Building a Turkish Text-to-Speech Engine: Addressing Linguistic and Technical Challenges

Tuğçe Melike Koçak
NLP Technologies
Turkcell Technologies
Istanbul, Turkey
Email: tugce.kocak@turkcell.com.tr

Mehmet Büyükzincir
NLP Technologies
Turkcell Technologies
Istanbul, Turkey
Email: mehmet.buyukzincir@turkcell.com.tr

***Abstract*—The pursuit of end-to-end text-to-speech (TTS) systems has resulted in generating natural sounding speech. However, due to the requirements of high quality data and pronouncing dictionary, certain languages may face obstacles in achieving comparable results. In this paper, we present innovative solutions to these challenges in the context of Turkish TTS applications. To demonstrate the effectiveness of our approach, we train Fastspeech2 and Tacotron models and evaluated them using mean opinion score (MOS) measure and Speaker Encoder Cosine Similarity (SECS). Evaluation results show that FastSpeech2 model performed well than Tacotron, also our approaches for the dataset is improved the performance of FastSpeech2.**

***Index terms*—text-to-speech, Turkish speech corpus, speech synthesis**

I. Introduction

The field of text-to-speech (TTS) has made significant advancements in recent years, resulting in the creation of TTS models that are capable of producing natural and expressive speech synthesis [1], [2], [3], [4]. However, the latest TTS models, such as Tacotron2 [2], and Fastspeech2 [4] require around 25 hours of high-quality data for training. Since the data collection for TTS is tedious, costly, and error-prone, there is potential for improvement in this area.

To certify effective TTS training, it is a must to use a clean dataset that is free from background noise [5]. Although some open-source datasets are available for languages such as English [6], Mandarin [7], and Korean [8], there are no open-source TTS datasets currently available for Turkish. Additionally, the Turkish language lacks a phoneme dictionary as it is pronounced as it is written. However, certain letters such as /a/ and /i/ exhibit changes in stress and length depending on their position within a word. Turkish also contains heteronyms, which are words that share the same spelling but have distant pronunciations. For instance, the Turkish words for "of course" and "naturally" are both spelled as "tabii", but are pronounced differently [9]. While a TTS model could learn from a few samples, it may face confusion when presented with several similar words, as demonstrated in the previous example.

Another challenge the Turkish language poses is changing the pronunciation of future and past tense suffixes to make it easier to speak in everyday language. For example, while the sentence "I will come" is written as "geleceğim", it is spoken as "gelicem" in daily life. Speakers tend to misread it unconsciously since this is a very common expression. Additionally, when dubbing movies or animations, it is desired to pronounce these suffixes as they are spoken in everyday language. However, such misreadings cause the model to learn and synthesize the suffixes "-ceğim", "-cağım" as "-icem", "-ıcam" which is an undesired outcome.

Creating high-quality TTS datasets poses a challenge for speakers as the recordings need to be free of any noise [10]. While reading the text, the speaker should avoid making any sounds with their mouth, like breathing or lip-smacking. Any such noises that are present should be removed during editing in the studio. These sounds and other similar noises can result in unwanted sounds during pauses. Additionally, manually listening to the entire dataset for these noises, and inconsistency between text and audio can be time-consuming and labor intensive, however, is essential for the overall quality of synthesized audio.

To address all the issues mentioned above, first, we added additional characters for the character set and modified the texts accordingly. Then, we employed Kaldi automatic speech recognition (ASR) model [11] to develop a system that could identify misspellings and mismatches between the audio and text. Additionally, for removing unwanted breathing sounds, we used the energy levels and durations of letters.

II. Dataset

The success of any research depends on the quality of the data used. Therefore, it is essential to have a complete understanding of the data collection process and the dataset details. This section aims to provide an overview of the data collection process and the dataset details that we have used in our study. We will give details of the speaker, the methods used to collect the data, topics of texts. Additionally, we will provide information about the dataset's technical specifications, such as the number of sentences, words, and number of hours of audio. By detailing the data collection process and the dataset details, we hope to provide readers with a clear understanding of the data used in our study.

A. Text collection

To generate our dataset, we used various articles about topics such as politics, health, life, and sports to increase word and phoneme coverage. In total, we collected over 45705 sentences of different lengths.

979-8-3503-3959-8/23/$31.00

One of Turkey's leading voice actresses was chosen to voice the selected sentences. The speaker is 57 years old with 28 years of work experience. The recording started in late 2018, and it took two years. The recordings took place in a studio. Since the recording process took many hours and multiple days, there is a variance in the speaker's voice throughout the dataset. This was one of the challenges we had to overcome.

The speaker was instructed to narrate the sentences with a neutral tone and pace. Additionally, she was asked to follow Turkish grammar rules, pause on commas, ending the sentences according to meaning. The recordings were sampled at 48 kHz and stored using 24 bits/sample. In total, the speaker read around 60 hours of text.

B. Audio-to-text alignment

To get more accurate alignment results for training of FastSpeech2, first, we trained our Kaldi ASR model with 700 college students and 500 hours of data. We created a channel in the instant messaging platform BiP [12], the speakers were given unique keys, and they registered to the channel. Each speaker has given a text, and they were asked to record their voices. All speakers were informed of the data collection and permitted to use their audio. In addition to the alignment, we used the Kaldi ASR model to detect misspellings, to add commas where the speaker pauses, and to identify mismatches between text id and text. This usage of Kaldi ASR would be explained in the next chapter.

C. Dataset specifications

The overall statistics of the dataset can be found in Table I. In total, the dataset contains 45705 sentences and 63 hours of audio. The distribution of sentence duration and sentence length are shown in Fig 1. The dataset is arranged as follows. The speaker's audio recordings and corresponding transcript text filenames are the same except for file extensions. Text files are stored using UTF-8 encoding due to the nature of Turkish characters such as "ç", "ğ", "ü", "ı", "ş". The recordings were downsampled to 22 kHz and stored as 16 bits/sample before the alignment and training process.

TABLE I. TURKISH TTS DATASET SPECIFICATIONS

	Category	Speaker
Sentence duration	Total	63.7 hr
	Mean	5 sec
	Min	1.9 sec
	Max	15.8 sec
Words	Total	402,410
	Mean	8.8
	Min	1
	Max	41
	# of unique words	83,660

III. MODEL ARCHITECTURE

In any research, the quality of the data processing is critical to obtaining accurate and successful results. Therefore, this section aims to describe the novelties in data processing that we have employed in our dataset. These novelties include the use of durations of letters to create a new symbol list,

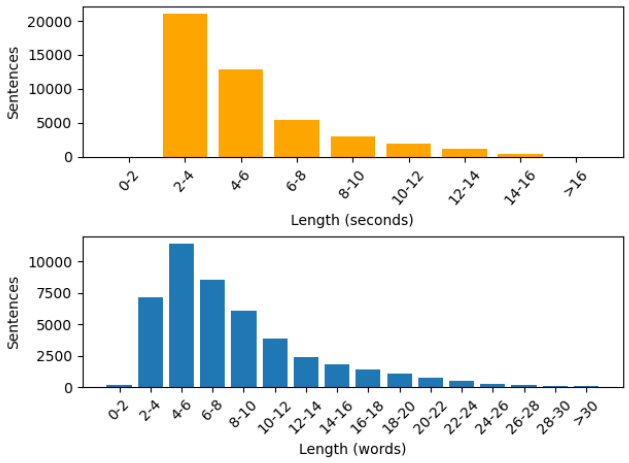

Fig. 1. Sentence duration and length distributions for speaker

a verification system to automatically detect errors between audio and transcript, and techniques that have enabled us to handle unwanted sounds. By discussing our novelties in data processing, we hope to provide insights into our model architecture and demonstrate how our approaches have led to our outcomes.

A. Additional Symbol List

The Turkish language has 29 letters alphabet and is one of the phonetic languages, however, certain vowels including /a/, /i/ require additional notations. To address this issue, we examined the letters' average, range, max, and min durations in our dataset to provided these notations. If there was a wide range between the length of durations, we changed these letters with different notations in those specific words. Furthermore, besides the duration of the letter, the stress of /a/ sound is different in some words. Such as "hikaye", "dükkan", "rüzgar". Using Turkish Language Association (TDK) website [13], we created a word list includes these samples. We manually modified these words in our transcript. We also created a dictionary to make these changes when we synthesize sentences containing these words.

B. Verification System

The effectiveness of a TTS model depends on the correctness of the dataset. That's why manually listening and detecting misspellings, misspoken words or inconsistencies between text and audio is a crucial step. Since the process is time-consuming and mundane, we trained a Kaldi model with 500 hundred hours of audio and created a lexicon corpus with our transcript of what the speaker ought to read. We set a threshold of 1% for each sentence, if this threshold is exceeded, we eliminate the sentence. After the elimination process, errors were searches manually in selected sentences. The error rate is 1% after the dataset is passed through the verification system.

C. Noise Eliminating

Even though all the audio had been recorded in the studio, there is some mouth noise such as breathing in our dataset.

Especially, whenever there is long sentences with multiple commas, the speaker exhaled in mid-sentence. While that is not a problem with synthesizing with Tacotron [1] using Griffin-Lim [14], it is a problem for FastSpeech2 with Hifi-gan [15].

To solve the noise problem in the dataset, we first got the energy of each frame of each sentence, then determined a threshold for a sound whether it is a letter or random noise looking through the dataset. After setting the threshold, we got the duration of each pause in the sentences. Replaced the pauses with silences with 50 ms margin at the start and end of the pauses. In total, we modified 66% of the all audio.

IV. Experiments and Results

In this section, we will describe the models, experiments, and results that we have utilized with our dataset. We will provide details about the models we have used, including their configurations, used vocoders. Additionally, we will explain the experiments we have conducted to test these models and the results we have obtained from these experiments.

A. TTS experiments

1) Experimental Setup: We conducted 3 different experiments. We want to see the performance differences between Tacotron and FastSpeech2 models with our dataset. Also, we want to see how well our approaches perform. We removed one word sentences from our dataset. We trained both FastSpeech2 and Tacotron with around 60 hours of data. We randomly split our data into 2 sections, training and testing. We downsampled the audio frame rate to 22050 kHz.

We trained both FastSpeech2 models with a new letter set to improve the pronunciation. To improve the quality of synthesized audio, we use noise eliminating technique mentioned above. Also, we used a verification system to verify the dataset for both FastSpeech2 models. However, to see the effectiveness of noise elimination, we only apply it to the dataset that we trained the FastSpeech2-Preprocessed model. For both Fastspeech 2 models, alignment is done with our Turkish Kaldi model.

2) TTS Model Configurations: Our Fastspeech 2 models consist of 8 feed-forward Transformer (FFT) blocks in the encoder and decoder. For each FFT block, the hidden size of the self-attention is set to 512. The number of attention heads is set to 8 and the kernel sizes of the 1D convolution in the 2-layer convolutional network after the self-attention layer are set to 9 and 1, with input and output sizes 512/1024 and 1024/512 respectively for the first and second layer. For the variance predictor, the kernel sizes of the 1D convolution are set to 3, with input and output sizes 512 for both layers, and the dropout rate is set to 0.5. The model training were done on seven NVIDIA V100 GPUs with batch size of 64. We trained FastSpeech models for 300k steps. Both FastSpeech models used Hifi-gan as a vocoder.

Tacotron is an RNN-based model that uses the CBHG (1-D convolution, bank highway network, and bidirectional GRU) module. Our Tacotron model consists of a fully-connected (FC)-512-ReLU/FC-256-ReLU encoder and decoder pre-net. The dropout rate is set to 0.5. Tacotron training was done on NVIDIA v100 GPU with batch size of 128. Griffin-Lim vocoder was used for Tacotron.

3) Vocoder Model Configurations: The TTS models generate a mel-spectrogram of desired audio. A trained vocoder synthesizes speech with generated mel-spectrogram [4]. As a vocoder we used Hifi-gan. For training the Hifi-gan, the sampling rate is set to 22.050 kHz with FFT size of 1024. The frame size and hop size are set to 1024 and 256 respectively. The model trained for 500k steps. Figure 2 shows generated sample compared to the ground truth with 500k steps. The training was on one NVIDIA V100 GPU with batch size of 16. The training took 86 hours.

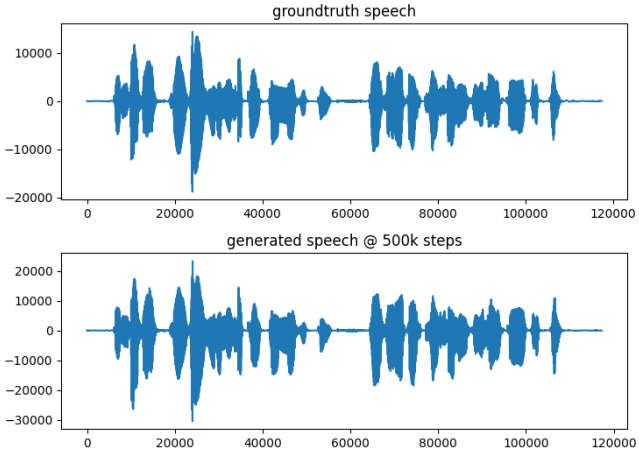

Fig. 2. A graph of sample audio generated at 500k steps with Hifi-gan compared to ground truth.

B. Experiment Results

We randomly selected 25 text sentences with various lengths from the test set. We generated these 25 text samples using 3 models to be evaluated. In total 450 audio samples by generated and evaluated by 50 individuals. Each audio was rated by 6 different listeners. Each listener independently scored each sample based on a 5-point Likert scale score. A test channel was opened on BiP so that each user could evaluate under the same conditions and test without knowing what they scored. User IDs were defined on the test system, allowing everyone to score 12 different audios.

TABLE II. MOS and SECS with 95% confidence intervals for all outputs.

Model	**MOS**	**SECS**
Ground truth	4.53 $\pm$ 0.12	0.88
Tacotron	3.72 $\pm$ 0.16	0.86
FastSpeech2	4.1 $\pm$ 0.16	0.83
FastSpeech2-Preprocessed	4.39 $\pm$ 0.13	0.85

The evaluation results are given in Table II. According to the MOS results, all three models passed the 3.5 score which is a good quality in terms of industry standards. The ground truth expectedly achieved the highest performance, however FastSpeech2 with pre-processed data closely followed the ground truth, then FastSpeech2 and Tacotron model. As we expected preprocessed data performed best, that's because

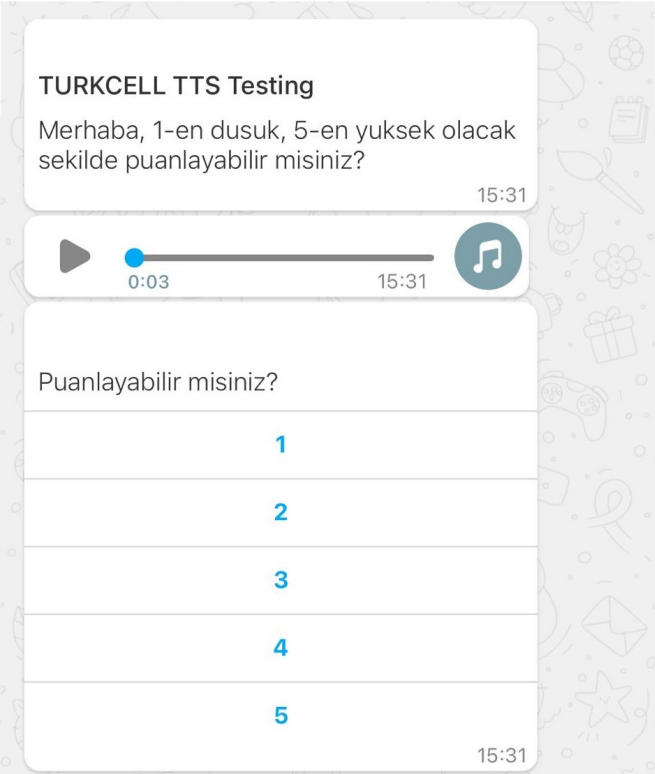

Fig. 3. An example of interaction with the BiP channel during an evaluation session.

breathing sounds in the dataset cause the mechanical sounds in the pauses of synthesized audio. As we can see in the Table III, noise is the highest in FastSpeech2 model. Although we did not eliminate the breathing sounds in the dataset for Tacotron, synthesized audio for Tacotron is a more natural-sounding breathing sound. However overall audio quality does not compute with FastSpeech2 models.

To analyze the similarity between ground truth and synthesized voices, we calculated the Speaker Encoder Cosine Similarity (SECS) [16] between reference audio from the speaker and synthesized voices. The score range between -1 to 1, the highest score indicates higher similarity. As it can be seen in Table II all model scores are similar to each other. However, Tacotron is slightly higher than others.

TABLE III. MANUAL ANALYSIS OF MOST COMMON ERRORS.

Error types	**Tacotron**	**FastSpeech2**	**FastSpeech2-Preprocessed**
Misprounounced words	1	1	1
Noise	1	7	2
Incomplete sentences	3	0	0
Long pauses	0	0	0
Skipped words	0	0	0

V. CONCLUSION

In this paper, we introduced problems commonly faced in training and synthesizing TTS. Especially, languages without proper phoneme dictionaries such as Turkish what to look for when creating a dataset, and how to remove silences, and creating a new symbol list. According to the experimental results, our proposed solutions helped successfully generated speech. Our proposed solutions, based on our own dataset, have shown efficient results in generated speech. These solutions can be modified and implemented in other languages and datasets as well. In the future, we plan to explore the possibilities of multi-speaker TTS and emotional TTS for Turkish.

REFERENCES

[1] Y. Wang, R. Skerry-Ryan, D. Stanton, Y. Wu, R. J. Weiss, N. Jaitly, Z. Yang, Y. Xiao, Z. Chen, S. Bengio *et al.*, "Tacotron: Towards end-to-end speech synthesis," *arXiv preprint arXiv:1703.10135*, 2017.

[2] J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. Skerrv-Ryan *et al.*, "Natural tts synthesis by conditioning wavenet on mel spectrogram predictions," in *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2018, pp. 4779–4783.

[3] Y. Ren, Y. Ruan, X. Tan, T. Qin, S. Zhao, Z. Zhao, and T.-Y. Liu, "Fastspeech: Fast, robust and controllable text to speech," *Advances in neural information processing systems*, vol. 32, 2019.

[4] Y. Ren, C. Hu, X. Tan, T. Qin, S. Zhao, Z. Zhao, and T.-Y. Liu, "Fastspeech 2: Fast and high-quality end-to-end text to speech," *arXiv preprint arXiv:2006.04558*, 2020.

[5] Q. Hu, E. Marchi, D. Winarsky, Y. Stylianou, D. Naik, and S. Kajarekar, "Neural text-to-speech adaptation from low quality public recordings," in *Speech Synthesis Workshop*, vol. 10, 2019.

[6] K. Ito and L. Johnson, "The lj speech dataset. 2017," *URL https://keithito. com/LJ-Speech-Dataset*, 2017.

[7] H. Zen, V. Dang, R. Clark, Y. Zhang, R. J. Weiss, Y. Jia, Z. Chen, and Y. Wu, "Libritts: A corpus derived from librispeech for text-to-speech," *arXiv preprint arXiv:1904.02882*, 2019.

[8] K. Park, "Kss dataset: Korean single speaker speech dataset," 2018.

[9] A. Göksel and C. Kerslake, *Turkish: A comprehensive grammar*. Routledge, 2004.

[10] R. Zandie, M. H. Mahoor, J. Madsen, and E. S. Emamian, "Ryanspeech: A corpus for conversational text-to-speech synthesis," *arXiv preprint arXiv:2106.08468*, 2021.

[11] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, and P. Schwarz, "The kaldi speech recognition toolkit. ieee 2011 workshop on automatic speech recognition and understanding. ieee signal processing society," 2011.

[12] B. BiP İletişim Teknolojileri ve Dijital Servisler A.Ş, https://bip.com/tr/, [Online].

[13] T. Turkish Language Association, https://sozluk.gov.tr, [Online].

[14] D. Griffin and J. Lim, "Signal estimation from modified short-time fourier transform," *IEEE Transactions on acoustics, speech, and signal processing*, vol. 32, no. 2, pp. 236–243, 1984.

[15] J. Kong, J. Kim, and J. Bae, "Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis," *Advances in Neural Information Processing Systems*, vol. 33, pp. 17 022–17 033, 2020.

[16] E. Casanova, C. Shulby, E. Gölge, N. M. Müller, F. S. de Oliveira, A. C. Junior, A. d. S. Soares, S. M. Aluisio, and M. A. Ponti, "Sc-glowtts: an efficient zero-shot multi-speaker text-to-speech model," *arXiv preprint arXiv:2104.05557*, 2021.