Speech Synthesis Using Stressed Sample Labels for Languages with Higher Degree of Phonemic Orthography

Arnas Radzevičius[1], Aistis Raudys[2], and Pijus Kasparaitis[3]

[1] AAI Labs, Suopių g. 21, Riešė, LT-14265, Lithuania arnas@aai-labs.com
[2] Institute of Informatics, Vilnius University, Naugarduko 24, LT-03225, Vilnius, Lithuania, aistis.raudys@mif.vu.lt
[3] Institute of Informatics, Vilnius University, Naugarduko 24, LT-03225, Vilnius, Lithuania, pijus.kasparaitis@mif.vu.lt

Abstract. Introducing neural networks into the field has improved the performance of speech synthesis systems significantly. Most research was done on the English language which has substantial speech data resources, however, a low degree of grapheme-phoneme correspondence. Other, low resource languages pose different challenges that may be overcome using different approaches to text embeddings. In the present paper we present the results of using stressed text labels in speech datasets to train a speech synthesis model for a low speech data resource language - Lithuanian. Nvidias implementation of the Tacotron 2 system and the Lithuanian language speech dataset (corpus) with stressed text labels were used to train speech synthesis models. By introducing accentuation into sample labels, we show a significant improvement in speech naturalness as measured by MOS.

Keywords: Speech synthesis · neural networks · datasets · WaveGlow · Tacotron 2 · phonemic orthography

1 Introduction

A phonemic orthography is an orthography in which the graphemes (written symbols) correspond to the phonemes (spoken sounds) [1]. Some languages have a higher degree of grapheme-phoneme correspondence. This means that words are pronounced very similarly to the way they are spelled. Finnish, Albanian, Georgian, and other languages are good examples of that. Other phonemic orthographies can be defective (e.g., English), or slightly defective (e.g., Lithuanian, which will be used as an example throughout this paper). This information is important to the implementation of a speech synthesis system. A lot of research was carried out to convert graphemes to phonemes in highly non-phonemic languages, such as English [2–4]. The converted graphemes are then used to train a deep neural network to synthesize speech. Without this conversion, some neural networks struggle to ensure correct pronunciation of synthesized speech [5].

However, the opposite may be the case for languages with higher grapheme-phoneme correspondence. In such languages, most of the letters are pronounced the way they are spelled. For example, the majority of letters in the Lithuanian language correspond to the same phoneme regardless of the context of a sentence. Still, it is not a complete phonemic orthography, since the vowels are not fully distinguished [6]. Nevertheless, the phonemic complexity is lower as compared to that in the English language; this facilitates the task of preparing the text for neural network training.

In the Lithuanian language, the same word often has more than one pronunciation. It depends on the accentuated (stressed) letter, e.g., the word "pastato" is pronounced differently in the following two sentences: "eĩkite tiẽsiai iki pãstato gãlo", "Jìs pastãto dáiktą añt stãlo.". The morphological analysis can help to determine the stress position in the word of the sentence [7]. Another approach is to use stressed text data to train a neural network that can put accents on appropriate words in a sentence. The latter is more preferable for a neural network TTS system application.

Since the main problem that sets the Lithuanian language apart from a phonemic orthography is that some of its vowels are not fully distinguished (the same letter can be used for the short and long vowels, e.g., "a", "e"), using a stressed-label speech dataset may be sufficient to train a neural network to synthesize speech. This eliminates (or replaces) the need for pronunciation dictionaries [8] which are not publicly available for many languages and are hard to compile. This is especially important to the languages that have low speech data resources.

To leverage all that, we will use Nvidia's implementation of Tacotron 2 as an acoustic model and a WaveGlow vocoder. It is possible to use raw text as dataset sample labels to train the Tacotron 2 model, however, our experiments show that this has a negative impact on the performance of a speech synthesis model. The results also show the advantages of using the stressed text as sample labels. Further work may include modifying TTS systems where input text embeddings originally are phonemes. An example of such a system is Microsofts FastSpeech 2 [9], in which the vocabulary of phonemes is used to train the model. We could experiment to discover whether the stressed text would work equally well for the languages with a higher degree of a phonemic orthography.

2 Background

In the present section, we are going to review briefly the background of this work, including the Tacotron 2 and WaveGlow neural networks, Lithuanian speech corpus stressing, the TTS system evaluation by MOS, and, finally, a brief history of other Lithuanian text-to-speech synthesizers.

2.1 Tactoron 2 and Nvidia

Tacotron 2 is a neural network architecture for speech synthesis directly from the text [10]. Like most TTS (text-to-speech) systems, the TTS synthesis process

takes place in two steps [11–13]. First, the input character embeddings are converted to a mel-scale spectrogram using a recurrent sequence-to-sequence feature prediction network that is based on encoder-decoder architecture with an autoregressive decoder. Such a system is typically called an acoustic model. The outputs of the acoustic model are then fed into the WaveNet [12] model (that was modified to accept mel-spectrograms as inputs) which uses the spectrogram to synthesize a time-domain waveform. WaveNet is called a vocoder model since it can synthesize audio from a given sequence of features. It generates audio by sampling one frame at a time and is conditioned on the previously generated frames. To summarize, the original Tacotron 2 system uses two main components - a Tacotron acoustic model and a WaveNet vocoder - to synthesize speech. The present authors claim that the quality of speech synthesis done by the model rivals that of a real human speaker. The main disadvantage of the system, however, is that the inference time is much slower than real-time because WaveNet generates each audio frame sequentially; consequently, it cannot be deployed in a real-world application.

In this paper, Nvidias implementation of Tacotron 2 [14] is used to train a Lithuanian speech synthesis model. Nvidias Tacotron 2 differs from the system described in the original paper in that it uses WaveGlow [13, 15] instead of WaveNet [12] as a vocoder. This greatly speeds up inference time. WaveGlow is a generative model that generates audio by sampling from the distribution of audio samples conditioned on a mel-spectrogram [13]. With the help of Tacotron, a trained WaveGlow model can sample a good quality waveform from a mel-spectrogram much faster than in real-time. The MOS score provided by the authors of the WaveGlow paper shows that the vocoder synthesizes audio quality similar to that generated by WaveNet.

2.2 Text Stressing

Instead of using raw text embeddings to train the Tacotron 2 model as it was done in the original paper and Nvidias implementation when training the English language speech synthesis model, we will use a Lithuanian language speech dataset where each sample label (sentence) is stressed. The program that uses the algorithm based on the morphological analysis [7] was used to semi-automatically stress the speech corpus sample labels. The program runs through dataset sample labels, stresses each word and marks the words that can be stressed in multiple ways or the words that were not stressed. Then the user of the program can choose appropriate stressing of the words from the proposed options or put the accents manually.

This program was used to stress 91-hour Lithuanian speech corpus sample labels. Each accent (stress mark) is actually a UTF-8 combining character [16] that is combined with a previous letter in a character sentence, e.g., two UTF-8 symbols a and \u0300 are combined into 'à'. We used three combining characters (accents) as character embeddings when training a TTS model: \u0300 (grave), \u0301 (acute), and \u0303 (tilde). Using these character embeddings, a neural network can learn which letter in a word should be accented, thus

improving naturalness of synthesized speech. See Section 3.1 for more details on the Lithuanian speech corpus stressing.

2.3 Mean Opinion Score

The mean opinion score (MOS) [17] is a subjective evaluation metric of text-to-speech systems. It is calculated by collecting the results of a subjective listening test. Subjective listening tests are generally regarded as the most reliable and definitive way of assessing speech quality and naturalness [18]. In general, subjective quality measures require that:

- there are enough listening subjects of sufficient diversity to produce statistically significant results;
- experiments are conducted in the controlled environment with specific acoustic characteristics and equipment;
- every subject receives the same instructions and stimuli.

In the MOS test, the listeners evaluate randomly selected samples from a pool of speech samples (either signals or utterances). In the case of TTS MOS, the pool of speech samples usually contains audio samples generated by multiple TTS systems and ground truth (real human speech samples). The MOS score is calculated for each of the sources. Also, the limitation is that the same listening test should never contain two samples created from the same utterance.

At the beginning of the test, listeners are asked to use headphones to achieve better results because by using loudspeakers people have a smaller discrimination capacity. They are also provided with the MOS score table, like that in Table 1.

Table 1. Mean opinion score table [18]

Rating	Quality	Distortion
5	Excellent	Imperceptible
4	Good	Just perceptible, but not annoying
3	Fair	Perceptible and slightly annoying
2	Poor	Annoying, but not objectionable
1	Bad	Very annoying and objectionable

2.4 Other Lithuanian Text-to-Speech Synthesizers

The synthesis of the Lithuanian speech has a history of about 30 years. A comprehensive review was published 5 years ago [19]. The first Lithuanian commercial format speech synthesizer Apollo was developed by Dolphin systems Inc. in 1994. Later, concatenative synthesizers, mainly based on diphones, were developed and refined for a long time. In 2013-2016, a significant breakthrough

in terms of voice quality was achieved with the development of unit selection synthesizers. In recent years, statistical-parametric synthesis using the Merlin neural network package has been mastered and this method has been shown to outperform unit selection synthesis [20].

3 Experimental Setup

3.1 Dataset

To train a neural network, a 91-hour one-speaker Lithuanian speech dataset was created. The first stage in creating the dataset was to collect multiple audiobooks narrated by Vytautas Radzevičius and their e-book counterparts. Then the text and audio speech were aligned using Aeneas, an open-source tool [21]. This process is called forced alignment [22] and it results in a sync-map that contains the start and end timestamp of each sentence in the audio. Then, using this sync-map, the audio was cut into samples of the length ranging from 2 to 14 seconds. Finally, the data was structured as a set of audio sample and label (transcription) pairs.

Nevertheless, the quality of the dataset is relatively poor. That is because the Aeneas aligning tool is not perfect and makes mistakes even after trying out multiple configurations. Another drawback is that the speaker does not always narrate the same words as in the corresponding e-book due to a human factor. Some manual work was done to correct the dataset as well as possible, but the audio-text pairs are still not completely accurate (correct). This affected the quality of the model, which is reflected in MOS (see later Sections).

After building the audio and text pair dataset, each sample label (text sentence) was automatically stressed using the program described in Section 2.2. The stressed text labels were manually reviewed and fixed by a professional philologist to ensure correctness. The stressed-text speech corpus was later used in model training.

3.2 Model Configuration

To train both Tacotron 2 and WaveGlow, the same hyperparameters as in corresponding papers were used. Only the batch sizes were reduced (from 64 to 16 for Tacotron 2 and from 24 to 12 for WaveGlow) because of a lack of computational resources. The hyperparameters for Tacotron 2 and WaveGlow are summarized in Tables 2 and 3, respectively. Other details can be found online.

3.3 Training Setup

We carried out the experiments on Google Virtual Machine with an 8-core CPU, 1 x Nvidia Tesla T4 GPU and 30 GB of RAM. We used publicly available pre-trained models published by Nvidia for both Tacotron 2 and WaveGlow. We used them to fine-tune the models on our data to save computational resources. The

Table 2. Tacotron 2 hyperparameters [14].

Hyperparameter	Value
Character embedding dimension	512
Encoder kernel size	5
Encoder number of convolutions	3
wpswps Encoder embedding dimension	512
Decoder number of frames per step	1
Decoder RNN dimension	1024
Decoder pre-network dimension	256
Max decoder steps	1000
Decoder gate threshold	0.5
Pre-attention dropout	0.1
Pre-decoder dropout	0.1
Attention RNN dimension	1024
Attention dimension	128
Attention location number of filters	32
Attention location kernel size	31
Mel-post processing network embedding dimension	512
Mel-post processing network kernel size	5
Mel-post processing network number of convolutions	5

Table 3. WaveGlow hyperparameters [15].

Hyperparameter	Value
Mel-spectrogram channels	80
Coupling layers	12
Invertible 1x1 convolutions	12
Coupling layer dilated convolutions	8
Residual connections	512
Skip connections	256
Early channel output after every N layers	4
Number of early channer outputs	2
Batch size	12

fine-tuning of the vocoder is of especial importance as the authors of the paper state that they used 8 Nvidia GV100 GPUs to train for 580000 iterations with a batch size of 24, and such resources are expensive.

4 Results

The WaveGlow vocoder was fine-tuned on our speech corpus first. It took 111 hours to train the model for 84000 iterations (42 epochs). Then the Tacotron 2 model was trained for 18 hours to reach 24000 iterations (24 epochs). We conducted a MOS survey to evaluate the following four acoustic model - vocoder model combinations:

- A Tacotron model trained on a dataset without stressed text labels and a pre-trained WaveGlow model. (NP)
- A Tacotron model trained on a dataset without stressed text labels and a finetuned WaveGlow model. (NF)
- A Tacotron model trained on a dataset with stressed text labels and a pre-trained WaveGlow model. (SP)
- A Tacotron model trained on a dataset with stressed text labels and a finetuned WaveGlow model. (SF)

The survey also contained ground-truth (GT) samples. There were 6 survey participants. The participants were familiarised with the purpose of the survey, given instructions on how to complete the survey, and were asked to use headphones throughout the whole survey process to achieve more reliable results [18]. Each participant had to rate 75 audio samples generated by 5 sources (NP, NF, SP, SF, and GT) on a scale from 1 to 5. To avoid response bias, the participants were not informed on which sample was generated by which source. The rated utterances (samples) contained no numbers or abbreviations, but they did have punctuation. The shortest utterance duration was 1.5 seconds (2 words), while the duration of the longest sample was 9 seconds (18 words). The participants were provided with the MOS table (like that in Table 1) for reference. The MOS evaluations were calculated with 95% confidence intervals (CI) computed from the t-distribution.

The MOS results are provided in Table 4. As we can see, the system that has a Tacotron model trained on the stressed text and a fine-tuned WaveGlow vocoder perform significantly better than the systems that have either a Tacotron model that is not trained on the stressed text or that uses a pre-trained WaveGlow model. On the other hand, there is a fairly big gap between the best-performing speech synthesis system and the ground truth. This may be due to several reasons. The first reason, as mentioned in Section 3.1, is a poor audio-text correspondence between the dataset samples, which has a negative impact on the performance of the synthesizer. Another reason may be that the MOS survey was not conducted completely correctly, as it contained multiple signals (utterances) generated by each of the 5 sources (models, ground-truth). This practice is not recommended

Table 4. The MOS evaluation with 95% confidence intervals (CI) computed from the t-distribution comparison between the ground-truth (GT), the model that was trained on non-stressed text labels (NF) and the model that was trained on stressed text labels (SF).

Model	MOS ±CI
NP	2.267 ±0.638
NF	2.612 ±0.442
SP	2.619 ±0.870
SF	3.525 ±0.573
GT	4.786 ±0.336

[18]. Another reason may be that there were not enough survey respondents - confidence intervals are quite big for each source.

Nevertheless, the trained speech synthesis model (SF) shows fairly good results in terms of speech naturalness and pronunciation. The model responds to the punctuation and can synthesize various (difficult) sentences. The observed disadvantages were that the synthesis might fail in very short or very long sentences. It also breaks when attempts are made to synthesize two sentences separated by a dot. This might be again the fault of the poor-quality dataset, as this behavior usually is not observed in Tacotron models.

5 Future Work

5.1 Speech Corpus

The next natural step in limproving the quality of synthesized speech is to fix the errors in the Lithuanian speech corpus. This has the potential of solving the above-described multiple currenty encountered synthesis problems.

5.2 Automatic Stressing Model

The stressed dataset sample labels were generated by using a semi-automatic tool (program) that still requires human expert intervention. Currently we are working on a completely automated text-stressing neural network model. This may provide high stressing accuracy results and thus remove the need for human expert work. It would be extremely useful in creating new stressed datasets and training of new models with different voices. Also, it would ease the inference process, since now the user himself needs to put accents on the words in the sentence he wants to synthesize. A high accuracy stressing neural network model may work as a pre-processor for speech synthesis model input.

Currently, we are creating multiple high quality speech datasets narrated by different speakers. Quality checks are done by human listeners to ensure the correctness of the audio-text correspondence between the dataset samples. Since the quality of these datasets will be better than that used for the experiments

in this paper, a speech synthesis model of better performance is expected to be trained by applying transfer learning to the model described therein. These multiple speaker datasets may also stimulate further research into multi-speaker speech synthesis for languages with higher orthographic degree.

6 Conclusions

In the present paper we proposed to use the stressed text for speech dataset sample labels for languages with a higher degree of a phonemic orthography. The main problem we tried to solve was an incorrect pronunciation of synthesized speech. We used the UTF-8 combining characters \u0300, \u0301, \u0303 (accordingly, grave, acute and tilde) as character embeddings to allow the Tacotron model to learn to pronounce words when explicitly specified by accents. Our experimental results show that the use of stressed dataset sample labels significantly outperforms the TTS models trained on a dataset with raw (not stressed) sample labels. Hence, by using stressed text sample labels for languages with a higher degree of a phonemic orthography, it is possible to replace the hard to create phoneme vocabulary necessary in some speech synthesis neural networks to achieve high quality speech.

Although the proposed approach is a relatively small modification to the usual speech synthesis model training workflow, this study may be useful to the developers whose aim is to train a text-to-speech model for a low resource language that has a higher orthographic degree.

References

1. Petr Sgall. *Towards a theory of phonemic orthography.* Amsterdam (Philadelphia): John Benjamins, 1987.
2. Rita Sloan Berndt, James A Reggia, and Charlotte C Mitchum. Empirically derived probabilities for grapheme-to-phoneme correspondences in english. *Behavior Research Methods, Instruments, & Computers*, 19(1):1–9, 1987.
3. Maximilian Bisani and Hermann Ney. Joint-sequence models for grapheme-to-phoneme conversion. *Speech communication*, 50(5):434–451, 2008.
4. Kanishka Rao, Fuchun Peng, Haşim Sak, and Françoise Beaufays. Grapheme-to-phoneme conversion using long short-term memory recurrent neural networks. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4225–4229. IEEE, 2015.
5. Yi Ren, Yangjun Ruan, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. Fastspeech: Fast, robust and controllable text to speech. *arXiv preprint arXiv:1905.09263*, 2019.
6. Laimute Balode and Axel Holvoet. The lithuanian language and its dialects. *Circum-Baltic Languages: Typology and Contact*, pages 41–80, 2001.
7. Pijus Kasparaitis. Automatic stressing of the lithuanian text on the basis of a dictionary. *Informatica*, 11(1):19–40, 2000.
8. John Christopher Wells and Tony TN Hung. Longman pronunciation dictionary. *RELC Journal*, 21(2):95–97, 1990.

9. Yi Ren, Chenxu Hu, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. Fastspeech 2: Fast and high-quality end-to-end text to speech. *arXiv preprint arXiv:2006.04558*, 2020.
10. Yuxuan Wang, RJ Skerry-Ryan, Daisy Stanton, Yonghui Wu, Ron J Weiss, Navdeep Jaitly, Zongheng Yang, Ying Xiao, Zhifeng Chen, Samy Bengio, et al. Tacotron: Towards end-to-end speech synthesis. *arXiv preprint arXiv:1703.10135*, 2017.
11. Aaron Oord, Yazhe Li, Igor Babuschkin, Karen Simonyan, Oriol Vinyals, Koray Kavukcuoglu, George Driessche, Edward Lockhart, Luis Cobo, Florian Stimberg, et al. Parallel wavenet: Fast high-fidelity speech synthesis. In *International conference on machine learning*, pages 3918–3926. PMLR, 2018.
12. Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.
13. Ryan Prenger, Rafael Valle, and Bryan Catanzaro. Waveglow: A flow-based generative network for speech synthesis. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3617–3621. IEEE, 2019.
14. Tacotron 2 (without wavenet). `https://github.com/NVIDIA/tacotron2`. Accessed: 2021-04-23.
15. Waveglow: a flow-based generative network for speech synthesis. `https://github.com/NVIDIA/waveglow`. Accessed: 2021-04-23.
16. François Yergeau. Utf-8, a transformation format of iso 10646. Technical report, STD 63, RFC 3629, November, 2003.
17. ITUT Recommendation. Telephone transmission quality subjective opinion tests. a method for subjective performance assessment of the quality of speech voice output devices. 1994.
18. Flávio Ribeiro, Dinei Florêncio, Cha Zhang, and Michael Seltzer. Crowdmos: An approach for crowdsourcing mean opinion score studies. In *2011 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 2416–2419. IEEE, 2011.
19. Pijus Kasparaitis et al. Lietuviško balso sintezatorių kokybės vertinimas. *Kalbų studijos*, (28):80–91, 2016.
20. Pijus Kasparaitis and Margarita Beniušė. Statistical parametric speech synthesis of lithuanian. In *http://lki. lt/26-oji-tarptautine-moksline-jono-jablonskio-konferencija*, page 43, 2019.
21. aeneas. `https://github.com/readbeyond/aeneas`. Accessed: 2021-04-23.
22. Pedro J Moreno, Chris Joerg, Jean-Manuel Van Thong, and Oren Glickman. A recursive algorithm for the forced alignment of very long audio segments. In *Fifth International Conference on Spoken Language Processing*, 1998.