VILNIUS UNIVERSITY

FACULTY OF MATHEMATICS AND INFORMATICS

DATA SCIENCE STUDY PROGRAMME

Master's thesis

Training Data Selection Strategies for Multi-Speaker Text-to-Speech Synthesis in Lithuanian

Mokymo duomenų parinkimo strategijos šnekos sintezės modelio apmokymui lietuvių kalba, naudojant kelių kalbėtojų balsus

Aleksandr Jan Smoliakov

Supervisor : Dr. Gerda Ana Melnik-Leroy

Scientific advisor : Dr. Gražina Korvel

Reviewer : pedagogical/scientific title Name Surname

Vilnius
2025

List of Figures

List of Tables

Contents

Introduction

The goal of creating machines that can speak like humans has captivated researchers for centuries. One of the earliest known attempts dates back to the 18th century, with Wolfgang von Kempelen's mechanical "speaking machine" [1] that utilized a bellows-driven lung and physical models of the tongue and lips to produce rudimentary speech sounds.

Over the centuries, understanding of human speech and advancements in technology have driven significant progress in this field. Today's state-of-the-art Text-to-Speech (TTS) systems, dominated by end-to-end (E2E) neural models such as Tacotron 2 [2] and Glow-TTS [3] have achieved highly natural speech with unprecedented acoustic quality. Notably, these E2E systems have unified the entire synthesis process into one or two neural networks, learning to map text inputs (graphemes or phonemes) directly to audio outputs, and eliminating the need for complex multi-stage pipelines.

These advancements have transformed TTS from a niche area of academic curiosity to become essential tools in daily life. High-quality speech synthesis now powers popular virtual assistants [4] and navigation systems, serving as a primary interface for human-computer interaction. More importantly, it plays a critical role in accessibility, enabling screen readers for the visually impaired [5], providing a voice for the non-verbal, and democratizing access to digital information [6].

However, the transition to deep learning techniques has introduced a new dependency: data. While modern neural TTS models are capable of producing remarkably natural speech, they are inherently "data-hungry". The common recommendation is to use at least 10 to 20 hours of high-quality recorded speech from a single speaker to achieve good synthesis quality. The majority of existing research and commercial TTS systems focus on high-resource languages like English, where professionally recorded single-speaker datasets (e.g., LJSpeech [7] with 24 hours of speech) are readily available.

However, less-resourced languages, such as Lithuanian, often lack such extensive datasets, and available corpora tend to be crowd-sourced with multiple speakers contributing small amounts of data each.

Liepa 2 [8] is a Lithuanian speech corpus, released in 2020, that contains 1000 hours of annotated speech; however, this data is distributed across 2621 speakers, with most speakers contributing under 30 minutes each. The top speaker has only around 2.5 hours of recorded speech, well below the standard recommendation for single-speaker TTS training.

Having such a fragmented dataset presents an engineering challenge. Using a single speaker's data from such a corpus yields insufficient data for high-quality synthesis. On the other hand, training on the entire 1000-hour corpus is a time-consuming and computationally prohibitive process, especially in the scope of a master's thesis. Therefore, multi-speaker TTS models must be utilized to aggregate data from multiple speakers.

This necessitates a choice between two competing strategies under a fixed computational budget: prioritizing **breadth** (many speakers with little data each) or **depth** (fewer speakers with more data each). Consequently, the question that arises is: what is the optimal strategy for sampling multi-speaker data to maximize synthesis quality?

Current literature generally advocates for transfer learning from large pre-trained models to improve low-resource TTS performance. However, there is a lack of research specifically addressing the internal composition of the training set for morphologically complex languages like Lithuanian.

In order to address this gap, this thesis investigates the optimal data selection strategy for training Lithuanian multi-speaker TTS models under a fixed data budget. It aims to measure how varying the balance between training dataset breadth (number of speakers) and depth (duration per speaker) affects the synthesis quality of multi-speaker TTS models. The hypothesis is that depth is a more critical factor, and quality will degrade as the number of speakers increases, given a constant total training duration.

To achieve this aim, the following research objectives are defined:

- Create three distinct TTS training datasets (with 30, 60, 180 speakers) that maintain a constant total duration but vary in speaker distribution.
- Implement a text processing pipeline capable of handling Lithuanian accentuation and grapheme-based input.
- Train two distinct acoustic model architectures (one autoregressive, one non-autoregressive) on each of the created datasets.
- Evaluate the models using objective metrics and conduct a subjective Mean Opinion Score listening test with native Lithuanian speakers to assess naturalness.

In this study, the scope is exclusively focused on the Lithuanian language and the Liepa 2 speech corpus. It investigates a fixed total training data size of 22.5 hours to simulate a realistic resource budget. The models are limited to Tacotron 2 and Glow-TTS architectures within the Coqui TTS framework, and a pre-trained HiFi-GAN vocoder is used to isolate the performance of the acoustic models.

The remainder of this thesis is organized as follows: **Literature review** presents a literature review of relevant concepts in digital signal processing, neural TTS architectures, and the specific challenges of Lithuanian TTS. **Methodology** describes the data selection, model configurations, and the experimental design. **Results** presents the findings of the objective and subjective synthesis quality evaluations, analyzing the failure modes of the models. Finally, **Conclusion** summarizes the key findings and offers potential recommendations for future research directions in low-resource TTS synthesis.

1 Literature review

This section analyzes the key concepts related to text-to-speech synthesis, and reviews the evolution from traditional methods to modern neural architectures.

1.1 Digital representation of audio

Speech, or sound in general, is a continuous pressure wave that propagates through a medium, such as air. The key properties of sound waves include frequency (pitch), amplitude (loudness), and phase.

Converting continuous sound waves into a digital format suitable for computer processing involves two main steps: sampling and quantization.

Sampling is the process of measuring the amplitude of the sound wave at regular time intervals. The rate at which these samples are taken is called the sampling rate. According to the Nyquist-Shannon [9] sampling theorem, accurate reconstruction of a continuous signal requires a sampling rate that is strictly greater than twice the highest frequency present in the signal. Frequencies in the range between 300 Hz and 3400 Hz contribute most to human speech intelligibility and recognition [10]. In text-to-speech applications, common sampling rates for audio are 22.05 kHz and 24 kHz, which can capture frequencies up to approximately 11 kHz and 12 kHz, respectively.

Quantization (also known as bit depth) is the mapping of continuous amplitude values to discrete levels for digital representation, which determines the precision of the representation. Common bit depths for audio are 16-bit and 24-bit formats. A visual representation of both sampling and quantization is provided in Figure 1.

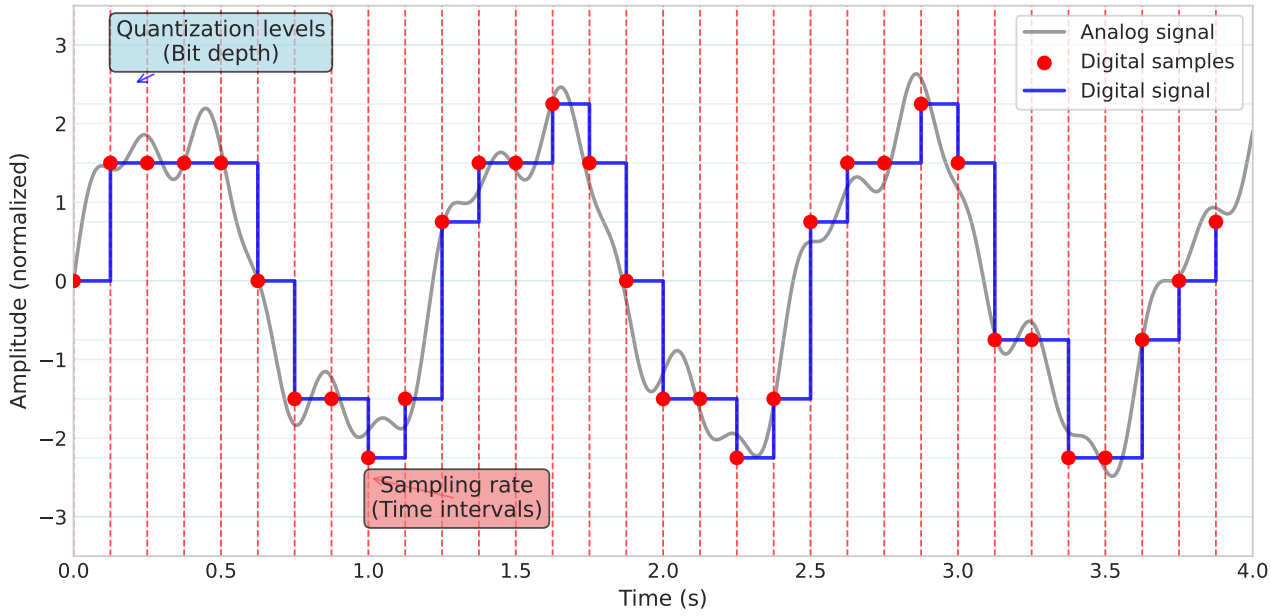


Figure 1. *Visual representation of Analog-to-Digital conversion. The continuous grey line represents the analog signal. The vertical lines represent the* ***sampling rate*** *(time intervals), and the horizontal grid lines represent* ***quantization levels*** *(bit depth).*

Pre-emphasis is a high-frequency filtering technique applied to audio signals before further processing. Natural speech signals tend to have more energy in the lower frequencies, with a gradual drop-off towards higher frequencies (typically around -6 dB per octave). Pre-emphasis compensates for this spectral tilt by boosting high frequencies using a first-order high-pass filter, which is defined as:

$$y[n] = x[n] - \alpha x[n-1] \tag{1}$$

where $y[n]$ is the pre-emphasized signal, $x[n]$ is the original signal, α is the pre-emphasis coefficient (typically between 0.9 and 1.0, and often set to 0.97), and n is the sample index.

This transformation balances the frequency spectrum, improving the signal-to-noise ratio for higher frequencies and preventing the model from optimizing only for low-frequency components.

1.2 Time-Frequency Analysis

1.2.1 Fourier Transform

Fourier Transform (FT) is a mathematical technique that transforms a time-domain signal (such as an audio waveform) into its frequency-domain representation. The signal is decomposed into a sum of sine and cosine waves at various frequencies, each with a specific amplitude and phase. This allows us to analyze the frequency content of the signal.

Short-Time Fourier Transform (STFT) [11] extends the FT by applying it to short, overlapping segments (frames) of the signal. This transformation provides a time-frequency representation, showing how the frequency content of the signal changes over time.

In TTS applications, the STFT is computed by dividing the audio signal into short frames (usually, 20–50 ms) with a certain overlap (usually, 50–75%) between frames, windowed by a Hamming or Hann function to reduce the spectral leakage.

1.2.2 Spectrogram and Mel-spectrogram

The spectrogram is a visual representation of the STFT, displaying frequency on the vertical axis, time on the horizontal axis, and amplitude represented by the color intensity.

However, the human ear does not perceive frequencies linearly — it is more sensitive to lower frequencies than higher ones. To mimic this perceptual characteristic, the Mel scale [12] maps linear frequency f (in Hz) to a perceptual scale m (in Mels) using the following formula:

$$m = 2595 \cdot \log_{10}\left(1 + \frac{f}{700}\right) \tag{2}$$

Mel-spectrograms are computed by applying a Mel filterbank of overlapping triangular filters (or kernels) to the magnitude spectrogram obtained from the STFT. This results in a compressed representation of the audio signal that aligns more closely with human auditory perception. Such Mel-spectrograms are commonly used as input features for modern TTS systems. The differences between the raw waveform, the standard spectrogram, and the Mel-spectrogram are illustrated in Fig-

ure 2. Note how the Mel-spectrogram has a higher resolution in the lower frequencies, where the majority of the speech energy is concentrated.

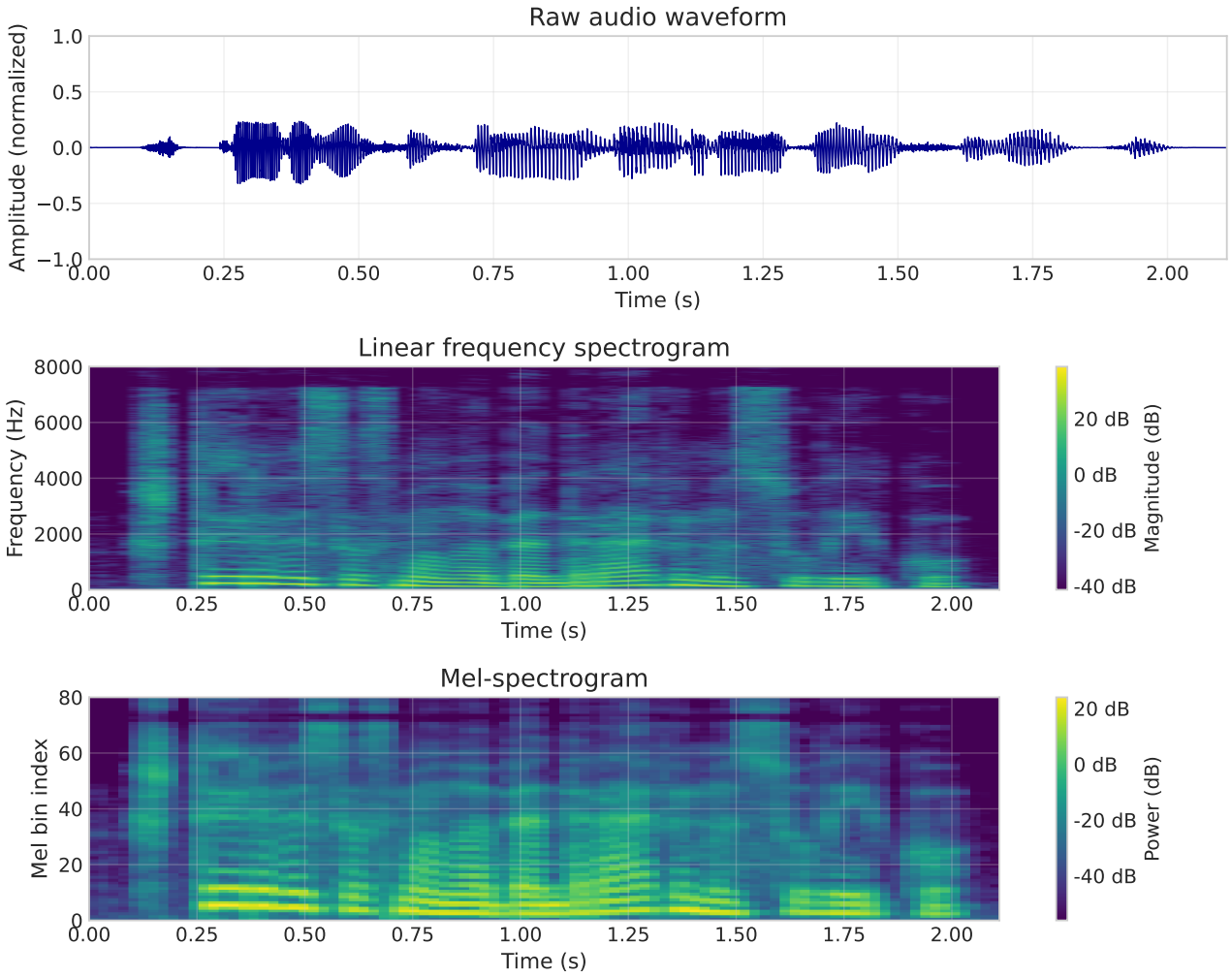


Figure 2. *Raw audio waveform (top), its linear frequency spectrogram (middle), and Mel-spectrogram (bottom) representations for the utterance "Štai ir visas mano bendravimas su vaiku".*

1.3 Linguistic Representation (Text Processing)

In TTS systems, the input text must be pre-processed and converted into a suitable linguistic representation that the synthesis model can use. The main goal is to map the raw sentences into a sequence of symbols that can be more closely mapped to the acoustic features of speech.

Although theoretically an end-to-end TTS model could learn to map raw text directly to audio, in practice, pre-processing the text makes the model convergence easier and improves the quality of the synthesized speech.

This process typically involves several steps, such as text normalization, grapheme-to-phoneme conversion, and possibly prosody prediction.

1.3.1 Text normalization

Text normalization [13] is the process of converting raw written text with non-standard words into a more standardized "spoken" form. Typical steps include expanding abbreviations (e.g., expanding "Dr." to "Doctor"), punctuation removal, number normalization (e.g., converting "123" to "one hundred twenty-three"), and lowercasing.

As an example, the input text "Dr. Smith has 2 cats." could be normalized to "doctor smith has two cats".

Text normalization helps reduce the variability and complexity in the input text, decreases the number of unique symbols, and removes the ambiguities that could confuse the TTS model. The resulting normalized text is not only easier for the model to process, but can also be further converted into phonemes, which provide an even closer representation of the spoken language.

1.3.2 Graphemes vs. Phonemes

Text-to-speech systems use a discrete input representation derived from text, generally divided into grapheme-based or phoneme-based sequences.

Grapheme-based models ingest raw character sequences (orthography). This approach simplifies the inference pipeline by eliminating the dependency on external grapheme-to-phoneme (G2P) converters. However, it forces the model to implicitly learn pronunciation rules from data, which can be a significant challenge for languages with complex orthographies or inconsistent grapheme-to-phoneme mappings (e.g., "read" /riːd/ vs. "read" /rɛd/, depending on the tense).

In contrast, the phoneme-based approach uses a phonetic transcription of the text, typically in the International Phonetic Alphabet (IPA) or ARPABET form. By resolving pronunciation ambiguities prior to training, phonemes provide a more direct mapping to acoustic features, simplifying the model's task of learning alignment. The downside is that this approach requires an external G2P conversion step [14]. Additionally, errors in the G2P conversion can propagate to the TTS model, affecting the quality of the synthesized speech.

There is another approach that augments the grapheme-based representation with explicit lexical stress markers or diacritics (e.g., tilde, acute, grave accents). This intermediate method helps the model disambiguate pronunciation of homographs and easier learn prosodic patterns without requiring a full phonetic transcription, particularly in languages where stress placement alters meaning.

1.3.3 Challenges in Lithuanian TTS

Lithuanian is a Baltic language with a rich inflectional morphology and complex prosodic structure. It is a pitch-accent language with free stress, meaning the stress can fall on any syllable in a word, and can change the position depending on the grammatical form.

In the context of TTS synthesis, Lithuanian presents several challenges. First, the extensive inflection leads to a high number of unique word forms, which is significantly higher than in English. This results in data sparsity issues, where many valid word forms may not appear in the training set,

i.e., a high out-of-vocabulary word rate. Second, the free stress system complicates pronunciation modeling. Typically, stress marks are omitted in written Lithuanian. However, stress position and tone (acute, circumflex, or short) determine the meaning of monographic words. Examples are shown in Table 1.. A grapheme-based model with accentuation marks has been shown to improve synthesis quality in Lithuanian [15].

Word	Accentuation	Meaning
Antis	*ántis* (Acute) *añtis* (Circumflex)	A duck (noun) Bosom/Chest (noun)
Kasa	*kãsa* (Circumflex) *kasà* (Short)	He/she digs (verb) Braid/Pancreas (noun)

Table 1. *Examples of Lithuanian homographs where accentuation determines meaning. A grapheme-only model cannot distinguish these without context or explicit stress marks.*

To overcome these challenges, tools like **Kirčiuoklis** [16] (Vytautas Magnus University) are often employed in the text normalization pipeline. Kirčiuoklis automatically assigns stress marks to raw text. One weakness of Kirčiuoklis is that it relies on simple word-dictionary based lookup, which does not take into account the context of the word. Thus, it suggests multiple possible accentuation variants for homographs, leaving it up to the user to select the correct one.

In the absence of a high-quality, context-aware G2P converter for Lithuanian, this thesis will focus on grapheme-based TTS synthesis with accentuation marks provided by Kirčiuoklis. In cases where Kirčiuoklis suggests multiple accentuation variants for a word, no stress marks will be added, leaving the TTS model to infer the correct prosody from context.

1.4 Embeddings and Representation Learning

1.4.1 The Concept of Embeddings

In machine learning, embeddings are dense vector representations of discrete entities (such as words, characters, or speakers) to a high-dimensional continuous vector space. Unlike one-hot encodings, which are sparse and highly dimensional, embeddings provide a dense, lower-dimensional representation that captures semantic relationships between underlying entities. For instance, in word embeddings, similar words tend to have more similar (correlated) vector representations, while dissimilar words map to more distant points in the vector space [17].

1.4.2 Text Embeddings

The “Encoder” part of a TTS model is responsible for converting a sequence of input symbols (characters or phonemes) into a sequence of feature vectors. Usually, this is done using an embedding layer, which maps each “categorical” input symbol to a learnable fixed-size vector representation. During training, these embeddings are learned jointly with the rest of the TTS model.

1.5 Text-to-speech synthesis

Text-to-Speech (TTS) synthesis, also known as speech synthesis, is the process of converting written text into human-like spoken words. Nowadays TTS is a key technology in numerous applications, including virtual assistants, accessibility tools, and language learning platforms.

The *one-to-many* nature of the mapping from text to speech adds presents a significant challenge, where a single text input can map to multiple speech outputs with a variety of speaking styles, emotions, and prosodic variations [18]. Thus, TTS is inherently a *large-scale inverse problem* [19] — reconstructing the original waveform from incomplete data (text) requires inferring missing information.

1.5.1 Traditional TTS approaches

Early attempts at artificial speech synthesis evolved from the first mechanical devices in the 18th century to electronic systems. Wolfgang von Kempelen's mechanical speaking machine [1] demonstrated basic phoneme, and later short phrase production using a physical model of the vocal tract. In 1939, Homer Dudley's invention of the Voder [20] became the first electronic speech synthesizer that could produce intelligible speech through operator-controlled acoustic parameters, establishing the foundation for modern electronic synthesis methods.

In the decades that followed, two main approaches for speech synthesis emerged: concatenative synthesis and parametric synthesis.

1.5.2 Concatenative synthesis

The concatenative synthesis approach [21] synthesizes speech by piecing together prerecorded segments of human speech. This method involves several steps. First, it requires prerecording a large database of speech segments spoken by a human voice actor in pristine, highly controlled studio conditions to ensure consistent audio quality and minimize background noise. Each segment is labeled and indexed based on its phonetic and prosodic properties.

During synthesis, the system breaks down the input text into short linguistic units (such as phonemes or syllables) using a text analysis module. Then, it queries the speech database to find the best-matching segments for each unit using selection cost functions [22]. The retrieved segments are blended and concatenated to form a continuous speech waveform. Finally, the system uses signal processing techniques to smooth the transitions between segments and adjust pitch and duration to match the desired output characteristics.

Although concatenative synthesis is a fast method capable of producing high-quality speech, its performance is heavily dependent on the size and quality of the underlying speech corpus [23]. A significant limitation is that the system fails to maintain naturalness when the required speech segments are not present in the database. However, even the largest corpora cannot cover all variants of contextual speech segments. When the system has to synthesize out-of-database segments, the final audio suffers from audible continuity distortions at the concatenation points [22]. The segments may not blend smoothly due to differences in pitch, duration, and timbre — as a result, stringing dis-

jointed segments together does not capture the natural rhythm and intonation patterns of connected speech.

The creation of a comprehensive corpus for such a purpose is costly and time-consuming. Improving a concatenative system requires language-specific expertise to design and update the corpus with good-quality, well-pronounced word units [23]. These rigid requirements are especially challenging for languages with rich morphology, and especially for low-resource languages like Lithuanian.

1.5.3 Parametric synthesis

In contrast, statistical parametric speech synthesis [24] (SPSS) uses statistical models, typically Hidden Markov Models (HMMs) [25], to generate the parameters that control a speech waveform.

The workflow has two distinct stages: training and synthesis [24]. In the training stage, the model analyzes a large corpus of recorded speech to learn the relationship between linguistic features (e.g., phonemes, prosody) and the statistical distributions of acoustic features (e.g., spectral envelope, fundamental frequency). In the synthesis stage, the system converts input text into a sequence of linguistic features, the trained model predicts the corresponding acoustic parameters, and finally the speech waveform is reconstructed based on these predictions.

Compared to concatenative synthesis, the statistical approach allows for more flexibility and control over the speech synthesis process, enabling the generation of a variety of voices, speaking styles, and emotions.

However, HMM-based synthesis [25] had a persistent problem: the statistical averaging built into the models tended to over-smooth the acoustic features, creating the characteristic “buzzy” or “muffled” sound that lacked the sharpness and detail of natural human speech. In general, well-tuned concatenative synthesis produced higher-quality speech than the best parametric methods.

1.6 Deep learning for end-to-end TTS

The limitations of complex, multi-stage pipelines motivated the creation of E2E models. E2E systems learn the entire speech synthesis process — from input text directly to acoustic output — using a single neural network. This approach promised to eliminate the need for hand-crafted pipelines that were difficult to design, required extensive expertise, and suffered from errors that accumulated across multiple components. By learning directly from text-audio pairs, E2E models showed they could produce speech with higher naturalness and expressiveness than previous methods, representing a significant leap in TTS technology.

Although deep learning TTS models are more robust to variations in data quality compared to concatenative approaches, they are essentially “data-hungry” systems that require large amounts of training data to achieve optimal performance. Extrapolating from results in language modeling, it is observed that model performance follows general scaling laws [26], improving as the amount of training data increases.

However, in the context of multi-speaker synthesis, there is a trade-off between the breadth of

the data (number of distinct speakers) and the depth of the data (duration of audio per speaker). In theory, training on a dataset with a massive number of speakers, even with limited data per speaker, may allow the model to learn a more generalized latent space of voice characteristics. This high variance in the training data could act as a form of regularization, preventing overfitting to noise and idiosyncrasies of individual speakers. In contrast, datasets with fewer speakers but high duration per speaker allow the model to capture fine-grained prosodic details specific to those voices, potentially achieving higher stability but lower generalization capabilities.

1.7 Foundational architectures

In order to understand modern TTS systems, it is necessary to review the underlying neural architectures that enable sequence modeling.

1.7.1 Feedforward neural networks

Feedforward Neural Networks (FNNs) are the simplest type of artificial neural networks, consisting of layers of interconnected nodes (neurons) where information flows in one direction — from the input, through hidden layers, to the output. While FNNs can be useful for basic regression or classification tasks, they lack the memory and context-awareness needed for processing sequential data like text and speech. Therefore, FNNs are generally not suitable for modelling TTS tasks that require understanding temporal dependencies.

1.7.2 Encoder-Decoder framework

The Encoder-Decoder architecture is a neural network design consisting of two components, namely an encoder and a decoder. The encoder processes the input data and compresses it into a high-dimensional latent representation. This vector captures the meaningful features of the input. The decoder uses this latent representation as context to generate the final output. This architecture is commonly used in sequence-to-sequence (seq2seq) tasks [27], such as machine translation (text-to-text) and text-to-speech synthesis (mapping text to audio frames).

1.7.3 Autoregressive and Non-autoregressive models

Within the sequence-to-sequence framework, TTS models can be generally categorized into autoregressive and non-autoregressive (parallel) models based on how they generate the output sequence.

Autoregressive models generate each output time-step (sample or frame) sequentially, conditioning the generation on previously generated time-steps. This approach allows the model to capture temporal dependencies effectively, producing high-quality speech. While earlier recurrent neural network-based systems utilized this approach, it was revolutionized by WaveNet [28] for raw waveform generation, and later by Tacotron [19] for spectrogram generation. Autoregressive models were the dominant architecture in neural TTS for several years. However, their sequential nature leads to slow inference times, as each frame must be generated one after another.

Non-autoregressive (parallel) models were developed to address the slow inference speed and stability issues of autoregressive systems. These models generate all output frames simultaneously, allowing for much faster synthesis. Examples include FastSpeech [29] and Glow-TTS [3]. Non-autoregressive models often rely on duration prediction modules to determine how long each input token should be stretched in the output sequence. While these models achieve significant speed-ups during inference, they may struggle to capture fine temporal dependencies, potentially leading to lower naturalness compared to autoregressive models.

1.8 TTS pipeline and acoustic models

As shown in Figure 3., modern neural TTS systems generally follow a two-stage pipeline. An **Acoustic model** first generates intermediate acoustic features (typically Mel-spectrograms) from the input text. Then, a **Vocoder** converts these features into the raw audio waveform.

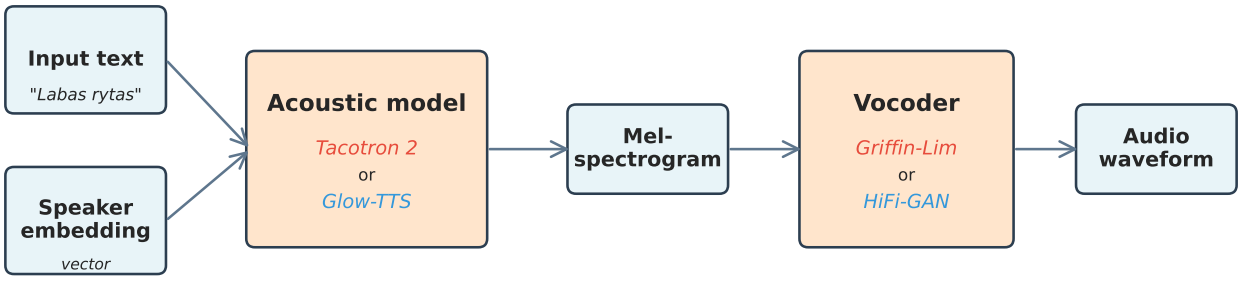


Figure 3. *Text-to-Speech synthesis pipeline. The TTS model generates Mel-spectrograms from input text, which are then converted to raw audio waveforms by a neural vocoder.*

The following sections detail the specific acoustic models used in this work.

1.8.1 Tacotron 2 (Autoregressive)

Tacotron [19] and Tacotron 2 [2] are notable TTS models based on the autoregressive sequence-to-sequence architecture. The variant that this thesis primarily focuses on is Tacotron 2 with Dynamic Convolution Attention (DCA) [30]. The complete architecture of Tacotron 2 is depicted in Figure 4.

The architecture has three main components:

1. **Encoder:** The encoder's input is a character or phoneme sequence. A stack of convolutional layers followed by a bidirectional LSTM converts the character sequence into a high-level hidden feature representation.

2. **Attention mechanism:** The attention mechanism bridges the encoder and decoder — it determines which parts of the input text should be attended to when generating each of the output audio frames. While the original Tacotron 2 used location-sensitive attention [31], a more modern variant employs DCA [30], which offers robust alignment for long inputs and prevents the model from repeating or skipping words.

3. **Decoder and Post-net:** The autoregressive LSTM decoder generates a coarse Mel-spectrogram frame. This output is then passed through a convolutional **Post-net** which predicts a residual to refine the spectral details and improve reconstruction quality.

Additionally, the model includes a **Stopnet** component. This linear layer projects the LSTM decoder's output to a scalar, predicting the probability that the current frame is the "stop token", which halts the synthesis process. This allows the model to dynamically determine the output duration. Unlike the standard architecture, the Stopnet is often separated from the decoder's gradient flow to prevent the stop-token loss from destabilizing the attention alignment.

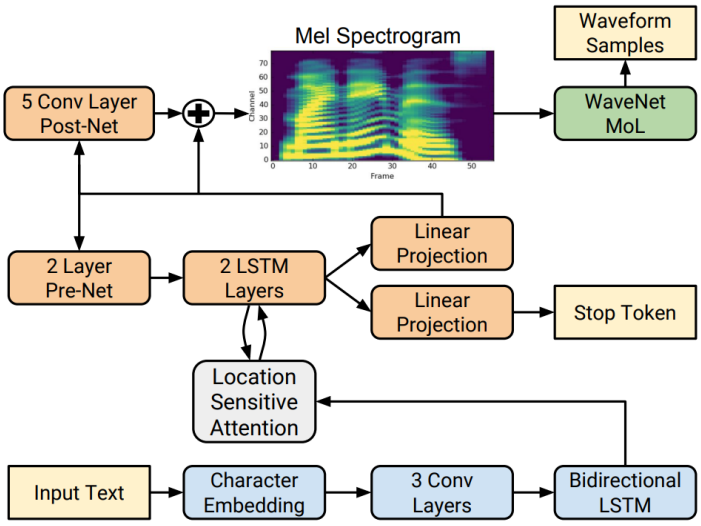


Figure 4. *Tacotron 2 architecture, diagram taken from the original paper [2]. Note the recurrent connections in the decoder and the attention mechanism aligning encoder outputs to decoder steps.*

The model is optimized by minimizing a combination of losses: the mean squared error (MSE) between the predicted and ground truth Mel-spectrograms (both the Decoder and Post-net outputs), the spectral similarity index (SSIM) loss, the "guided attention" loss to encourage diagonal alignments, and the binary cross-entropy loss for the stop token prediction.

While Tacotron 2 can generate high-quality speech, being an autoregressive model, it suffers from slow inference times. Additionally, the attention mechanism can suffer from stability issues, such as attention failures that result in skipped or repeated words in the synthesized speech.

1.8.2 Glow-TTS (Non-autoregressive)

Glow-TTS [3] is a notable non-autoregressive TTS model that applies flow-based generative models to text-to-speech tasks. Unlike Tacotron 2, Glow-TTS generates the entire Mel-spectrogram in parallel, significantly speeding up inference.

A key innovation of Glow-TTS is its ability to learn alignment internally without requiring external priors (aligners), achieved through the following components:

- **Monotonic Alignment Search (MAS):** Unlike FastPitch [32], which relies on external aligners to train a duration predictor, Glow-TTS treats alignment as a latent variable. It employs

MAS to find the most probable monotonic path between the input text and the target Mel-spectrogram, maximizing the log-likelihood of the data.

- **Flow-based decoder:** The model uses a stack of normalizing flows — specifically invertible 1×1 convolutions and affine coupling layers. This decoder transforms a simple prior distribution (conditioned on the text input) into the complex distribution of Mel-spectrograms.

By utilizing the properties of flows, Glow-TTS allows for varied speech synthesis by sampling from the latent space and manipulating the noise temperature. Furthermore, the duration of the speech can be controlled by modifying the predicted duration of the alignment.

The high-level architecture of Glow-TTS is shown in Figure 5.

Figure 5. *The Glow-TTS architecture. It utilizes a Transformer-based text encoder and a flow-based decoder, connected via Monotonic Alignment Search to enable parallel Mel-spectrogram generation [3].*

The primary advantages of Glow-TTS over Tacotron 2 are inference speed, robustness (the monotonic constraint prevents skipping or repeating words), and the elimination of the need for an external aligner during training.

1.8.3 Other notable TTS models

Besides Tacotron 2 and Glow-TTS, other notable TTS architectures include **FastPitch** [32], which uses a feed-forward Transformer architecture with explicit pitch and duration predictors, and **VITS** (Conditional Variational Autoencoder with Adversarial Learning) [33], which combines the acoustic TTS model (Glow-TTS) with a neural vocoder (HiFi-GAN) into a single end-to-end architecture.

1.9 Neural vocoders

As previously illustrated in the pipeline overview Figure 3., acoustic models like Tacotron 2 and Glow-TTS generate Mel-spectrograms rather than raw audio waveforms. Mel-spectrograms are lossy representations that capture the magnitude of the sound frequency bands, but discard phase information. Converting a lossy spectrogram into audio is a non-trivial task, requiring a component called a vocoder to estimate the missing phase and reconstruct the waveform.

Traditionally, the Griffin-Lim algorithm [34] was used to iteratively estimate and reconstruct the phase information from the magnitude spectrogram. However, this method often produces audio with noticeable artifacts and lower quality compared to natural speech. Modern TTS systems use neural vocoders — deep generative models trained to map acoustic features to raw waveforms.

1.9.1 Evolution of neural vocoders

WaveNet [28] was one of the first autoregressive models to produce high-fidelity audio, but its sequential generation process made it prohibitively slow for real-time applications. To address these speed limitations, Generative Adversarial Network (GAN) based vocoders were introduced.

HiFi-GAN [35] is currently one of the state-of-the-art neural vocoders. It consists of a Generator that upsamples the Mel-spectrograms using transposed convolutions and a set of Discriminators (multi-scale and multi-period discriminators) that ensure the generated audio is indistinguishable from real human speech. HiFi-GANs are highly efficient and capable of faster-than-real-time synthesis on consumer hardware while maintaining high perceptual quality.

1.9.2 Cross-lingual vocoding

The framework used in this thesis, Coqui TTS [36], provides a pre-trained HiFi-GAN vocoder trained on the VCTK dataset [37], a large multi-speaker dataset with 110 English speakers.

The use of a vocoder trained on English data for Lithuanian synthesis is justified by the language-agnostic nature of the phase reconstruction task. Neural vocoders' primary function is to model the physics of human speech production rather than linguistic features. While language-dependent phonetic nuances exist, studies have shown that vocoders trained on large, diverse datasets can effectively generalize to unseen speakers and languages [38]. Therefore, this thesis utilizes the pre-trained HiFi-GAN model for waveform generation, ensuring that the acoustic parameters (sampling rate, FFT size, Mel-filterbank limits, etc.) of the input Mel-spectrograms are configured to match those used during the vocoder's training.

1.10 Multi-speaker TTS

Multi-speaker TTS models are designed to synthesize speech in the voices of multiple speakers. In order to achieve this, these models are indeed trained on data from many different speakers, allowing them to learn the characteristics of each voice and synthesize speech that sounds like a specific individual, while still being able to generalize the shared linguistic and acoustic patterns across speakers.

1.10.1 Speaker embeddings

To enable multi-speaker synthesis, TTS models require a representation of the speaker's identity. In multi-speaker models, the network is conditioned on a speaker embedding. The model learns a shared representation of phonetics (how text maps to sound generally) while using an additional input — the speaker embedding — to adjust the timbre and prosodic characteristics specific to a voice.

Early successful implementations of this approach include Deep Voice 2 [39], which demonstrated effective multi-speaker synthesis by learning speaker-specific embeddings.

Nowadays there are several techniques for incorporating speaker embeddings into TTS models:

Lookup Tables (LUT): Early multi-speaker approaches used simple, learnable embeddings where each speaker ID is mapped to a unique vector. The vectors are initialized randomly and learned jointly with the TTS model. While this method is straightforward and efficient, it cannot generalize to speakers not seen during training.

d-vectors and x-vectors: Transfer learning approaches [40] have demonstrated adapting speaker verification models for multispeaker TTS synthesis, enabling better speaker adaptation and higher voice quality. The general architecture of such a speaker encoder is illustrated in Figure 6. A speaker encoder model pre-trained on a massive, noisy dataset with thousands of speakers (e.g., the VoxCeleb dataset [41]) learns the general speaker space. Its pre-trained weights are frozen and used to extract embeddings for the TTS training data, allowing the TTS model to effectively account for multi-speaker variation.

Figure 6. *General architecture of a Speaker Encoder. A reference audio of arbitrary length is processed (typically by LSTM or TDNN layers) and pooled to produce a fixed-length embedding vector (e.g., d-vector) representing the speaker identity.*

d-vectors: d-vectors [42] are fixed-length speaker embeddings derived from a separate speaker verification model. A reference encoder network takes a reference audio recording of arbitrary length and compresses it into a fixed-length vector known as a d-vector, that summarizes the speaker's timbral and prosodic characteristics. These d-vectors are then provided as additional input to the TTS model, and are kept fixed during TTS training.

x-vectors: An evolution of d-vectors, x-vectors [43] use a Time Delay Neural Network (TDNN) architecture to capture the temporal context more effectively. These embeddings have shown an improved ability in zero-shot TTS scenarios.

One limitation of d-vectors and x-vectors is that if the reference audio is of poor quality or contains background noise, the resulting speaker embedding may not accurately represent the speaker's identity, leading to degraded synthesis quality.

1.10.2 Challenges

One key challenge in multi-speaker TTS is ensuring that the model can generalize across many speakers while still maintaining high quality for each. There is a trade-off between the *breadth* of the dataset (number of speakers) and the *depth* (minutes of audio per speaker).

Standard TTS systems historically required 10 to 20 hours of recorded speech for a single professional speaker. However, deep learning models capable of *transfer learning* can produce intelligible speech for a new speaker with significantly less data, potentially as little as a few minutes — if the base model has been pre-trained on a sufficiently diverse multi-speaker dataset.

1.11 Evaluation metrics

Evaluating Text-to-Speech systems is notoriously difficult because "quality" and "naturalness" are subjective metrics defined by human perception.

1.11.1 Objective metrics

Several objective metrics have been proposed to quantify the quality of synthesized speech. While there is no single mathematical function that perfectly aligns with human judgement of natu-

ralness and intelligibility, these metrics are easy to compute and can provide useful insights during model development. Common objective metrics are summarized in Table 2.

Acronym	Full name	Description
MCD [44]	Mel-Cepstral Distortion	Measures spectral distortion between synthesized and reference audio. Lower values indicate better quality.
F_0 RMSE	Fundamental Frequency Root Mean Square Error	Quantifies pitch accuracy by computing the root mean square error of fundamental frequency (pitch). Lower is better.
PESQ [45]	Perceptual Evaluation of Speech Quality	Predicts perceived audio quality on a scale from -0.5 to 4.5. Higher is better.

Table 2. *Common objective metrics for TTS evaluation. These metrics provide quantitative measures of spectral similarity, pitch accuracy, and perceived audio quality.*

1.11.2 Subjective metrics: Mean Opinion Score

The gold standard for evaluating speech synthesis quality is the Mean Opinion Score (MOS), originally derived from telecommunications quality standards (ITU-T P.800) [46].

In a MOS test, human listeners (raters) are presented with a set of synthesized speech audio samples and asked to rate them on a 5-point Likert scale. The standard scale for "naturalness" is presented in Table 3.

Score	Description
5	Excellent (Imperceptible difference from real speech)
4	Good (Perceptible but not annoying)
3	Fair (Slightly annoying)
2	Poor (Annoying)
1	Bad (Very annoying / Unintelligible)

Table 3. *Mean Opinion Score (MOS) scale for TTS evaluation. Raters assign scores based on the perceived naturalness of synthesized speech samples.*

The final score is the arithmetic mean of all ratings collected for a specific TTS system. Although MOS is subjective, with a sufficient number of raters (typically, at least 15–20), the scores tend to converge and provide a reliable ranking between different models.

1.11.3 Latin square design

A major challenge in subjective listening tests is controlling for biases. If a rater hears the same sentence produced by different TTS systems in a row, their ratings may be influenced by the repetition (repetition effect) or by the relative order of presentation (order effect). For instance, a "Slightly annoying" sample may be rated more harshly if it follows an "Excellent" sample (contrast effect).

In order to mitigate these biases, a Latin square design [47] is often employed for MOS tests. In this experimental design:

1. A set of test sentences (utterances) is selected.
2. The listeners are divided into groups.
3. The presentation is balanced such that each listener hears every test sentence exactly once, and every TTS system (model) exactly once per block of trials, but never the same sentence-system combination twice.

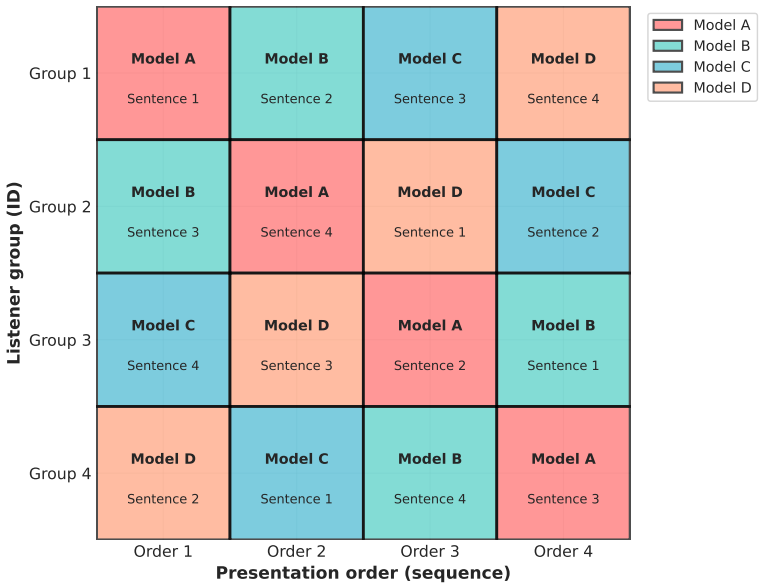


Figure 7. *Latin square design for TTS evaluation. Each listener group hears each sentence exactly once, and each TTS system exactly once per block, ensuring balanced exposure and mitigating order/repetition biases.*

An example Latin square design for 4 TTS systems and 4 test sentences is illustrated in Figure 7.

For a multi-speaker TTS evaluation (as is the case in this thesis), the Latin square design ensures that the ratings reflect the quality of the model rather than the linguistic content of the sentence or listener fatigue. By rotating the systems and sentences across listener groups, the influence of specific difficult sentences is averaged out across all models.

1.12 Research gap

While the literature demonstrates the capabilities of modern deep learning TTS architectures like Tacotron 2 and Glow-TTS to produce highly natural-sounding speech, several questions remain unanswered regarding their application to low-resource, morphologically complex languages like Lithuanian.

Firstly, although neural TTS models may follow general neural model scaling laws [26], implying that performance improves with more data, there is limited understanding of the optimal composition of training data under a fixed budget. In low-resource settings, scaling up the dataset size is not always feasible, and this may be further constrained by the computational resources required for training large models. A critical question is whether it is more beneficial to train on a smaller number

of speakers with more data per speaker (high depth) or a larger number of speakers with less data per speaker (high breadth).

Current research primarily focuses on high-resource languages like English, where the availability of large, balanced multi-speaker datasets masks the nuances of this trade-off. For a pitch-accent language like Lithuanian, the requirements may be different. It is hypothesized that high-diversity datasets may help the model learn a richer representation of prosodic patterns, while high-depth datasets may improve the model's naturalness for the target speakers.

Secondly, most multi-speaker TTS research assumes access to large-scale datasets with thousands of utterances per speaker. There is a lack of research exploring how different TTS architectures (autoregressive vs. non-autoregressive) perform when the data per speaker is scarce (e.g., under 10 minutes).

This thesis aims to fill the research gap by systematically evaluating the efficiency of Tacotron 2 and Glow-TTS models trained on Lithuanian speech data. By controlling the total dataset size and varying the distribution of speakers and data per speaker, this study will provide insights into the optimal data composition for training multi-speaker TTS models in low-resource settings.

To summarize, the key research questions this thesis seeks to answer are:

- How does the trade-off between data breadth (number of speakers) and data depth (minutes per speaker) affect the performance of multi-speaker TTS models for Lithuanian?
- How do different TTS architectures (Tacotron 2 vs. Glow-TTS) perform under varying data selection strategies in low-resource settings?

The experiments will involve training models on three distinct data selection strategies:

- Lower diversity (30 speakers), but high fidelity (45 min each), total: 22.5 hours.
- Moderate diversity (60 speakers), moderate data (22.5 min each), total: 22.5 hours.
- High diversity (180 speakers), low resource (7.5 min each), total: 22.5 hours.

The extreme low-depth condition (7.5 minutes per speaker) might pose convergence challenges for the models, especially for Tacotron 2, which relies on learning robust attention alignment. Thus, alignment convergence and training stability will be monitored to assess how data composition affects model robustness.

1.13 Summary

This literature review has provided an overview of the theoretical foundations required for modern Text-to-Speech synthesis. The evolution of TTS systems from mechanical apparatuses, through concatenative and statistical methods, to end-to-end deep learning architectures capable of generating natural-sounding speech has been discussed.

We have reviewed the entire TTS pipeline — from signal processing (sampling, quantization, Fourier transforms, and Mel-spectrogram extraction), through text normalization and representation

(graphemes vs. phonemes), to deep learning architectures for acoustic modeling and vocoding. The literature highlights two architectures for acoustic modeling: the autoregressive Tacotron 2, known for high-quality spectral output but slow inference and stability issues, and the non-autoregressive Glow-TTS, which offers parallel generation and improved robustness.

We have examined the challenges specific to Lithuanian TTS synthesis. Unlike English, Lithuanian languages's high inflectional morphology leads to a large number of unique word forms, and its prosodic system requires handling of free stress and pitch accents, necessitating the use of tools like Kirčiuoklis for accentuation marking.

Finally, we have reviewed the role of speaker embeddings in enabling multi-speaker synthesis and the use of neural vocoders, specifically HiFi-GAN, to reconstruct high-fidelity waveforms from Mel-spectrograms. Despite these advancements, a gap remains in understanding how data diversity versus quantity affects model performance for complex, low-resource languages — a challenge this thesis addresses through the experiments detailed in the following chapters.

2 Methodology

This chapter details the experimental setup, data processing pipeline, training configurations, and evaluation protocol used to determine the optimal composition of multi-speaker training data for Lithuanian TTS. The study uses factorial design to compare the performance of two distinct architectures — Tacotron 2 (autoregressive) and Glow-TTS (non-autoregressive) — under varying degrees of data breadth and depth, while controlling for the total training budget.

The high-level experimental workflow is illustrated in Figure 8.

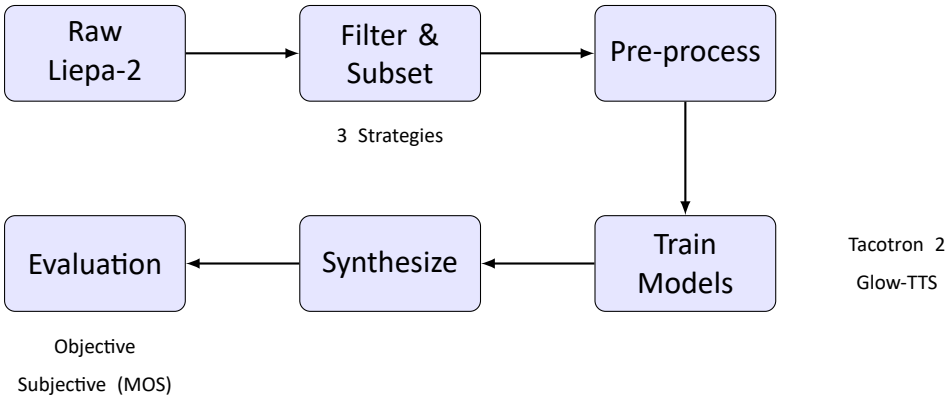


Figure 8. *Experimental pipeline overview. The process flows from raw corpus selection to comparative evaluation.*

2.1 Research Design

To investigate the impact of data distribution on synthesis quality, the experiments vary the balance between the number of speakers (N) and the amount of data per speaker.

2.1.1 Variables

. Independent Variables:

- **Data selection strategy:** Three subsets varying in speaker count versus duration per speaker (breadth vs. depth).
- **Model architecture:** Autoregressive (Tacotron 2) vs. Non-autoregressive (Glow-TTS).

. Dependent Variables:

- **Objective metrics:** MCD, F0 RMSE, and attention alignment convergence.
- **Subjective metrics:** Naturalness ratings via MOS.

. Controlled Variables:

- **Training budget:** Fixed at 22.5 hours of audio data per model.

- **Training duration:** 200 epochs for Tacotron 2 and 400 epochs for Glow-TTS, adjusted for convergence characteristics.
- **Vocoder:** Pre-trained HiFi-GAN (frozen).
- **Domain:** Read speech (adults only).

2.2 The Liepa 2 Dataset

The primary dataset of this study is the **Liepa 2** Lithuanian speech corpus [8]. The full corpus contains 1,000 hours of annotated audio from 2,621 unique speakers. Of this, 939 hours are transcribed speech, while the remaining 61 hours consist of silence or noise segments. The recordings span diverse speech styles and contexts, including read speech (audiobooks, studio recordings, dictaphone) and spontaneous speech (phone, radio, TV), sampled at 16 kHz in 16-bit PCM WAV format.

The dataset was obtained from `https://huggingface.co/datasets/isLucid/liepa-2`.

According to the documentation, the dataset filenames encode metadata about each recording. The filenames were used to retrieve the following information about each audio file: lossiness (lossy or raw), speech type (read or spontaneous), recording type (audiobook, dictaphone, phone, radio, studio, TV), gender (male or female), age group (0–12, 13–17, 18–25, 26–60, 60+), speaker ID, recording sequence number, and utterance number.

The documentation notes that speaker IDs are unique only within the context of an annotator, but not globally across the entire corpus. However, the documentation claims that duplicate speakers are rare and not specifically marked in the dataset.

2.2.1 Corpus Analysis

The dataset was analyzed to determine the distribution of audio duration, sentence lengths, and speaker counts across different speech types and recording conditions. Due to the large size of the dataset, the duration was estimated using metadata extraction rather than loading and processing each file, which would be computationally expensive.

Analysis reveals that spontaneous speech contributes under 10% of the total corpus duration (82 hours), with read speech (audiobook, dictaphone, studio) making up the majority (857 hours). Read speech is generally more suitable for TTS training due to its consistent pronunciation and prosody.

Additionally, it was decided to exclude speakers under 18 years of age to maintain compatibility with the HiFi-GAN vocoder, which was pre-trained on adult speakers (VCTK dataset) and will be used for waveform generation. Speakers in the 0–17 age group constitute a small fraction (5%) of the total corpus duration, so their exclusion is not expected to significantly impact the available training data.

Finally, TV, phone, and radio recordings were excluded due to the less controlled recording conditions and potential background noise, which could degrade the quality of the synthesized speech. These categories account for 6% of the total read speech duration. Thus, the final filtered dataset focuses on the following categories:

- Speech type: Read speech only

- Recording type: Audiobook, Dictaphone, Studio
- Age group: Adults only (18–25, 26–60, 60+ age groups)
- Duration: $<$ 15 seconds in order to prevent GPU memory overflows during Tacotron 2 training

Table 4. summarizes the statistics of the filtered Liepa 2 dataset, which contains approximately 714 hours of read speech from 1,792 unique adult speakers.

Table 4. *Statistics of the filtered Liepa 2 dataset (Adult Read Speech).*

Source	Sum Duration (s)	Mean Sent. Len.	Speakers	Files
Audiobook	87,087	39.52	50	32,129
Dictaphone	672,918	37.15	453	243,513
Studio	1,819,086	42.85	1,301	616,314
Total	**2,570,992**	**—**	**1,792**	**891,956**

2.2.2 Experimental Data Subsets

The Liepa 2 corpus presents a challenge as most speakers contribute under 30 minutes of audio. Three datasets were generated from the filtered Liepa 2 data. All strategies maintain a fixed total training budget of 22.5 hours to ensure fair comparison across experiments. The configurations are defined in Table 5. and visually represented in Figure 9.

Table 5. *Experimental data subsets. Total duration is constant, while speaker count (N) and duration per speaker vary inversely.*

Number of Speakers (N)	Time/Speaker	Total Time
30	45.0 min	22.5 hours
60	22.5 min	22.5 hours
180	7.5 min	22.5 hours

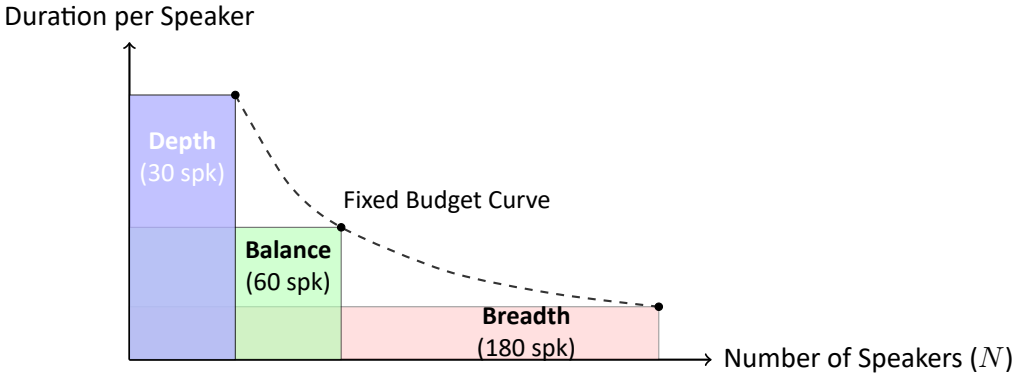


Figure 9. *Visual representation of the data strategies. The area of each rectangle (Total Audio) remains constant ($\approx 22.5hours$), illustrating the trade-off between speaker diversity (Breadth) and data density (Depth).*

Only speakers with at least the required minimum duration for the specific subset were eligible for selection (e.g., at least 45 minutes for the 30-speaker set).

To ensure fair evaluation, speaker sets were nested: the 30 speakers in the 30-speaker set are included in the 60-speaker set, which are included in the 180-speaker set. An exact 50/50 male-female split was maintained in all datasets to avoid gender bias.

This decision will allow using the same speakers for evaluation across all models.

2.3 Preprocessing

2.3.1 Text Normalization and Accentuation

Raw Liepa 2 transcripts are largely normalized (numbers, dates, abbreviations, and acronyms are expanded), however, some additional normalization was required to standardize the text for grapheme-based TTS training:

1. **Cleaning:** Rare and non-standard punctuation was mapped to a standard set (.,-?!) and remaining extraneous characters were removed.
2. **Whitespace:** Consecutive whitespace characters were collapsed, and leading/trailing whitespace was trimmed.
3. **Accentuation:** Raw text was processed using **Kirčiuoklis** [16] for automatic stress assignment. Ambiguous homographs were left unaccentuated, relying on the model to infer prosody from context.
4. **Lowercasing:** All text was converted to lowercase to reduce the vocabulary size.
5. **Letter substitution:** Non-Lithuanian letters were replaced with equivalents (‘q’ $\rightarrow$ ‘k’, ‘w’ $\rightarrow$ ‘v’, ‘x’ $\rightarrow$ ‘ks’).

As a result of these normalization steps, the vocabulary size is reduced from 140 characters to 41 characters. The final alphabet used for training consists of the following characters:

```
a ą b c č d e ę ė f g h i į y j k l m n o p r s š t u ų ū v z ž ´ ` ~ (space)
. , - ? !
```

2.3.2 Audio Preprocessing

Audio recordings were resampled from their original **16,000 Hz** to **22,050 Hz**. While resampling to a higher frequency does not add new information, the resampling was performed to match the pre-trained vocoder. Leading and trailing silence was trimmed. Acoustic features were extracted using the parameters in Table 6.

A complete list of audio parameters is presented in the Appendix 7

Table 6. *Mel-spectrogram extraction parameters.*

Parameter	Value
Sampling Rate	22,050 Hz
FFT Size	1024 samples (46 ms)
Hop Length	256 samples (11.6 ms)
Window Length	1024
Mel Channels	80
Frequency Range	0–8000 Hz
Pre-emphasis	0.98

2.4 Model Architectures

Models were implemented using the **Coqui TTS** [36] framework.

2.4.1 Speaker Conditioning

To enable multi-speaker synthesis, speaker identity was provided via fixed-length embeddings, specifically **x-vectors** [43]. These 512-dimensional were extracted using a speaker encoder [40] pre-trained on VoxCeleb (available in the Coqui TTS model zoo) and kept frozen during TTS model training.

Tacotron 2 used both the external x-vectors (concatenated to encoder output) and a learnable embedding layer.

Glow-TTS used only learnable fixed-length speaker embeddings since the Glow-TTS implementation does not support both types simultaneously.

2.4.2 Tacotron 2 (Autoregressive)

The autoregressive model used is **Tacotron 2**, modified with DCA to improve alignment stability for long-form utterances.

- **Encoder:** 3-layer convolutional stack + bi-directional LSTM (512 units).
- **Decoder:** 2-layer LSTM (1024 units) with dynamic convolution attention.

The training objective $\mathcal{L}_{T2}$ is a weighted sum of auxiliary losses:

$$\mathcal{L}_{T2} = \mathcal{L}_{L1} + \mathcal{L}_{Post} + \lambda_{SSIM}(\mathcal{L}_{SSIM}) + \lambda_{attn}(\mathcal{L}_{Guided}) + \lambda_{stop}(\mathcal{L}_{Stop}) \tag{3}$$

Where $\mathcal{L}_{Guided}$ enforces diagonal attention alignment, critical for long-form synthesis.

2.4.3 Glow-TTS (Non-autoregressive)

The non-autoregressive model is **Glow-TTS**, a flow-based architecture with MAS.

- **Backbone:** Transformer encoder and flow-based decoder.

- **Alignment:** Trained using unsupervised Soft-DTW (Dynamic Time Warping) to generate duration targets without external aligners.
- **Predictors:** Explicit 1D-convolutional duration predictor.

The objective function is the exact log-likelihood of the data:

$$\log P_X(x|c) = \sum_{j=1}^{T_{mel}} \log \left| \det \frac{\partial z_j}{\partial x_j} \right| + \sum_{j=1}^{T_{mel}} \log \mathcal{N}(z_j; \mu, \sigma) \tag{4}$$

Additionally, a duration predictor is trained via MSE loss $\mathcal{L}_{dur}$ to predict phoneme durations.

2.4.4 Vocoder

A **HiFi-GAN** [35] model, pre-trained on the multi-speaker VCTK corpus [37], was used as the vocoder for all acoustic models. All weights were frozen to isolate the performance differences to the acoustic models only.

2.5 Model training configurations

2.5.1 Environment and Framework

Experiments were conducted on a personal workstation equipped with an AMD Epyc 7642 CPU, 256 GB RAM, and NVIDIA GeForce RTX 3090 (24 GB) GPU. The software environment included Ubuntu 25.04, Python 3.13.3, Coqui TTS v0.27.2, and CUDA 13.0 for GPU acceleration. The pipeline was automated via Make, with separate steps for data preprocessing, speaker embedding computation, model training, inference, and synthesized sample deployment to the evaluation web app.

The exact Python environment configuration is provided in the accompanying GitHub repository, file `pyproject.toml`.

In the TTS training stage, the validation loss was evaluated every epoch ($\approx$ 450 steps) using a held-out 1% validation split. The best model checkpoint was selected based on the lowest validation loss.

2.5.2 Tacotron 2 configuration

The Tacotron 2 model was trained using the DCA mechanism to improve alignment stability.

The model optimization utilized a composite loss function consisting of Decoder L1 loss ($\alpha = 0.25$), Post-net L1 loss ($\alpha = 0.25$), Decoder and Post-net SSIM losses ($\alpha = 0.25$ each), Guided Attention loss ($\alpha = 5.0$), and a weighted Stop token loss (weight=15.0).

Notably, the default NoamLR learning rate scheduler caused high gradient values, instability, and sub-optimal convergence for Tacotron 2. Therefore, a MultiStepLR scheduler with more aggressive decay of 0.5 every 10,000 steps was used instead after empirical testing. The main hyperparameters are shown in Table 7.

Table 7. *Tacotron 2 with DCA training configuration.*

Parameter	Value
Validation split	1%
Batch size	64
Initial Learning Rate	0.0005
Optimizer	RAdam
LR schedule	MultiStepLR (Decay 0.5 at steps 20k, 30k, …, 70k)
Max epochs	200 ($\approx$ 90,000 steps)
Attention type	Dynamic Convolution
Separate stopnet	True
Speaker embedding dim	512
Number of speakers	30 / 60 / 180

2.5.3 Glow-TTS configuration

The Glow-TTS was trained using Negative Log-Likelihood (NLL) for the flow decoder and a monotonic alignment search. Unlike Tacotron 2, Glow-TTS converged stably with the NoamLR scheduler, but required a higher number of epochs for convergence. The configuration is shown in Table 8.

Table 8. *Glow-TTS training configuration.*

Parameter	Value
Validation split	1%
Batch size	64
Maximum Learning Rate	0.001
Optimizer	RAdam
LR scheduler	NoamLR
Warmup steps	4000
Max epochs	400 ($\approx$ 180,000 steps)
Encoder type	Rel. Pos. Transformer
Encoder layers	6
Encoder heads	2
Encoder hidden dim	192
Decoder hidden dim	192
Decoder flow blocks	12
Decoder block layers	4
Mel-spectrogram channels	80
Speaker embedding dim	512
Number of speakers	30 / 60 / 180

The loss function was a combination of Negative Log-Likelihood (NLL) loss for the flow-based decoder, Duration loss (MSE).

2.6 Evaluation protocol

The synthesized speech from the trained models was evaluated using a combination of objective and subjective metrics.

2.6.1 Model convergence

Alignment stability during training was monitored via attention alignment plots, provided by the TensorBoard [48] integration in Coqui TTS.

The **attention alignment plots** were generated during every epoch, and inspected regularly. A failure to converge to a diagonal alignment indicates that the model has failed to learn the text-to-audio mapping. This is especially relevant for the 180-speaker scenario to detect convergence failures caused by data sparsity.

2.6.2 Objective evaluation

A held-out test set of 60 standardized sentences (using seen speakers) was used to calculate objective metrics: MCD, F0 RMSE, and PESQ.

Mel-Cepstral Distortion (MCD): MCD [44] evaluates the timbre and spectral envelope of synthesized speech. It measures the difference between the Mel-frequency cepstral coefficients of the synthesized and reference speech. A lower MCD value indicates closer spectral similarity, i.e., better timbre reconstruction. The MCD is defined as:

$$\text{MCD} = \frac{10}{\ln 10}\sqrt{2\sum_{n=1}^{K}(c_n^{\text{synth}} - c_n^{\text{ref}})^2} \tag{5}$$

where c_n^{synth} and c_n^{ref} are the n-th coefficients of the synthesized and reference frames, respectively, and K is the number of coefficients (here: 24).

Fundamental Frequency Root Mean Square Error (F_0 RMSE): F_0 RMSE measures the Root Mean Square Error between the fundamental frequency (F_0) contours of the synthesized and reference speech. Lower F_0 RMSE values indicate better pitch accuracy, i.e., closer intonation reproduction. F_0 RMSE is defined as:

$$\text{RMSE}_{F_0} = \sqrt{\frac{1}{T}\sum_{t=1}^{T}(F_0^{\text{synth}}(t) - F_0^{\text{ref}}(t))^2} \tag{6}$$

$F0_t^{(syn)}$ and $F0_t^{(gt)}$ are the fundamental frequency values at time t for synthesized and ground truth speech, respectively, and T is the total number of frames. where T is the number of frames where both signals are voiced. Lower values indicate more accurate intonation.

Perceptual Evaluation of Speech Quality (PESQ): PESQ [45] is an industry-standard algorithm designed to predict human perception of speech quality. It uses a psychoacoustic model that accounts for masking effects and auditory sensitivity to compare the synthesized speech to a reference

signal. PESQ score ranges from -0.5 to 4.5, with higher scores indicating higher perceived quality and intelligibility.

Since the original and synthesized audio may differ in length due to alignment variations, Dynamic Time Warping (DTW) is applied to align the Mel-spectrograms and F0 contours before computing MCD and F0 RMSE.

2.6.3 Subjective evaluation (MOS)

Naturalness was evaluated via a web-based listening test employing a **Latin square design** to mitigate order and repetition biases. The application was developed specifically for this study, and the source code is available in the accompanying GitHub repository linked in the Appendix 6, folder `tts_rating_app`.

The participants were 21 native Lithuanian speakers recruited through university networks and social media platforms. Each rater evaluated a randomized block of sentences, ensuring balanced exposure to all models and sentences.

Naturalness was rated using the standard 5-point MOS scale. The test samples were generated using the 6 experimental models plus a human ground truth, evaluated on the same set of 60 held-out test sentences uttered by 6 speakers (3 male, 3 female, 10 sentences each) randomly selected from the 30-speaker subset.

2.6.4 Rating procedure

Raters accessed the web application via a public URL.

The interface presented one audio sample at a time, along with the corresponding text prompt. Raters listened to each sample and rated its naturalness on a scale from 1 (Bad) to 5 (Excellent). They could replay samples as needed before submitting their ratings. A progress bar indicated the number of samples rated out of the expected total (60 samples per rater).

A screenshot of the web application interface is shown in Figure 10.

2.6.5 Statistical analysis

Collected MOS ratings were screened for outliers and inconsistencies. The mean MOS and 95% confidence intervals were computed for each model and data subset, as well as for any cross-sections of interest (e.g., model type, data strategy, speaker ID).

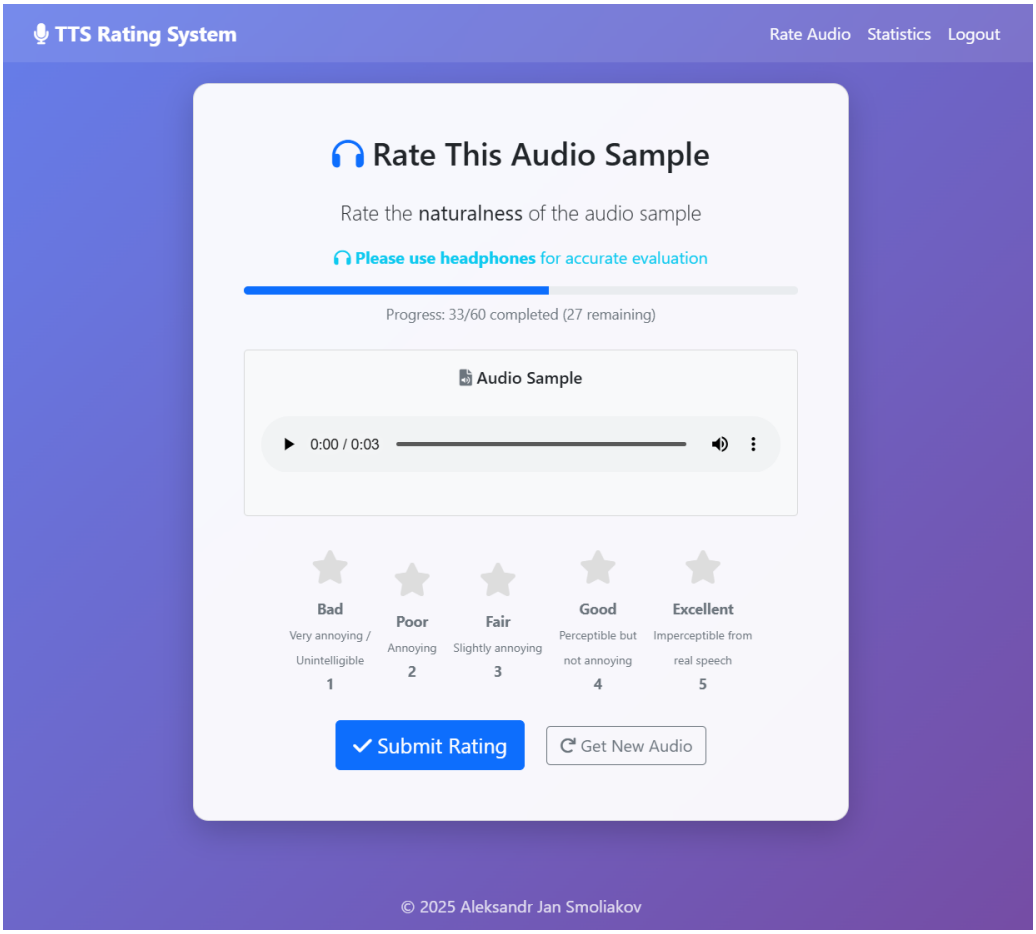


Figure 10. *Screenshot of the web-based MOS evaluation application. Raters listen to audio samples and rate their naturalness on a 5-point scale.*

3 Results and Analysis

This chapter presents the quantitative and qualitative findings of the study. The performance of the autoregressive (Tacotron 2) and non-autoregressive (Glow-TTS) models is analyzed across the three data subsets defined in Chapter 3: 30 speakers (45 minutes per speaker), 60 speakers (22.5 minutes per speaker), and 180 speakers (7.5 minutes per speaker).

3.1 Objective evaluation

Table 9. summarizes the MCD and F0 RMSE on the held-out test set.

Table 9. *Objective evaluation results. Lower is better.* ***Bold*** *indicates best performance per architecture.*

Model	Speaker Count (N)	MCD (dB)	F0 RMSE (Hz)
Tacotron 2	30	9.58	31.28
	60	**9.55**	**30.49**
	180	9.63	31.06
Glow-TTS	30	**9.90**	37.86
	60	10.00	36.18
	180	9.98	**35.69**

Across all data subsets, Tacotron 2 moderately, but consistently outperformed Glow-TTS in pitch accuracy (F0 RMSE). Interestingly, both models showed remarkable insensitivity to the data composition strategy in terms of objective metrics, with only minor variations observed — the intra-model differences were relatively small for MCD (within 0.1 dB) and for F0 RMSE (within 3 Hz). The 180-speaker configuration outperformed the other two in F0 RMSE for Glow-TTS, suggesting that increased speaker diversity did not harm pitch modeling, and may have even helped generalization.

3.1.1 Alignment convergence

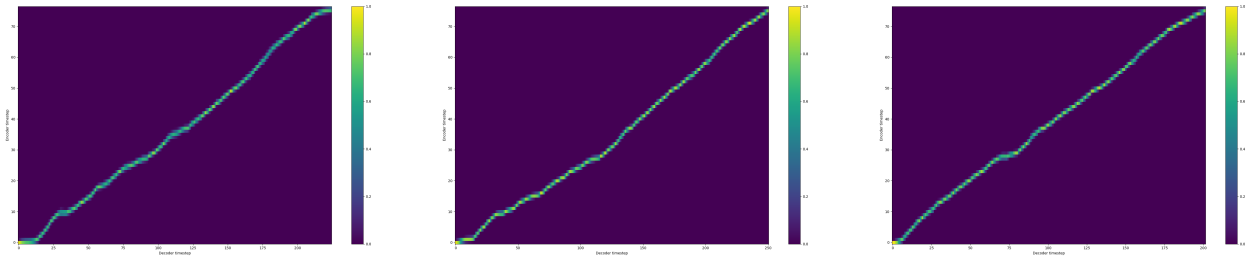
Tacotron 2 demonstrated no visible sensitivity to data sparsity. On all subsets, the attention mechanism converged to a clear diagonal alignment within 10k steps. While there were occasional minor misalignments during training, by the end of training, all models exhibited stable alignments on all test sentences.

As seen in Figure 11., the attention maps for Tacotron 2 remain stable across all data subsets, indicating robust alignment learning even in low-resource conditions.

Glow-TTS, utilizing MAS, also converged successfully across all three subsets. However, despite robust alignment, the audio reconstruction quality was significantly lower than Tacotron 2.

3.1.2 Pitch and Spectral Accuracy

Tacotron 2 consistently outperformed Glow-TTS in pitch accuracy (F0 RMSE), achieving values around 31 Hz compared to Glow-TTS's 36 Hz. Furthermore, Tacotron 2 achieved slightly lower MCD

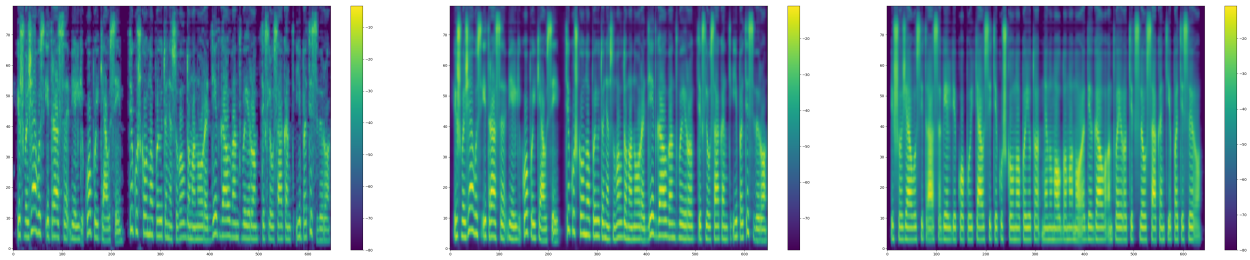


(a) *30 speakers* ***(b)*** *60 speakers* ***(c)*** *180 speakers*

Figure 11. *Example Tacotron 2 attention alignments after training on different data subsets. All configurations exhibit clear diagonal alignments, indicating successful convergence.*

scores on all subsets, suggesting it captures fine-grained spectral details better than the flow-based decoder. The raters noted that Glow-TTS outputs often sounded monotone and robotic, possibly due to Glow-TTS not having an explicit pitch predictor.

Spectral analysis confirms the objective metrics. As illustrated in the Mel-spectrogram comparison Figure 12., Tacotron 2 captures visibly more fine-grained spectral details and intonation contours more accurately than Glow-TTS, which tends to produce flatter, less dynamic outputs.



(a) *Ground truth* ***(b)*** *Tacotron 2 output* ***(c)*** *Glow-TTS output*

Figure 12. *Comparison of ground truth Mel-spectrogram (a), Tacotron 2 (b) and Glow-TTS (c) outputs for the same speaker and input text. Tacotron 2 captures more intonation variability, while Glow-TTS produces flatter contours.*

3.2 Subjective evaluation (MOS)

Naturalness was evaluated via a Latin square design listening test with 21 native speakers, who rated samples from all six experimental models plus the ground truth on a 5-point MOS scale. The results are presented in Table 10.

These results are in line with the objective findings — Tacotron 2 outperforms Glow-TTS in naturalness across all data strategies. Interestingly, the models exhibited very little sensitivity to data composition in terms of MOS. In both architectures, the 180-speaker configuration resulted in a slight drop in naturalness compared to the other two, however, the differences were not statistically significant.

3.2.1 Architecture comparison

The MOS results reveal a significant performance gap between the two architectures.

Table 10. *5-point MOS results with 95% Confidence Intervals.*

Model	Speaker count (N)	MOS (95% CI)
Ground Truth	—	**4.84** $\pm$ 0.06
Tacotron 2	30	3.11 $\pm$ 0.16
	60	**3.12** $\pm$ 0.17
	180	3.03 $\pm$ 0.18
Glow-TTS	30	2.13 $\pm$ 0.12
	60	2.17 $\pm$ 0.15
	180	2.02 $\pm$ 0.14

Tacotron 2 achieved the highest synthesis quality in the study, with the 60-speaker configuration scoring 3.12 ± 0.17. Listeners noted that when Tacotron 2 works, it produces highly expressive prosody. Increasing the speaker count to 180 caused a moderate drop in naturalness to 3.03 ± 0.18.

Glow-TTS lagged behind Tacotron 2 in naturalness, with the 60-speaker configuration scoring 2.17 ± 0.15. The listeners reported that Glow-TTS seemed to have strong robotic artifacts, and was at times unintelligible. This was especially pronounced in the 180-speaker condition where the MOS dropped to 2.02 ± 0.14.

3.2.2 Optimal composition

There was no clear winner between the data strategies across both architectures. The 60-speaker configuration had a marginal (insignificant) edge over the 30-speaker setup in MOS for both models. Both models performed worse in the 180-speaker scenario, however, the reduction in quality was not statistically significant.

3.2.3 Speaker analysis

An analysis of individual speakers' MOS scores was conducted to identify any patterns related to speaker identity. Table 11. summarizes the average MOS per speaker across all models.

Table 11. *Average MOS per speaker across all models.*

Speaker ID	Tacotron 2 MOS	Glow-TTS MOS
AS009	4.17 ± 0.20	2.61 ± 0.23
IS031	3.26 ± 0.20	2.13 ± 0.19
IS038	3.48 ± 0.21	2.50 ± 0.19
MS052	2.26 ± 0.17	1.87 ± 0.16
VP131	2.43 ± 0.19	1.93 ± 0.16
VP427	2.92 ± 0.22	1.59 ± 0.14

Notably, speaker **AS009** consistently received the highest ratings across both models (Tacotron 2 MOS: 4.17, Glow-TTS MOS: 2.61), suggesting that certain speaker or dataset characteristics (e.g., clear articulation, consistent recording quality, or clean transcripts) may facilitate

better synthesis quality. On the other hand, speakers like **MS052** and **VP427** scored significantly lower, indicating potential data quality issues or inherent speaker traits (e.g., strong accents, background noise) that challenge the TTS models.

3.3 Discussion

The results provide several insights into the effects of data composition and model architecture on multi-speaker TTS quality in low-resource settings.

Firstly, the autoregressive Tacotron 2 consistently outperformed the non-autoregressive Glow-TTS in both objective (F0 RMSE, MCD, PESQ) and subjective (MOS) metrics, across all dataset configurations. The 180-speaker scenario caused a slight to moderate quality drop for both models, but Tacotron 2 was able to produce intelligible speech.

The effects of data composition were very subtle. Both models showed only minor variations in performance across the three data strategies, with no statistically significant differences in MOS between the 30-speaker and 60-speaker configurations, and only a 0.09 (Tacotron 2) to 0.15 (Glow-TTS) point drop in MOS for the 180-speaker setup. This suggests that within the tested range, the total amount of data (22.5 hours) is more critical than the specific balance between speaker diversity and data density per speaker.

Finally, speaker-specific analysis revealed that certain speakers consistently yielded significantly higher synthesis quality across models and data strategies, indicating that speaker characteristics and data quality play a crucial role in TTS performance.

4 Conclusion

This thesis set out to determine the optimal data selection strategy for training multi-speaker TTS models for the Lithuanian language under a fixed resource budget. By systematically varying the number of speakers and the amount of data per speaker, the performance of one autoregressive (Tacotron 2) and one non-autoregressive (Glow-TTS) architecture was evaluated using both objective and subjective metrics.

4.1 Summary of findings

The experimental results only weakly support the hypothesis that data "depth" is a more critical factor than "breadth".

The autoregressive Tacotron 2 model consistently outperformed the non-autoregressive Glow-TTS model across all data configurations. Tacotron 2 achieved a peak MOS score of 3.48 in the 60-speaker setup, while Glow-TTS lagged behind with a maximum MOS of 2.40 in the 30-speaker setup. Qualitatively, listeners noted that Tacotron 2 produced more natural and expressive speech, while Glow-TTS outputs were often described as robotic and "scratchy".

4.2 Contributions

This work contributes to the field of speech synthesis in the following ways:

4.3 Limitations of the study

The study has several limitations that should be acknowledged.

Fistly, the experiments were conducted solely on read speech from the Lithuanian Liepa 2 corpus. Therefore, the findings may not generalize to other languages, datasets, or speech styles.

Secondly, only 22.5-hour training data size is a practical constraint and may not reflect performance at other data scales, such as very low-resource (e.g., 5 hours) or high-resource (e.g., 100+ hours) scenarios.

Additionally, only two TTS architectures were evaluated. While Tacotron 2 and Glow-TTS are widely used, other TTS systems may exhibit different sensitivities to data composition.

Finally, the subjective evaluation relied on a relatively small sample of 21 raters. While the confidence intervals were sufficient to draw conclusions about model differences, a larger-scale listening test would provide higher statistical power to the findings.

4.4 Future work

There are several potential directions for future research building on this study.

Firstly, replicating the experiments on different languages and datasets would help validate the generalizability of the findings.

Exploring a wider range of data budgets could provide insights into how data composition effects scale with more or less data. For instance, testing with a 10-hour or 50-hour budget could reveal different trade-offs.

Exploring additional architectures, such as VITS or FastSpeech 2, could provide further insights into how data composition affects different model types.

The evaluation could be expanded to include intelligibility metrics (e.g., WER) to assess how data composition impacts not just naturalness, but also clarity of the synthesized speech.

Finally, a deeper analysis of speaker characteristics (e.g., accent, speaking style) and their influence on synthesis quality could inform better data selection strategies for multi-speaker TTS training.

5 References

[1] H. Dudley, T. H. Tarnoczy. “The Speaking Machine of Wolfgang von Kempelen.” In: *The Journal of the Acoustical Society of America* 22.2 (1950), pages 151–166.

[2] J. Shen, R. Pang, R. J. Weiss, M. Schuster, et al. “Natural TTS Synthesis by Conditioning Wavenet on MEL Spectrogram Predictions.” In: *International Conference on Machine Learning* (2018), pages 4779–4788.

[3] J. Kim, S. Kim, J. Kong, S. Yoon. *Glow-TTS: A Generative Flow for Text-to-Speech via Monotonic Alignment Search*. 2020. URL: https://arxiv.org/abs/2005.11129.

[4] M. B. Hoy. “Alexa, Siri, Cortana, and More: An Introduction to Voice Assistants.” In: *Medical Reference Services Quarterly* 37.1 (2018). PMID: 29327988, pages 81–88. https://doi.org/10.1080/02763869.2018.1404391. URL: https://doi.org/10.1080/02763869.2018.1404391.

[5] I. Isewon, J. Oyelade, O. Oladipupo. “Design and implementation of text to speech conversion for visually impaired people.” In: *International Journal of Applied Information Systems* 7.2 (2014), pages 25–30.

[6] P. Taylor. *Text-to-Speech Synthesis*. Cambridge University Press, 2009.

[7] K. Ito, L. Johnson. *The LJ Speech Dataset*. https://keithito.com/LJ-Speech-Dataset/. 2017.

[8] Vilnius University. *Lietuvių šneka valdomų paslaugų plėtra - LIEPA 2 (Development of Services Controlled by Lithuanian Speech - LIEPA 2)*. Project funded by the European Regional Development Fund. Available at https://xn--ratija-ckb.lt/liepa-2/apie-projekta-liepa-2/. 2020.

[9] C. E. Shannon. “Communication in the Presence of Noise.” In: *Proceedings of the IRE* 37.1 (1949), pages 10–21.

[10] S. Jothilakshmi, V. Gudivada. “Chapter 10 - Large Scale Data Enabled Evolution of Spoken Language Research and Applications.” In: *Cognitive Computing: Theory and Applications*. Edited by V. N. Gudivada, V. V. Raghavan, V. Govindaraju, C. Rao. Volume 35. Handbook of Statistics. Elsevier, 2016, pages 301–340. https://doi.org/https://doi.org/10.1016/bs.host.2016.07.005. URL: https://www.sciencedirect.com/science/article/pii/S0169716116300463.

[11] D. Gabor. “Theory of communication. Part 1: The analysis of information.” In: *Journal of the Institution of Electrical Engineers - Part III: Radio and Communication Engineering* 93 (26 1946), pages 429–441. https://doi.org/10.1049/ji-3-2.1946.0074. URL: https://digital-library.theiet.org/doi/abs/10.1049/ji-3-2.1946.0074.

[12] S. S. Stevens, J. Volkmann, E. B. Newman. “A Scale for the Measurement of the Psychological Magnitude Pitch.” In: *The Journal of the Acoustical Society of America* 8.3 (1937), pages 185–190.

[13] R. Sproat, A. Black, S. Chen, S. Kumar, M. Ostendorf, C. Richards. “Normalization of Non-Standard Words.” In: *Computer Speech and Language* 15 (2001), pages 287–333. https://doi.org/10.1006/csla.2001.0169.

[14] M. Bisani, H. Ney. “Joint-sequence models for grapheme-to-phoneme conversion.” In: *Speech Communication* 50.5 (2008), pages 434–451. ISSN: 0167-6393. https://doi.org/https://doi.org/10.1016/j.specom.2008.01.002. URL: https://www.sciencedirect.com/science/article/pii/S0167639308000046.

[15] P. Kasparaitis, D. Antanavičius. “Investigation of Input Alphabets of End-to-End Lithuanian Text-to-Speech Synthesizer.” In: *Baltic Journal of Modern Computing* 11 (2023). https://doi.org/10.22364/bjmc.2023.11.2.05.

[16] V. M. University. *Kirčiuoklis*. URL: https://kalbu.vdu.lt/mokymosi-priemones/kirciuoklis/ (viewed 2025-12-07).

[17] T. Mikolov, K. Chen, G. Corrado, J. Dean. *Efficient Estimation of Word Representations in Vector Space*. 2013. URL: https://arxiv.org/abs/1301.3781.

[18] A. Jawaid, S. S. Chandra, J. Lu, B. Sisman. *Style Mixture of Experts for Expressive Text-To-Speech Synthesis*. 2024. URL: https://arxiv.org/abs/2406.03637.

[19] Y. Wang, R. Skerry-Ryan, D. Stanton, Y. Wu, et al. “Tacotron: Towards End-to-End Speech Synthesis.” In: *Interspeech*. 2017, pages 4006–4010.

[20] H. Dudley, R. Riesz, S. Watkins. “A synthetic speaker.” In: *Journal of the Franklin Institute* 227.6 (1939), pages 739–764. ISSN: 0016-0032. https://doi.org/https://doi.org/10.1016/S0016-0032(39)90816-1. URL: https://www.sciencedirect.com/science/article/pii/S0016003239908161.

[21] A. J. Hunt, A. W. Black. “Unit selection in a concatenative speech synthesis system using a large speech database.” In: *1996 IEEE International Conference on Acoustics, Speech, and Signal Processing Conference Proceedings*. Volume 1. IEEE. 1996, pages 373–376.

[22] A. Black, N. Campbell. “Optimising Selection Of Units From Speech Databases For Concatenative Synthesis.” In: 1 (1996).

[23] K. Kuligowska, P. Kisielewicz, A. Włodarz. “Speech synthesis systems: Disadvantages and limitations.” In: *International Journal of Engineering and Technology(UAE)* 7 (2018), pages 234–239. https://doi.org/10.14419/ijet.v7i2.28.12933.

[24] H. Zen, K. Tokuda, A. W. Black. “Statistical parametric speech synthesis.” In: *Speech communication* 51.11 (2009), pages 1039–1064.

[25] K. Tokuda, Y. Nankaku, T. Toda, H. Zen, J. Yamagishi, K. Oura. “Speech Synthesis Based on Hidden Markov Models.” In: *Proceedings of the IEEE* 101.5 (2013), pages 1234–1252. https://doi.org/10.1109/JPROC.2013.2251852.

[26] J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, et al. *Scaling Laws for Neural Language Models*. 2020. URL: https://arxiv.org/abs/2001.08361.

[27] I. Sutskever, O. Vinyals, Q. V. Le. *Sequence to Sequence Learning with Neural Networks*. 2014. URL: https://arxiv.org/abs/1409.3215.

[28] A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, K. Kavukcuoglu. "Wavenet: A generative model for raw audio." In: *arXiv preprint arXiv:1609.03499* (2016).

[29] Y. Ren, Y. Ruan, X. Tan, T. Qin, S. Zhao, Z. Zhao, T.-Y. Liu. *FastSpeech: Fast, Robust and Controllable Text to Speech*. 2019. URL: https://arxiv.org/abs/1905.09263.

[30] E. Battenberg, R. Skerry-Ryan, S. Mariooryad, D. Stanton, D. Kao, M. Shannon, T. Bagby. *Location-Relative Attention Mechanisms For Robust Long-Form Speech Synthesis*. 2020. URL: https://arxiv.org/abs/1910.10288.

[31] J. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, Y. Bengio. *Attention-Based Models for Speech Recognition*. 2015. URL: https://arxiv.org/abs/1506.07503.

[32] A. Łańcucki. "FastPitch: Parallel Text-to-speech with Pitch Prediction." In: *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2021, pages 6588–6592.

[33] J. Kim, J. Kong, J. Son. "Conditional Variational Autoencoder with Adversarial Learning for End-to-End Text-to-Speech." In: *arXiv preprint arXiv:2106.06103* (2021).

[34] D. Griffin, J. Lim. "Signal estimation from modified short-time Fourier transform." In: *IEEE Transactions on Acoustics, Speech, and Signal Processing* 32.2 (1984), pages 236–243. https://doi.org/10.1109/TASSP.1984.1164317.

[35] J. Kong, J. Kim, J. Bae. "HiFi-GAN: Generative Adversarial Networks for Efficient and High Fidelity Speech Synthesis." In: *NeurIPS*. 2020.

[36] Coqui. *Coqui TTS*. https://github.com/coqui-ai/TTS. 2021.

[37] J. Yamagishi, C. Veaux, K. MacDonald. *CSTR VCTK Corpus: English Multi-speaker Corpus for CSTR Voice Cloning Toolkit (version 0.92)*. https://datashare.ed.ac.uk/handle/10283/3443. doi:10.7488/ds/2645. 2019.

[38] J. Lorenzo-Trueba, T. Drugman, J. Latorre, T. Merritt, B. Putrycz, R. Barra-Chicote, A. Moinet, V. Aggarwal. *Towards achieving robust universal neural vocoding*. 2019. URL: https://arxiv.org/abs/1811.06292.

[39] S. Arik, G. Diamos, A. Gibiansky, J. Miller, K. Peng, W. Ping, J. Raiman, Y. Zhou. *Deep Voice 2: Multi-Speaker Neural Text-to-Speech*. 2017. URL: https://arxiv.org/abs/1705.08947.

[40] Y. Jia, Y. Zhang, R. J. Weiss, Q. Wang, et al. *Transfer Learning from Speaker Verification to Multispeaker Text-To-Speech Synthesis*. 2019. URL: https://arxiv.org/abs/1806.04558.

[41] A. Nagrani, J. S. Chung, A. Zisserman. "VoxCeleb: A Large-Scale Speaker Identification Dataset." In: *Interspeech 2017*. ISCA, 2017. https://doi.org/10.21437/interspeech.2017-950. URL: http://dx.doi.org/10.21437/Interspeech.2017-950.

[42] E. Variani, X. Lei, E. McDermott, I. L. Moreno, J. Gonzalez-Dominguez. “Deep neural networks for small footprint text-dependent speaker verification.” In: *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2014, pages 4052–4056. https://doi.org/10.1109/ICASSP.2014.6854363.

[43] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, S. Khudanpur. “X-Vectors: Robust DNN Embeddings for Speaker Recognition.” In: *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2018, pages 5329–5333. https://doi.org/10.1109/ICASSP.2018.8461375.

[44] R. Kubichek. “Mel-cepstral distance measure for objective speech quality assessment.” In: *Proceedings of IEEE Pacific Rim Conference on Communications Computers and Signal Processing*. Volume 1. IEEE. 1993, pages 125–128.

[45] A. Rix, J. Beerends, M. Hollier, A. Hekstra. “Perceptual evaluation of speech quality (PESQ)-a new method for speech quality assessment of telephone networks and codecs.” In: *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.01CH37221)*. Volume 2. 2001, 749–752 vol.2. https://doi.org/10.1109/ICASSP.2001.941023.

[46] ITU-T. *P.800 : Methods for subjective determination of transmission quality*. Recommendation P.800. International Telecommunication Union, 1996.

[47] E. J. Williams. “Experimental Designs Balanced for the Estimation of Residual Effects of Treatments.” In: *Australian Journal of Chemistry* 2 (1949), pages 149–168. URL: https://api.semanticscholar.org/CorpusID:96306494.

[48] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems*. 2016. URL: https://arxiv.org/abs/1603.04467.

6 Appendix: GitHub repository with source code

The complete source code for data preprocessing, model training, inference, and evaluation is available in the GitHub repository:

`https://github.com/spacegrapefruit/msc-thesis`

The structure of the repository is as follows:

7 Appendix: Audio preprocessing parameters

The following Mel-spectrogram extraction parameters were used by all models (Tacotron 2, Glow-TTS, and HiFi-GAN vocoder):

Table 12. *Complete Mel-spectrogram extraction parameters.*

Parameter	Value
fft_size	1024
win_length	1024
hop_length	256
frame_length_ms	null
frame_shift_ms	null
stft_pad_mode	“reflect”
sample_rate	22050
resample	false
preemphasis	0.98
ref_level_db	20
do_sound_norm	false
log_func	“np.log10”
do_trim_silence	true
trim_db	60
do_rms_norm	false
db_level	null
power	1.5
griffin_lim_iters	60
num_mels	80
mel_fmin	0.0
mel_fmax	8000.0
spec_gain	20
do_amp_to_db_linear	true
do_amp_to_db_mel	true
pitch_fmax	640.0
pitch_fmin	1.0
signal_norm	true
min_level_db	-100
symmetric_norm	true
max_norm	4.0
clip_norm	true
stats_path	null