to the audio files, they had to rate the similarity and naturalness of the audio on a scale of one to five. One sample from the survey is shown in Figure *5*.

Figure 5. One sample in the survey.

No publicly existing solution meeting all of the requirements was found, so a simple web application was made for conducting the survey. The application was made using the Vue.js[7] framework, it was hosted on Netlify[8] , and the participant's answers were stored using Google Sheets API[9]. The audio samples used in the survey can be listened to in the link provided in Appendix II.

4.2 Survey results

In the survey, there were in total sixty-three participants. The participants were assigned a study group at random when they opened the survey; due to the small sample size, the different study groups were assigned slightly unevenly. Two groups got eighteen, one ten, one nine, and one eight. Each study group contained random samples from the five-speaker groups; thus, the unevenness should not affect the results. Table *1* shows the results of the survey by each study group and the overall naturalness and similarity.

The naturalness of the model was quite low; the average naturalness rating was 1.918 out of five. The best naturalness score (2.156) was for speakers from group two (*kõnekorpus* speakers with training samples count between two thousand and thousand samples). The lowest

[7] Vue.js webpage: https://vuejs.org/
[8] Netlify webpage: https://www.netlify.com/
[9] Google Sheets API webpage: https://developers.google.com/sheets/api

naturalness score (1.36) was expectably from speaker group five (speakers with less than thirty training samples). Unexpectedly, study group one, which had the most amount of training data, performed worse than groups two, three, and four.

The similarity scores were better than naturalness; the average similarity score was 2.42 out of five. Study group one had the best similarity scores (2.55), and speaker group five had the lowest similarity score (2.1).

	Naturalness	**Similarity**
Speaker group 1	1.82 ± 0.83	2.55 ± 1.18
Speaker group 2	2.15 ± 0.92	2.43 ± 1.18
Speaker group 3	1.99 ± 0.95	2.40 ± 1.13
Speaker group 4	2.02 ± 0.98	2.49 ± 1.19
Speaker group 5	1.36 ± 0.64	2.10 ± 0.96
Average	**1.91 ± 0.92**	**2.42 ± 1.15**

Table 1. Surveys naturalness and similarity MOS scores.

The results from the survey were interesting; the speaker groups one to four had very similar results in naturalness and similarity. The participants found the converted voices with thirty training samples performed as well as voices with thousands of training samples. These results show that the model needs about thirty samples of a speaker to learn how to convert the speech to that speaker's voice. The results also show that a lot of training data per one speaker will not make it better.

The speaker group five performed expectedly worse than other groups. Most of the converted senates from that group were not understandable, which resulted in average naturalness of one out of five. The speech similarity scores were better than naturalness; they had an average rating of two out of five. This shows that the model is much better at capturing speakers' voices than synthesizing natural-sounding results.

Some participants of the survey were asked to give feedback about the model's performance. Multiple people pointed out that the model performed better on male voices than female voices. Figure *6* shows that the model's naturalness scores were better for female voices, and the similarity was better for male voices. The difference in the conversion quality by gender might be caused by mel spectrograms as input for the model. The mel scale is logarithmic to mimic human hearing, which means higher pitch sounds have a lower resolution in the spectrogram.

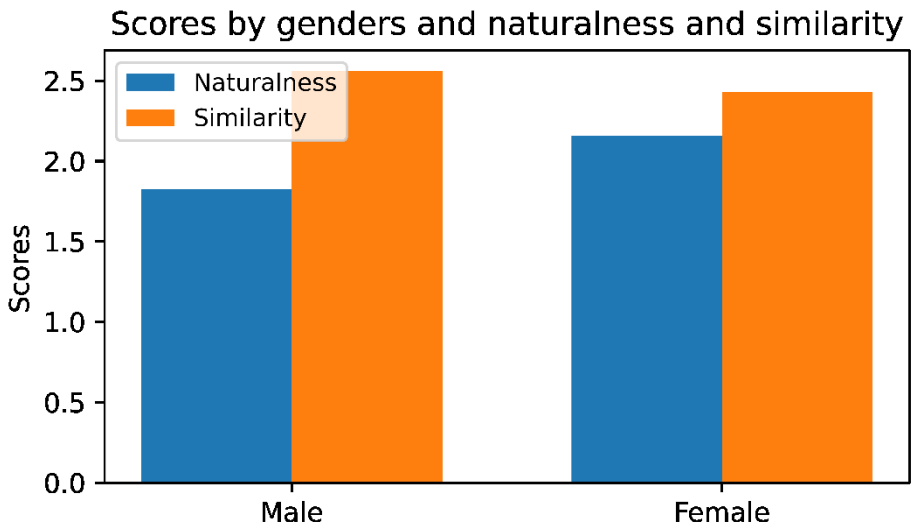

Figure 6. Naturalness and similarity scores by gender.

Table 1 shows that speaker group 1 had a lower mean similarity and naturalness score. This result is surprising because these speakers had more training data than other groups. One possible cause for this might be overfitting. The model does not learn how to generalize for the speakers in the study group. This theory could not be verified due to time limitations.

4.3 Results compared to the original English model

The original Cotatron model trained by Seung-won Park *et al.* [1] for the English language achieved a naturalness mean opinion score of 3.41 and a similarity mean opinion score of 3.89. These results also show that the Cotatron model performs better in similarity than naturalness. The results differ from the Seung-won Park et al. [1] English model. The Estonian model trained for this thesis is the average values of naturalness and similarity; the naturalness score is 44% lower, and the similarity is 38% lower.

5. Conclusion

In this thesis, the Cotatron model was retrained using multiple Estonian text-to-speech datasets. The model was slightly modified to make the model compatible with the unique letters in the Estonian alphabet. The performance of the Estonian Cotatron model was evaluated by a study, where participants rated the conversion naturalness and similarity to the target speaker.

The Cotatron model was trained on multiple Estonian text-to-speech corpora; after pre-processing, the whole dataset used for training contained 495 different speakers and had 71 434 sentences totaling 111 hours of speech and an average sentence length of 5.3 seconds. The Cotatron was trained for six days, and the voice conversion decoder was trained for nine days; the total training time was fifteen days.

Five different speaker groups were made; each one contained speakers with different amounts of training data. Then, for each speaker group, twenty-five conversion samples were generated. The samples were randomly shuffled into five study groups. The study was conducted as a web survey, where participants were randomly assigned one of the study groups containing twenty-five samples of the conversion. The participants rated the conversion on a scale of one to five in naturalness and similarity. In total, sixty-three people took part in the survey.

The Estonian Cotatron model's mean naturalness and similarity score were 1.91 and 2.42, respectively. Compared to the English model, the performance was about 40% worse, which shows that the implementation, training process, or the data used for training could be improved to get a better voice conversion system.

6. References

[1] Seung-won Park, Doo-young Kim, Myun-chul Joe, Cotatron: Transcription-Guided Speech Encoder for Any-to-Many Voice Conversion without Parallel Data, 2020, arXiv:2005.03295

[2] Sarvepalli, Sarat Kumar, Deep Learning in Neural Networks: The science behind an Artificial Brain, 2015, DOI:10.13140/RG.2.2.22512.71682.

[3] Ilya Sutskever, Oriol Vinyals, Quoc V. Le, Sequence to Sequence Learning with Neural Networks, 2014, arXiv:1409.3215.

[4] Oleh Matsuk, Multi-speaker Text-to-speech Synthesis in Estonian 2021, UT computer science institute master's thesis, https://comserv.cs.ut.ee/ati_thesis/datasheet.php?id=71053&year=2021

[5] Dzmitry Bahdanau, Kyunghyun Cho, Yoshua Bengio, Neural Machine Translation by Jointly Learning to Align and Translate, 2014 arXiv:1409.0473.

[6] Berrak Sisman, Junichi Yamagishi, Simon King, Haizhou Li, An Overview of Voice Conversion and its Challenges: From Statistical Modeling to Deep Learning, 2020, arXiv:2008.03648

[7] Yi Zhao, Wen-Chin Huang, Xiaohai Tian, Junichi Yamagishi, Rohan Kumar Das, Tomi Kinnunen, Zhenhua Ling, Tomoki Toda, Voice Conversion Challenge 2020: Intra-lingual semi-parallel and cross-lingual voice conversion, 2020, arXiv:2008.12527

[8] Griffin, D. and Lim, J. Signal estimation from modified short-time fourier transform. IEEE Transactions on Acoustics, Speech, and Signal Processing, 1984, 32(2):236–243

[9] Yuxuan Wang RJ Skerry-Ryan Daisy Stanton Yonghui Wu Ron J. Weiss Navdeep Jaitly Zongheng Yang Ying Xiao Zhifeng Chen Samy Bengio Quoc Le Yannis Agiomyrgiannakis Rob Clark Rif A. Saurous. Tacotron: Towards end-to-end speech synthesis. 2017 arXiv:1703.10135.

[10] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, Koray Kavukcuoglu, WaveNet: A Generative Model for Raw Audio,2016, arXiv:1609.03499

[11] Kundan Kumar, Rithesh Kumar, Thibault de Boissiere, Lucas Gestin, Wei Zhen Teoh, Jose Sotelo, Alexandre de Brebisson, Yoshua Bengio, Aaron Courville, MelGAN: Generative Adversarial Networks for Conditional Waveform Synthesis, 2019, arXiv:1910.06711

[12] Aaron van den Oord, Yazhe Li, Igor Babuschkin, Karen Simonyan, Oriol Vinyals, Koray Kavukcuoglu, George van den Driessche, Edward Lockhart, Luis C. Cobo, Florian Stimberg, Norman Casagrande, Dominik Grewe, Seb Noury, Sander Dieleman, Erich Elsen, Nal

Kalchbrenner, Heiga Zen, Alex Graves, Helen King, Tom Walters, Dan Belov, Demis Hassabis, Parallel WaveNet: Fast High-Fidelity Speech Synthesis 2017, arXiv:1711.10433

[13] Jonathan Shen, Ruoming Pang, Ron J. Weiss, Mike Schuster, Navdeep Jaitly, Zongheng Yang, Zhifeng Chen, Yu Zhang, Yuxuan Wang, RJ Skerry-Ryan, Rif A. Saurous, Yannis Agiomyrgiannakis, Yonghui Wu, Natural TTS Synthesis by Conditioning WaveNet on Mel Spectrogram Predictions, 2017. arXiv:1712.05884

[14] Ye Jia, Yu Zhang, Ron J. Weiss, Quan Wang, Jonathan Shen, Fei Ren, Zhifeng Chen, Patrick Nguyen, Ruoming Pang, Ignacio Lopez Moreno, Yonghui Wu, Transfer Learning from Speaker Verification to Multispeaker Text-To-Speech Synthesis, 2018. arXiv:1806.04558

[15] Joan Serrà, Santiago Pascual, Carlos Segura, Blow: a single-scale hyperconditioned flow for non-parallel raw-audio voice conversion, 2019, arXiv:1906.00794

[16] Diederik P. Kingma, Prafulla Dhariwal, Glow: Generative Flow with Invertible 1x1 Convolutions, 2018, arXiv:1807.03039

[17] Kaizhi Qian, Yang Zhang, Shiyu Chang, Xuesong Yang, Mark Hasegawa-Johnson, AUTOVC: Zero-Shot Voice Style Transfer with Only Autoencoder Loss, 2019, arXiv:1905.05879

[18] Yist Y. Lin, Chung-Ming Chien, Jheng-Hao Lin, Hung-yi Lee, Lin-shan Lee, FragmentVC: Any-to-Any Voice Conversion by End-to-End Extracting and Fusing Fine-Grained Voice Fragments With Attention, 2020, arXiv:2010.14150

[19] Alexei Baevski, Henry Zhou, Abdelrahman Mohamed, Michael Auli, wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations, 2020, arXiv:2006.11477

[20] Liisa Rätsep, Liisi Piits, Hille Pajupuu, Indrek Hein, Mark Fišel, Neural Speech Synthesis for Estonian. 2020, arXiv:2010.02636.

[21] Mark Fišel and Liisa Rätsep, "Eesti uudislausete kõnekorpus," 2020, https://doi.org/10.15155/9-00-0000-0000-0000-001ABL.

[22] E. K. Instituut, "Ilukirjanduse kõnekorpused Meelis ja Külli," 2020. https://www.eki.ee/litsents/

[23] E. K. Instituut, "EKI representatiivsed üksiklausete kõnekorpused," 2019, 2020. https://www.eki.ee/litsents/

[24] Radio News of Estonian Public Broadcasting, 2010, https://doi.org/10.15155/9-00-0000-0000-0000-00021L.

[25] Rosana Ardila, Megan Branson, Kelly Davis, Michael Henretty, Michael Kohler, Josh Meyer, Reuben Morais, Lindsay Saunders, Francis M. Tyers, Gregor Weber, Common Voice: A Massively-Multilingual Speech Corpus, 2019, arXiv:1912.06670.

[26] University of Tartu, "UT Rocket." share.neic.no, DOI: 10.23673/PH6N-0144.

[27] Diederik P. Kingma, Jimmy Ba, Adam: A Method for Stochastic Optimization, 2014, arXiv:1412.6980

[28] Revilla M, Ochoa C. Ideal and Maximum Length for a Web Survey. International Journal of Market Research. 2017;59(5):557-565. doi:10.2501/IJMR-2017-039

Appendix

I. Estonian Cotatron implementation and audio samples

The code used for training the Estonian Cotatron model can be accessed at: https://github.com/hzuppur/estonian_cotatron

II. Estonian Cotatron audio samples

The audio samples can be listened to at: https://estonian-cotatron-audio-samples.netlify.app/

Licence

Non-exclusive licence to reproduce thesis and make thesis public

I, Hain Zuppur

1. Herewith grant the University of Tartu a free permit (non-exclusive licence) to reproduce, for the purpose of preservation, including for adding to the DSpace digital archives until the expiry of the term of copyright, "Creating a Voice Conversion Model for Estonian", supervised by Liisa Rätsep.

2. I grant the University of Tartu a permit to make the work specified in p. 1 available to the public via the web environment of the University of Tartu, including via the DSpace digital archives, under the Creative Commons licence CC BY NC ND 3.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright.

3. I am aware of the fact that the author retains the rights specified in p. 1 and 2.

4. I certify that granting the non-exclusive licence does not infringe on other persons' intellectual property rights or rights arising from the personal data protection legislation.

Hain Zuppur

07.05.2021