Baltic J. Modern Computing, Vol. 11 (2023), No. 2, 285-296
https://doi.org/10.22364/bjmc.2023.11.2.05

Investigation of Input Alphabets of End-to-End Lithuanian Text-to-Speech Synthesizer

Pijus KASPARAITIS, Danielius ANTANAVIČIUS

Institute of Informatics, Faculty of Mathematics and Informatics, Vilnius University
Didlaukio 47, LT-03225 Vilnius, Lithuania

pijus.kasparaitis@mif.vu.lt, antanavicius.danielius@gmail.com

Abstract. The present paper deals with choosing the input alphabet for the end-to-end synthesizer of the Lithuanian language. Tacotron 2 is a state-of-the-art end-to-end speech synthesis model. Characters, phonemes or their combinations can be used as an input of the model. The model was trained on Lithuanian speech recordings using the following five input alphabets: letters, lowercase letters, accented letters, reduced set of accented letters, letters with separate accent marks. Acceptability of the synthesized speech was evaluated on the basis of human listeners' subjective judgment. Experimental testing showed that accent marks significantly improved the quality of the synthesized speech. Reducing the size of the input alphabet also has a slight positive impact. Putting accent marks into the text produced the best results as compared to using the accented letters.

Keywords: neural networks, natural language processing, speech synthesis, Tacotron 2, text encoding, Lithuanian language.

1. Introduction

Speech synthesis is a process of converting the input text into the corresponding speech, also known as text-to-speech (TTS). The history of the Lithuanian TTS dates back almost three decades. A detailed overview of the Lithuanian synthesis until 2016 is presented in (Kasparaitis, 2016). It includes the first formant synthesizer Apollo 2, the concatenative synthesizers Aistis, Aistis 2, Gintaras, Egidius, and finally the unit selection synthesizers SINT.AS and LIEPA. In subsequent years, a Statistical Parametric Speech Synthesizer (SPSS) was developed, which uses the neural network package Merlin (Kasparaitis and Beniušė, 2019). In the above-mentioned works, the main focus is on signal formation. For an overview of other aspects related to Lithuanian synthesis, such as intonation and sound durations, see (Melnik-Leroy et al., 2022). Lately, end-to-end Lithuanian synthesizers have started to appear, e.g. (Radzevičius at al., 2021).

Lithuanian has a very complex stress system, so it is difficult to expect end-to-end models to be able to learn it. One solution is to supplement the text with stress information, but it is not clear how best to encode it. In this work, the methods of end-to-end synthesizer input coding will be considered.

Chapter 2 reviews how input is encoded in other languages, Chapter 3 describes the five alphabets used to encode Lithuanian text, Chapter 4 – data and training conditions, Chapter 5 – the evaluation procedure, and Chapter 6 – a discussion of the results.

2. Related works

Traditional speech synthesis techniques such as SPSS are composed of several independent components (Liu and Zheng, 2019): a text analyser for extracting linguistic features such as syntactic and prosodic tags from the text, a duration model for predicting the duration of a phoneme, an acoustic model for predicting acoustic features such as mel-cepstral coefficients and F0, and a vocoder for generating a waveform from acoustic features (Yasuda et al., 2020) (see Fig. 1a). The development of these components requires deep knowledge of the domain and is labour intensive. In addition, the components are independent, so their errors can accumulate.

An integrated end-to-end TTS system that can be trained on <text, audio> pairs with minimal human annotation has many advantages (Wang et al., 2017). It requires less expert work, allows a larger number of attributes to be considered, is more robust. Typically, such systems consist of a single neural network, although a separate network can be used as a vocoder (see Fig. 1b). Nowadays, end-to-end TTS has demonstrated high effectiveness in generating a natural and emotional speech. We chose to use Tacotron 2 (Shen et al., 2018), which is a state-of-the-art end-to-end speech synthesis model capable to generate speech directly from graphemes or phonemes (Liu and Zheng, 2019).

Typically, end-to-end TTS systems use character (Wang et al., 2017) or phoneme (Jiang et al., 2019) input representations or their various combinations (Kastner et al., 2018; Ping et al., 2018). Phoneme inputs are usually preferred over graphemes (Perquin et al, 2020); this fact was reported by Fong et al. (2019) and Zhang et al. (2019). On the contrary, Yasuda et al. (2020) and Perquin et al. (2020) found that both types of inputs led to similar results if certain conditions were met.

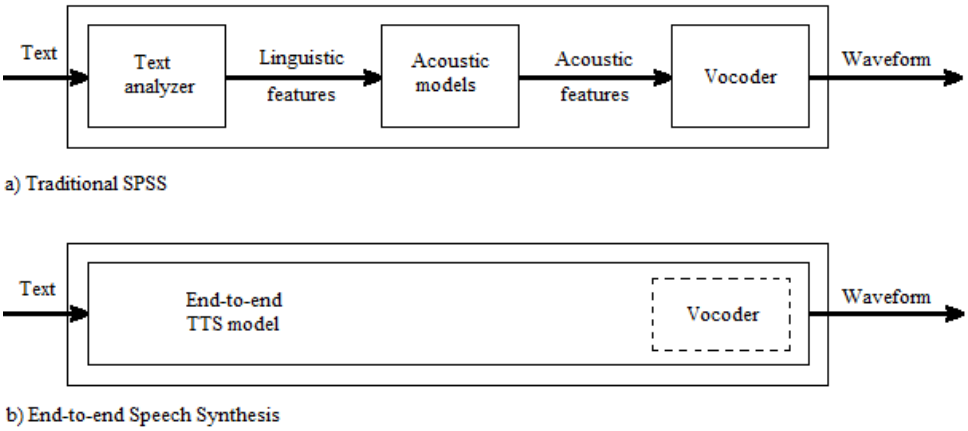

Figure 1. a) Traditional SPSS; b) End-to-end Speech Synthesis (adapted from https://www.slideshare.net/jyamagis/tutorial-on-endtoend-texttospeech-synthesis-part-1-neural-waveform-modeling slide 7)

Attempts have also been made to use richer linguistic information as additional inputs for end-to-end based TTS (Yasuda et al., 2020). E.g., Chinese is a tonal language, therefore phones with tone (Zhang et al., 2018) or pinyin (the standard system of Romanized spelling for transliterating Chinese) with tone (Liu et al., 2020) can be used. Japanese is a pitch-accented language, hence phonemes with accentual type were used by Yasuda et al. (2019).

Linguistic information can be used not only as additional inputs but also instead of graphemes or phonemes. E.g., Staib et al. (2020) used the following set of 10 categorical, multi-valued phonological features (PF): consonant/vowel, voicing (voiced/unvoiced), vowel frontness, vowel openness, vowel roundedness, stress on vowel, consonant place, consonant manner, and diacritic (e.g., nasalized, velarized). The tenth feature, "symbol type", is used to integrate symbols that mark, e.g., silences, the end of a sentence, or word boundaries.

Letters supplemented with PF (i.e., accent marks) were used as an input alphabet in the present work. Accent marks in Lithuanian contain three pieces of information: stress position, accent type (falling or rising accent) and sometimes indicates whether the sound is long or short. The Lithuanian language has a very complicated accentuation system. In Lithuanian, there are four stressing paradigms of nouns and adjectives, which are formed depending on where the stress is in the stem or in the suffix in the dative and accusative case of the plural. However, for other cases, the stress position also depends on the inflection. The stress of verbs can be not only in the stem and ending, but also in a prefix, e.g., several forms of the word "nešti" (to carry): "nùnešu", "nunèšiu", "nuneštám". However, the biggest problem is that many new word forms can be formed using prefixes, suffixes, and endings. Fortunately, there exists a well-developed dictionary and rule-based text stressing software (Kasparaitis, 2000) that has been successfully used for text stressing for many years. It is impossible for a Tacotron 2 based system to learn the accentuation system of the Lithuanian language from several-hour recordings. This was proved by Radzevičius et al. (2021): text stressing has a positive effect on the results obtained.

Encoding of accented letters is not unambiguous either. E.g., bytes of UTF-8 encoded text can be used as was proposed by Li et al. (2019). We used letters with diacritics, as well as letters and accents as separate characters. Finally, the size of the input alphabet was addressed in this work. Some authors simply concatenate grapheme sets when adding a new language to the model without worrying about the size (Zhang et al., 2019). We expect the small alphabet to be a better choice.

3. Input representations

Input alphabets we used are presented in this chapter. We will refer to elements of the input alphabet as graphemes primarily to show contrast with phonetic input. A grapheme is the smallest unit of text that is passed to a single input of the model. In this work, a grapheme usually corresponds to a single letter, a punctuation mark, or an accent mark, but may also correspond to a letter and an accent mark, or several letters (such as ⟨ch⟩).

3.1. Lithuanian alphabet

There are 32 uppercase and 32 lowercase letters in Lithuanian. Letters were arranged in the following order: vowels, plosives, fricatives, other consonants. In addition, the following punctuation marks were added to the alphabet: the dot, the question mark, the exclamation mark and the space. All commas, colons and semicolons were replaced with a dot in the text, other characters were simply removed. Hence, we get the following set A containing 68 graphemes:

A = {⟨A⟩, ⟨a⟩, ⟨Ą⟩, ⟨ą⟩, ⟨E⟩, ⟨e⟩, ⟨Ę⟩, ⟨ę⟩, ⟨Ė⟩, ⟨ė⟩, ⟨I⟩, ⟨i⟩, ⟨Į⟩, ⟨į⟩, ⟨Y⟩, ⟨y⟩, ⟨O⟩, ⟨o⟩, ⟨U⟩, ⟨u⟩, ⟨Ų⟩, ⟨ų⟩, ⟨Ū⟩, ⟨ū⟩, ⟨B⟩, ⟨b⟩, ⟨D⟩, ⟨d⟩, ⟨G⟩, ⟨g⟩, ⟨P⟩, ⟨p⟩, ⟨T⟩, ⟨t⟩, ⟨K⟩, ⟨k⟩, ⟨C⟩, ⟨c⟩, ⟨Č⟩, ⟨č⟩, ⟨S⟩, ⟨s⟩, ⟨Š⟩, ⟨š⟩, ⟨Z⟩, ⟨z⟩, ⟨Ž⟩, ⟨ž⟩, ⟨F⟩, ⟨f⟩, ⟨H⟩, ⟨h⟩, ⟨J⟩, ⟨j⟩, ⟨L⟩, ⟨l⟩, ⟨M⟩, ⟨m⟩, ⟨N⟩, ⟨n⟩, ⟨R⟩, ⟨r⟩, ⟨V⟩, ⟨v⟩, ⟨ ⟩, ⟨?⟩, ⟨!⟩, ⟨.⟩} (1)

3.2. Lowercase letters

Uppercase letters always denote the same sounds as their lowercase counterparts. Moreover, only about 2% of the letters in our texts used for training are uppercase. Seeking to avoid the shortage of data we decided to replace all uppercase letters with lowercase ones in the text and to remove uppercase letters from the alphabet. The following set B containing 36 graphemes was obtained:

B = {⟨a⟩, ⟨ą⟩, ⟨e⟩, ⟨ę⟩, ⟨ė⟩, ⟨i⟩, ⟨į⟩, ⟨y⟩, ⟨o⟩, ⟨u⟩, ⟨ų⟩, ⟨ū⟩, ⟨b⟩, ⟨d⟩, ⟨g⟩, ⟨p⟩, ⟨t⟩, ⟨k⟩, ⟨c⟩, ⟨č⟩, ⟨s⟩, ⟨š⟩, ⟨z⟩, ⟨ž⟩, ⟨f⟩, ⟨h⟩, ⟨j⟩, ⟨l⟩, ⟨m⟩, ⟨n⟩, ⟨r⟩, ⟨v⟩, ⟨ ⟩, ⟨?⟩, ⟨!⟩, ⟨.⟩} (2)

3.3. Accented letters

There are short and long syllables in Lithuanian. The short syllables can be stressed and unstressed, while the long ones can be unstressed, stressed with rising accent and stressed with falling accent. The following accent marks will be used respectively: ⟨a`⟩, ⟨a~⟩, ⟨a^⟩. If the syllable contains a diphthong or a mixed diphthong, the rising accent mark is put on the second letter of the diphthong. Accented lowercase letters were added to the set B, hence the set C containing 69 graphemes was built:

C = {⟨a⟩, ⟨a`⟩, ⟨a~⟩, ⟨a^⟩, ⟨ą⟩, ⟨ą~⟩, ⟨ą^⟩, ⟨e⟩, ⟨e`⟩, ⟨e~⟩, ⟨e^⟩, ⟨ę⟩, ⟨ę~⟩, ⟨ę^⟩, ⟨ė⟩, ⟨ė~⟩, ⟨ė^⟩, ⟨i⟩, ⟨i`⟩, ⟨i~⟩, ⟨i^⟩, ⟨į⟩, ⟨į~⟩, ⟨į^⟩, ⟨y⟩, ⟨y~⟩, ⟨y^⟩, ⟨o⟩, ⟨o`⟩, ⟨o~⟩, ⟨o^⟩, ⟨u⟩, ⟨u`⟩, ⟨u~⟩, ⟨u^⟩, ⟨ų⟩, ⟨ų~⟩, ⟨ų^⟩, ⟨ū⟩, ⟨ū~⟩, ⟨ū^⟩, ⟨b⟩, ⟨d⟩, ⟨g⟩, ⟨p⟩, ⟨t⟩, ⟨k⟩, ⟨c⟩, ⟨č⟩, ⟨s⟩, ⟨š⟩, ⟨z⟩, ⟨ž⟩, ⟨f⟩, ⟨h⟩, ⟨j⟩, ⟨l⟩, ⟨l~⟩, ⟨m⟩, ⟨m~⟩, ⟨n⟩, ⟨n~⟩, ⟨r⟩, ⟨r~⟩, ⟨v⟩, ⟨ ⟩, ⟨?⟩, ⟨!⟩, ⟨.⟩} (3)

3.4. Reduced set of accented letters

The input alphabet was reduced based on the assumption that all letters denoting the same phoneme should be represented by the same grapheme. There are several letters in Lithuanian that always denote the same phoneme and different letters are used for historical rather than phonological reasons, namely ⟨į⟩ and ⟨y⟩, ⟨ų⟩ and ⟨ū⟩. Some letters denote the same phoneme only in stressed long syllables, e.g., ⟨a~⟩ and ⟨ą~⟩, ⟨e~⟩ and ⟨ę~⟩. In Lithuanian the voiced affricates are denoted with pairs of letters ⟨d⟩⟨z⟩ and ⟨d⟩⟨ž⟩ whereas the unvoiced ones are specified with the single letter ⟨c⟩ and ⟨č⟩,

respectively. We replaced unvoiced affricates with the following pairs of the letters ⟨t⟩⟨s⟩ and ⟨t⟩⟨š⟩. The new grapheme ⟨ch⟩ was introduced. A full list of replacements: ⟨į⟩ → ⟨y⟩, ⟨į~⟩ → ⟨y~⟩, ⟨į^⟩ → ⟨y^⟩, ⟨ų⟩ → ⟨ū⟩, ⟨ų~⟩ → ⟨ū~⟩, ⟨ų^⟩ → ⟨ū^⟩, ⟨c⟩ → ⟨t⟩⟨s⟩, ⟨č⟩ → ⟨t⟩⟨š⟩, ⟨ą~⟩ → ⟨a~⟩, ⟨ą^⟩ → ⟨a^⟩, ⟨ę~⟩ → ⟨e~⟩, ⟨ę^⟩ → ⟨e^⟩, ⟨i^⟩ → ⟨i`⟩, ⟨u^⟩ → ⟨u`⟩. The reduced set D containing 56 graphemes looks as follows:

D = {⟨a⟩, ⟨a`⟩, ⟨a~⟩, ⟨a^⟩, ⟨ą⟩, ⟨e⟩, ⟨e`⟩, ⟨e~⟩, ⟨e^⟩, ⟨ę⟩, ⟨ė⟩, ⟨ė~⟩, ⟨ė^⟩, ⟨i⟩, ⟨i`⟩, ⟨i~⟩, ⟨y⟩, ⟨y~⟩, ⟨y^⟩, ⟨o⟩, ⟨o`⟩, ⟨o~⟩, ⟨o^⟩, ⟨u⟩, ⟨u`⟩, ⟨u~⟩, ⟨ū⟩, ⟨ū~⟩, ⟨ū^⟩, ⟨b⟩, ⟨d⟩, ⟨g⟩, ⟨p⟩, ⟨t⟩, ⟨k⟩, ⟨s⟩, ⟨š⟩, ⟨z⟩, ⟨ž⟩, ⟨f⟩, ⟨h⟩, ⟨ch⟩, ⟨j⟩, ⟨l⟩, ⟨l~⟩, ⟨m⟩, ⟨m~⟩, ⟨n⟩, ⟨n~⟩, ⟨r⟩, ⟨r~⟩, ⟨v⟩, ⟨ ⟩, ⟨?⟩, ⟨!⟩, ⟨.⟩} (4)

3.5. Separate accent marks

Finally, we decided to treat accent marks as separate graphemes. This seems somewhat irrational because inserting an accent mark shifts all the letters forward from their positions; however, it enables us to have an alphabet of minimal size. The set E containing 39 graphemes is as follows:

E = {⟨a⟩, ⟨ą⟩, ⟨e⟩, ⟨ę⟩, ⟨ė⟩, ⟨i⟩, ⟨į⟩, ⟨y⟩, ⟨o⟩, ⟨u⟩, ⟨ų⟩, ⟨ū⟩, ⟨b⟩, ⟨d⟩, ⟨g⟩, ⟨p⟩, ⟨t⟩, ⟨k⟩, ⟨c⟩, ⟨č⟩, ⟨s⟩, ⟨š⟩, ⟨z⟩, ⟨ž⟩, ⟨f⟩, ⟨h⟩, ⟨j⟩, ⟨l⟩, ⟨m⟩, ⟨n⟩, ⟨r⟩, ⟨v⟩, ⟨~⟩, ⟨^⟩, ⟨`⟩, ⟨ ⟩, ⟨?⟩, ⟨!⟩, ⟨.⟩} (5)

Everything mentioned above is summarized in Table 1.

Table 1. Summary of the input sets

Set	Uppercase letters	Separate accent marks	Accented letters	Reduced set of accented letters
A	+	–	–	–
B	–	–	–	–
C	–	–	+	–
D	–	–	+	+
E	–	+	–	–

Sample sentence written using all five input sets is given in Table 2. Accent marks are put above the letter where they are treated as a single grapheme.

Table 2. Sample sentence written using input sets A-E

Set	Sample sentence
A	*Lietuvos Respublikos įstatymai.*
B	*lietuvos respublikos įstatymai.*
C	*lietuvõs respùblikos įstãtymai.*
D	*lietuvõs respùblikos ystãtymai.*
E	*lietuvo~s respu`blikos įsta~tymai.*

4. Data and training conditions

Data for training were taken from the project LIEPA (Laurinčiukaitė et al., 2018). One of the aims of project LIEPA was to create speech corpora for development of speech recognition and synthesis. Databases of four voice talents (two male and two female) were created for speech synthesis. We chose the single young female voice for our experiments because it has the highest pitch, making it easier to detect audible speech distortions. Data were recorded in a professional studio and down-sampled to 16 kHz 16 bit mono. The total duration of the recordings was over 3.5 hours containing 5,034 phrases, more than 200,000 characters. The texts representing the recordings already contained accent marks, so no extra effort was required to create the input sets C-E, whereas the accent marks were simply removed when the input sets A and B were created. The texts did not require any preprocessing or data cleaning. Data were prepared according to the LJSpeech format (Ito and Johnson, 2017), i.e., a pipe-delimited two column text file with the name of a sound file and the text representing the sound. See Fig. 2.

```
...
0007.wav|lietuvo~s respu`blikos įsta~tymai.
0008.wav|nors ir vi`siškai nereikali`nga.
0009.wav|gyve^ntojai pradė^jo protestu^oti.
0010.wav|kad jauni`mas netu`ri sa`vo lė^šų.
0011.wav|ką~ rei~kia kei~sti dary^ti kitai~p?
...
```

Figure 2. The piece of sample input file in LJSpeech format encoded using set E.

Tacotron 2 (Shen et al., 2018) together with the vocoder WaveGlow (Prenger et al., 2018) is a state-of-the-art end-to-end speech synthesis model. Tacotron 2 is a recurrent sequence-to-sequence feature prediction network which can be trained to predict a sequence of mel spectrogram frames from an input character sequence. I.e., the neural network is initialized with random numbers, the text is converted to an internal encoding and propagated forward through the network, the result is compared with the target result, the error is calculated and propagated back through the neural network to adjust the weights of the network. By repeating this process many times with all the available examples, the network weights converge to values that can predict the spectrum from the text.

WaveGlow is a flow-based network which generates time-domain waveform samples from mel-spectrograms. The WaveGlow network training procedure is analogous.

Package Tacotron 2 from https://github.com/NVIDIA/tacotron2/ and package WaveGlow from https://github.com/NVIDIA/waveglow/ were used. Seeking to speed up training with the help of GPU, both packages and all the scripts were moved to cloud-based platform “Google Colaboratory” (https://colab.research.google.com/).

The Tacotron 2 model was trained on all data we had (5,034 phrases) using all five input sets A-E, resulting in five trained models. Training was stopped after 300,000 steps (approximately 168 hours) for each model. For more details see (Antanavičius, 2021).

5. Evaluation procedure

To evaluate the acceptability of the synthesized speech, human listeners were involved. It was decided not to use physical metrics for assessment, which, while useful in some cases, should not be considered as substitutes for listener tests (Schmidt-Nielsen, 1995). The evaluation procedure was divided into two sessions, 27 listeners participated in the first session and 31 participants took part in the second session. Listeners were divided into 4 groups: 6, 5, 8, 8 listeners in the first session and 8, 8, 7, 8 in the second one.

A total of 80 short meaningful sentences containing only common Lithuanian words (no names or international words) were used in the evaluation. The length of a sentence was 4-7 words, the average sentence length was 5.5 words. These sentences were not used in model training.

Test sentences were encoded using input sets A-E, then they were converted to audio recordings using trained models. Because the sets C-E use accented text, an automatic accentuation algorithm (Kasparaitis, 2000) was used for this purpose. Sample test sentences encoded using set E see in Fig. 3.

```
...
fi`lmą nei~giamai įve^rtino dau~gelis kri`tikų.
gera` sveikata` didžia^usias žmogau~s tur~tas.
pasa^ulis yra` ma^rgas kai~p geny~s.
gy^dytojai pa`taria nerūky^ti ir daugiau~ judė^ti.
ano^t patarlė~s da^rbas la^imės šalti`nis.
ar ga~li da^r ko~ nors reikė^ti?
...
```

Figure 3. Sample test sentences encoded using set E.

The pair comparison test (Schmidt-Nielsen, 1995) was performed, i.e., the pair of the sounds obtained by synthesizing the same sentence with two different trained models was presented to the listener. The latter had to choose the most appropriate answer:

- The first sound is much better than the second one;
- The first sound is slightly better than the second one;
- Both sounds are of the same quality;
- The second sound is slightly better than the first one;
- The second sound is much better than the first one.

An online form with GUI elements was developed for this purpose (accessible at https://danieliusa.github.io/lit-tacotron2/, last accessed on November 18, 2022). The form looks like a list of tasks with two player pictograms on the left-hand side (see Fig. 4) and five radio buttons on the right-hand side (see Fig. 5). Later, the answers

obtained were converted into numerical form, i.e., the values -2, -1, 0, 1, 2, respectively. These numerical values were used to calculate the mean and standard error of the responses (see Table 5). One of the possible interpretations of the obtained averages could be as follows: if the obtained average is close to the value -2 – First much better, if it is close to -1 – First slightly better, if it is close to 0 – Same quality etc.

Figure 4. Left-hand side of the evaluation form.

Figure 5. Right-hand side of the evaluation form.

The evaluation procedure was divided into two sessions. During the first session two experiments were performed: the models trained with the input sets A and B were compared, then the models trained with the sets C and D were compared. On the basis of the results of the first session, two more experiments were carried out during the second session: a comparison of B and D, and a comparison of D and E.

As many as 20 pairs of synthesized sentences were used in each experiment. The pairs were formed so that each model was presented in the first and second place exactly 10 times. The relationship between the sentences, listener groups and experiments is summarized in Table 3.

Table 3. Relationship between the sentences, listener groups and experiments

Sentences	Group of listeners			
	1	2	3	4
1-20	*A & B*	*D & E*	*B & D*	*C & D*
21-40	*C & D*	*A & B*	*D & E*	*B & D*
41-60	*B & D*	*C & D*	*A & B*	*D & E*
61-80	*D & E*	*B & D*	*C & D*	*A & B*

Answers obtained are presented in Table 4. Answers of the last (most interesting) experiment (where sets D and E were compared) are visualized in Fig. 6.

Table 4. Answers obtained in the experiments

Experiment	First much better	First slightly better	Same quality	Second slightly better	Second much better
A & B	82	94	154	120	130
C & D	41	94	267	117	61
B & D	42	88	120	148	262
D & E	35	74	327	152	72

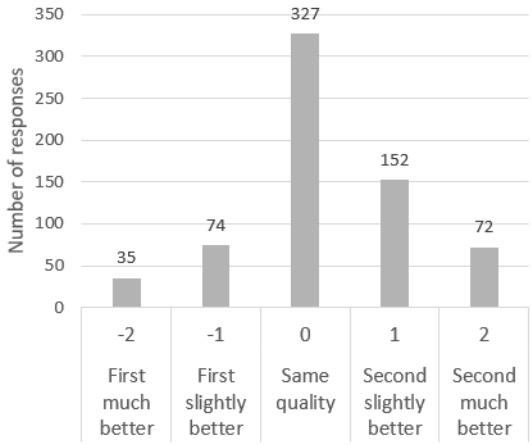

Figure 6. Number of responses in the experiment with the sets D and E.

6. Results

The mean value and the standard error were calculated for all listeners and all sentences. The results obtained are presented in Table 5. The winners are written in bold.

Table 5. Experimental results of relative subjective acceptability

Experiment	Description	Acceptability
A & B	*All letters vs.* ***Lowercase letters***	*0.21 ± 0.06*
C & D	*Accented letters vs.* ***Reduced set of accented letters***	*0.11 ± 0.04*
B & D	*Lowercase letters vs.* ***Reduced set of accented letters***	*0.76 ± 0.05*
D & E	*Reduced set of accented letters vs.* ***Separate accent marks***	*0.23 ± 0.04*

From Table 5, we can see that in the experiment with sets A and B, slightly better results were obtained using only lowercase letters. This is what was expected, because uppercase and lowercase letters always denote the same sounds. The difference is small, as the capitalization of the text makes up only a small percentage.

When using full and reduced sets of accented letters (C and D), slightly better results were obtained with reduced set because, as before, only letters denoting the same sounds were removed, and these letters make up a small fraction. From these two results, it can be summarized that replacing letters denoting the same sound by one letter and thus reducing the alphabet improves the results.

Comparing the sets of unaccented and accented letters (B and D), significantly better results were obtained using accented letters. This is because about 97 percent of the words are stressed correctly using accented letters, and only about 80 percent using non-accented letters, i.e., the neural network has been able to learn about 80 percent of word stress from plain text, and this percentage can be slightly improved by increasing the training data, but it still lags far behind the stress algorithm we developed. The percentage estimates provided are only approximate, as it is difficult to determine accurately from the hearing whether the word is stressed correctly.

Finally, a reduced set of accented letters and a set with separate accent marks (D and E) were compared. The set with separate accent marks gave slightly better results. It was a surprise to us. Let's take two words that are pronounced very similarly, only the place of the stress is different, for example, let's take the nominative and instrumental case of the word “parama” (support) and encode them using sets D and E:

Set D:
paramà
pãrama
Set E:
parama`
pa~rama

Using the set D, two letters have changed. Using the set E, the letters are the same, but all the letters except the first two have shifted, so different letters will be fed to the last five neural network inputs. We expected this would be a problem for the neural network, but it turns out that the neural network is capable of learning this.

7. Conclusions

The Tacotron 2 model was trained on Lithuanian speech data encoded with five different input alphabets. Acceptability testing by human listeners was performed. The main conclusion is that although the Tacotron 2 model can partially learn the complicated Lithuanian accentuation system from several hours of recordings and plain text, and increasing the amount of training data would increase that part even more, but putting accent marks to the text significantly improves the quality of the synthesized speech, so it seems like a more promising method. Reducing the size of the input alphabet also has a slight positive impact. Finally, putting accent marks into the text produced the best results as compared to using the accented letters.

References

Antanavičius, D. (2021). *Lithuanian speech synthesis using neural network package Tacotron 2* (Lithuanian), Bachelor thesis, Vilnius University, Vilnius, Lithuania.

Fong, J., Taylor, J., Richmond, K., King, S. (2019). A Comparison of Letters and Phones as Input to Sequence-to-Sequence Models for Speech Synthesis, *Proc. 10th ISCA Speech Synthesis Workshop*, 223-227, https://doi.org/10.21437/SSW.2019-40.

Ito, K., Johnson, L. (2017). The lj speech dataset, https://keithito.com/LJ-Speech-Dataset/, accessed on January, 2023.

Jiang, Z., Qin, F., Zhao, L. (2019). A Phoneme Sequence Driven Lightweight End-To-End Speech Synthesis Approach, *Journal of Physics: Conference Series*, 1267, 012052, https://doi.org/10.1088/1742-6596/1267/1/012052.

Kasparaitis, P. (2000). Automatic Stressing of the Lithuanian Text on the Basis of a Dictionary, *Informatica*, **11**(1), 19-40.

Kasparaitis, P. (2016). Evaluation of Lithuanian Text-to-Speech Synthesizers, *Studies about languages*, 28, 80-91. (in Lithuanian)

Kasparaitis, P., M. Beniušė (2019). Statistical Parametric Speech Synthesis of Lithuanian, 26th International Scientific Conference of Jonas Jablonskis "Linguistic Diversity in the Modern World: Language Power and Prestige", Lithuania, Vilnius, 3–4 October 2019, pp. 43-45.

Kastner, K., Santos, J. F., Bengio, Y., Courville, A. C. (2018). Representation mixing for TTS synthesis, http://arxiv.org/abs/1811.07240.

Laurinčiukaitė, S., Telksnys, L., Kasparaitis, P., Kliukienė, R., Paukštytė, V. (2018). Lithuanian Speech Corpus Liepa for Development of Human-Computer Interfaces Working in Voice Recognition and Synthesis Mode, *Informatica*, **29**(3), 487–498.

Li, B., Zhang, Y., Sainath, T., Wu, Y., Chan, W. (2018). Bytes are all you need: End-to-end multilingual speech recognition and synthesis with bytes, *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, 5621-5625, https://doi.org/10.1109/ICASSP.2019.8682674.

Liu, R., Sisman, B., Li, J., Bao, F., Gao, G., Li, H. (2020). Teacher-student training for robust tacotron-based TTS, *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 6274–6278, https://doi.org/10.1109/ICASSP40776.2020.9054681.

Liu, Y., Zheng, J. (2019). Es-Tacotron2: Multi-Task Tacotron 2 with Pre-Trained Estimated Network for Reducing the Over-Smoothness Problem, *Information*, **10**(4), 131, https://doi.org/10.3390/info10040131.

Melnik-Leroy, G. A., Bernatavičienė, J., Korvel, G., Navickas, G., Tamulevičius, G., Treigys, P. (2022). An Overview of Lithuanian Intonation: A Linguistic and Modelling Perspective. *Informatica*, **33**(4), 795-832.

Perquin, A., Cooper, E., Yamagishi, J. (2020). Grapheme or phoneme? An Analysis of Tacotron's Embedded Representations, https://arxiv.org/abs/2010.10694.

Ping, W., Peng, K., Gibiansky, A., Arik, S. O., Kannan, A., Narang, S., Raiman, J., Miller, J. (2018). Deep Voice 3: Scaling text-to-speech with convolutional sequence learning, *International Conference on Learning Representations (ICLR)*, 214-217.

Prenger, R., Valle, R., Catanzaro, B. (2018). Waveglow: a flow-based generative network for speech synthesis, https://arxiv.org/abs/1811.00002.

Radzevičius, A., Raudys, A., Kasparaitis, P. (2021). Speech Synthesis Using Stressed Sample Labels for Languages with Higher Degree of Phonemic Orthography, *in* Lopata A., Gudonienė D., Butkienė R. (eds) *Information and Software Technologies*, vol **1486**, Springer, Cham, pp. 378-387.

Schmidt-Nielsen, A. (1995). Intelligibility and Acceptability Testing for Speech Technology. *in* Syrdal, A., Bennett, R., Greenspan, S. (eds.), *Applied Speech Technology*, CRC Press, Boca Raton/ Ann Arbor/ London/ Tokyo, pp.195–232.

Shen, J., Pang, R., Weiss, R. J., Schuster, M., Jaitly, N., Yang, Z., Chen, Z., Zhang, Y., Wang, Y., Skerrv-Ryan, R. et al. (2018). Natural TTS Synthesis by Conditioning Wavenet on MEL Spectrogram Predictions. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4779–4783.

Staib, M., Teh, T., Torresquintero, A., Mohan, D., Foglianti, L., Lenain, R., Gao, J. (2020), Phonological features for 0-shot multilingual speech synthesis, in *Proc. Interspeech*, 2942–2946, http://dx.doi.org/10.21437/Interspeech.2020-1821.

Wang, Y., Skerry-Ryan, R., Stanton, D., Wu, Y., Weiss, R. J., Jaitly, N., Yang, Z., Xiao, Y., Chen, Z., Bengio, S. et al. (2017). Tacotron: A fully end-to-end text-to-speech synthesis model, in *Proc. Interspeech 2017*, 4006–4010, http://dx.doi.org/10.21437/Interspeech.2017-1452.

Yasuda, Y., Wang, X., Takaki, S., Yamagishi, J. (2019). Investigation of enhanced tacotron text-to-speech synthesis systems with self-attention for pitch accent language, *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 6905–6909, https://doi.org/10.1109/ ICASSP.2019.8682353.

Yasuda, Y., Wang, X., Yamagishi, J. (2020). Investigation of learning abilities on linguistic features in sequence-to-sequence text-to-speech synthesis, *Computer Speech & Language*, **67**, 101183. https://doi.org/10.1016/j.csl.2020.101183.

Zhang, J.-X., Ling, Z.-H., Dai, L.-R. (2018). Forward attention in sequence-to sequence acoustic modeling for speech synthesis, *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 4789–4793, https://doi.org/10.1109/ICASSP.2018.8462020.

Zhang, Y., Weiss, R. J., Zen, H., Wu, Y., Chen, Z., Skerry-Ryan, R. J., Jia, Y., Rosenberg, A., Ramabhadran, B. (2019). Learning to Speak Fluently in a Foreign Language: Multilingual Speech Synthesis and Cross-Language Voice Cloning, in *Proc. Interspeech*, 2080-2084, http://dx.doi.org/10.21437/Interspeech.2019-2668.

Authors' information

Pijus Kasparaitis is currently Associate Professor in the Institute of Informatics, Faculty of Mathematics and Informatic, Vilnius University, Lithuania. He received his honours diploma in Applied mathematics from Vilnius University in 1991 and PhD in 2001. He has more than thirty years of academic and research in the field of Computer Science. His areas of research are text-to-speech synthesis of Lithuanian and other areas of Computer Linguistics.

Danielius Antanavičius got a Bachelor's degree at Vilnius University (Faculty of Mathematics and Informatics) in 2021. Current research interest is text-to-speech synthesis.

Received January 27, 2023, revised May 13, 2023, accepted June 5, 2023