Degree Project in Machine Learning

Second cycle, 30 credits

Enhancing ASR performance on entities

Improving transcription accuracy of entities in speech recognition models by fine-tuning on synthetic data

PHILIP RETTIG

Enhancing ASR performance on entities

Improving transcription accuracy of entities in speech recognition models by fine-tuning on synthetic data

PHILIP RETTIG

Master's Programme, Machine Learning, 120 credits
Date: June 12, 2025

Supervisor: Jonas Beskow
Examiner: Joakim Gustavsson
School of Electrical Engineering and Computer Science
Host company: Tonar Technologies AB
Swedish title: Förbättrad transkription av entiter
Swedish subtitle: Ökad transkriptions-träffsäkerhet av entititer genom fine-tuning på syntetisk data.

Abstract

Automatic speech recognition (ASR) has significantly advanced in recent years. Despite these advancements, accurately transcribing rare and out-of-vocabulary words remains a challenge. To improve performance, large datasets annotated by experts are necessary. However, creating such datasets is expensive and raises privacy concerns. Simultaneously, synthetically generated data has advanced to be nearly indistinguishable from real human speech This thesis investigates whether fine-tuning ASR models on synthetic speech can improve their ability to accurately transcribe targeted words, in this case entities. Efforts in this field have been made and show promise. However, to our knowledge, not specifically using synthetic data to increase performance on targeted entities. Our method involved fine-tuning three Swedish-specific Whisper ASR models on high-quality synthetic datasets and evaluating the models' ability to recognize entities while maintaining Word Error Rate (WER) and Character Error Rate (CER).
Results indicate that fine-tuning with synthetic data improved transcription accuracy for targeted entities, however with a reduction of overall transcription accuracy. Although the increase in performance was incremental rather than groundbreaking, the method's validity and potential for scalability were demonstrated. Future work includes exploring different base models and architectures, both as base models for fine-tuning and for filtering the synthetic dataset, to improve generalizability to lower-resource languages, as well as developing a more cost-efficient pipeline, given this project's reliance on high-quality synthetic data. Ultimately, this project highlights the practical benefits and potential for overcoming privacy and performance challenges in speech recognition within specialized domains.

Keywords

Automatic Speech Recognition, Synthetic Speech, Text-to-speech, Whisper, Fine-Tuning, Domain-Adaptation, Speech-Technology

Sammanfattning

Automatisk taligenkänning har gjort betydande framsteg under de senaste åren. Trots dessa framsteg kvarstår utmaningar med att korrekt transkribera ovanliga ord. För att förbättra prestandan krävs stora dataset annoterade av experter, men att skapa sådana dataset är kostsamt och medför integritetsproblem. Samtidigt har syntetiskt genererad data utvecklats till en nivå där den är nästintill omöjlig att urskilja från verkligt mänskligt tal. Denna uppsats undersöker huruvida syntetiskt tal kan förbättra taligenkännings-modellers förmåga att korrekt transkribera entiteter. Tidigare forskning inom detta område har visat lovande resultat, men, så vitt vi vet, har syntetisk data ännu inte använts specifikt för att öka prestandan vid transkribering av utvalda entiteter.

Metoder innefattar fine-tuning av tre Whisper modeller, nyligen släppta av Kungliga Biblioteket samt utvärdering av modellernas förmåga att känna igen entiteter utan att försämra transkriptions-kvaliteten överlag. Resultaten visar att fine-tuning med syntetisk data förbättrar träffsäkerheten för de utvalda entiteter. Däremot försämras den generella transkriberingskvaliteten. Även om förbättringarna var marginella demonstrerade metoden sin potential. Framtida arbete inkluderar att undesöka olika basmodeller och arkitekturer både som grund för fine-tuning men också för utvärdering av den syntetiska datans kvalitet. Projektet i sin helhet belyser de pratiska fördelarna och potentialen för att övervinna integritets-och prestandautmaningar inom taligenkänning inom specialiserade domäner.

Nyckelord

Automatisk taligenkänning, Syntetiskt tal, Text-till-tal, Whisper, Fine-tuning, Domänanpassning, Talteknologi

Acknowledgments

I would like to thank Jonas Beskow at KTH and Tomas Lundberg at Tonar Technologies AB for supervising this project, providing valuable feedback and resources. Also thank you to examiner Joakim Gustafsson and participants in the presentation.

Stockholm, June 2025
Philip Rettig

Contents

1 Introduction **1**
1.1 Background . . . 1
1.2 Problem . . . 2
1.2.1 Research question & Objective . . . 2
1.3 Purpose . . . 3
1.4 Goals . . . 4
1.5 Research Methodology . . . 4
1.6 Delimitations . . . 5
1.7 Structure of the thesis . . . 5

2 Background **7**
2.1 Sound and representations . . . 7
2.1.1 Fourier Transform and Time-Frequency Analysis . . . 8
2.1.2 Log-Mel Spectrograms in Speech Processing . . . 9
2.2 Representation Learning . . . 9
2.2.1 Embeddings . . . 10
2.2.2 Transformers . . . 11
2.2.3 Contrastive Learning . . . 11
2.3 Speech technology applications . . . 12
2.3.1 Text-to-speech . . . 13
2.3.2 Automatic speech recognition . . . 13
2.3.2.1 From statistical approaches to deep-neural networks . . . 14
2.3.2.2 End-to-End models . . . 14
2.3.2.3 Domain Adaptation . . . 15
2.3.2.4 Wav2vec 2.0 . . . 16
2.3.2.5 Whisper . . . 17
2.3.2.6 Evaluation metrics for ASR-models . . . 17
2.3.2.7 Evaluation metrics for binary classification . 18

2.4 Related work . . . 18
2.4.1 When whisper meets TTS . . . 18
2.4.2 Contrastive Audio-Language Learning for Music . . . 18
2.4.3 Enhancing Automatic Speech Recognition: Effects of Semantic Audio Filtering on Models Performance . . . 19
2.5 Summary . . . 19

3 Pipeline overview **21**
3.1 Experiment Design . . . 21
3.1.1 Benchmarks . . . 22
3.1.2 Corpus Collection . . . 22
3.1.3 Text to Speech model . . . 24
3.1.4 Filtering Framework . . . 24
3.1.4.1 Loss function . . . 26
3.1.5 Fine tuning . . . 28
3.1.6 Evaluation . . . 29
3.2 Validity and Reliability . . . 30
3.2.1 Validity of method . . . 30
3.2.2 Reliability of method . . . 30
3.2.3 Data validity . . . 30
3.2.4 Reliability of data . . . 31

4 Implementation details **33**
4.1 Implementation pipeline . . . 33
4.1.1 Benchmark . . . 33
4.1.2 Corpus . . . 34
4.1.3 Creating synthetic dataset . . . 34
4.1.4 Applying filtering model . . . 36
4.1.5 Fine-tuning . . . 36
4.1.6 Dataset . . . 36
4.1.7 Hardware and software used . . . 37

5 Results and Analysis **39**
5.1 Fine-tuning . . . 39
5.1.1 Training . . . 39
5.1.2 Evaluation . . . 42
5.2 Reliability Analysis . . . 47
5.3 Validity Analysis . . . 47
5.4 Discussion . . . 48

6 Conclusions and Future work **49**
6.1 Conclusions 49
6.2 Future work 50
6.2.1 Cost analysis 50
6.2.2 Next obvious things to be done 51

References **53**

List of Figures

2.1 Soundwave, Fourier transformed and Mel-Spectrogram 10
2.2 Illustration of word embeddings 11
2.3 Transformers architecture. With permission to reproduce from the authors at Google [17] 12
2.4 Wav2Vec 2.0 16

3.1 High level experiment overview 21
3.2 Pipeline for collecting corpus, extracting entities and creating the synthetic dataset 25
3.3 The Filtering framework 26
3.4 Training the filtering framework 27
3.5 Fine-tuning the models 29
3.6 Evaluating our collecting or fine-tuned models 29

4.1 Distribution of entities in terms of categories 37

5.1 Training and validation loss curves for the three models. From left to right, small, medium, large. 40
5.2 Training and validation loss curves for the three models trained on both the synthetic dataset and fleurs. From left to right, small, medium, large. 41
5.3 Training and validation loss curves for the three models trained on both the synthetic dataset and commonvoice. 42

List of Tables

3.1 Caption 23
3.2 Example of the collected corpus with extracted entities 23
3.3 Example of the collected corpus with replaced entities 24

4.1 Entity accuracy for Common Voice and Fleurs 34

5.1 Entity accuracy and Word Error Rate (WER) for Common Voice and Fleurs, with separate groupings for un-normalized and normalized values. The models are fine-tuned solely on synthetic data. Models that show improvement in comparison to the base-models are marked with **bold**. Datsasets contain sentences in commonvoice and fleus that the transcription model struggle to recognize. 43
5.2 Entity accuracy and Word Error Rate for models trained on datasets consisting of synthetic and real data. Comparing non-normalized and normalized results. Models fine-tuned on training data containing data from the fleurs dataset was evaluated on the commonvoice dataset and vice versa. **Bold** indicates improvement in comparison to the base-models . . . 44
5.3 Normalized and non-normalized character error rate (CER) for models trained on datasets consisting of varuous combinations synthetic and real data, evaluated on the subsets of CommonVoice and Fleurs. 45
5.4 Normalized and non-normalized character error rate (CER) within the extracted entities from the sentences for models trained on datasets consisting of variuous combinations synthetic and real data, evaluated on the subsets of CommonVoice and Fleurs. The fine-tuned models that showed improvement in comparison to the base-models are marked in **bold** 46

5.5 WER comparison of base and tuned models on Common Voice (CV) and Fleurs. Lower scores indicate better performance. Bold indicates best result within a row. 47

Chapter 1

Introduction

Automatic speech recognition (ASR) is considered one of the most important areas in Artificial Intelligence research [1]. Significant progress has been made in this area, and state-of-the-art performance has been achieved in several scenarios. However, when words appear infrequently in the systems training data it leads to poor performance on out of vocabulary words, particularly in minority languages [2]. The success of deep learning in speech and speaker recognition heavily relies on the use of large datasets. Due to the scarcity of high-quality annotated speech data in specific domains, practitioners often resort to using pre-trained models fine-tuned on domain-specific datasets. This approach presents a significant challenge in accurately transcribing rare phrases and out-of-vocabulary named entities. Moreover, industries with domain-specific vocabulary often face data shortages due to privacy concerns [1]. Simultaneously, text-to-speech conversion has advanced to the point where, in many cases, synthetic speech is nearly indistinguishable from human speech. This synthetic speech shows promise as training data for fine-tuning ASR models [1] [3], and in this project, we adopt a filtering framework developed by [4] to improve synthetic data quality and fine-tune several pre-trained ASR models with the ultimate goal of enhancing performance on unseen entities in Swedish.

1.1 Background

Recently ASR have experienced significant advancements through the use of End-to-End (E2E) models. Notable developments include Transformer-based models like Wav2Vec2.0 [5], Conformer networks, and fully supervised

models such as Whisper [6], which have set new benchmarks in ASR performance. Despite these achievements, the field still faces considerable challenges. State of the art ASR models often depend on large volumes of transcribed speech for effective training. This becomes a problem in areas like healthcare, forensics, and government, where data is either scarce or restricted due to privacy concerns [1]. In fields with limited data and specialized vocabulary, state-of-the-art models often underperform. To address this challenge, a prevalent strategy is Domain Adaptation (DA). This process entails training a machine learning model on data from one domain to perform effectively on data from another, distinctly different domain [7]. The limitations are exacerbated in environments with uncontrolled acoustics or specialized vocabularies. The challenge is even more pronounced for low-resource languages, which lack extensive labeled datasets needed for training. Even using domain adaptation, data collection and augmentation is an expensive, time-consuming process. Recent studies consider a different approach, namely increasing training data size artificially using state of the art text-to-speech (TTS) models to generate synthetic data. This approach is particularly useful when dealing with out-of-vocabulary words and domain-specific language [1].

1.2 Problem

Collecting training data for ASR models is time-consuming and expensive, particularly in low-resource languages and private domains [1]. Furthermore, ASR models struggle with entities and out-of-vocabulary words. Language models frequently fail to recognize rare or ambiguous entities due to limitations in training data and contextual modeling. Recent work has explored the use of query domain classification to better tailor language models to specific application contexts and the incorporation of external knowledge graphs to reinforce entity relations during decoding. However, these approaches are constrained by real-time processing requirements and struggle with the long-tail distribution of entities, resulting in a persistent feedback loop of recognition errors [8]. Improving entity accuracy for ASR models would be beneficial for several domains.

1.2.1 Research question & Objective

Can synthetic data improve accuracy on entities in speech recognition? The hypothesis presented is that synthetic data can be used to create

specialized ASR models with enhanced ability to recognize entities. However, in the event of success the proof-of-concept extends to general domain adaptation of ASR models. The objective is to improve entity recognition on Entities in ASR performed by the Swedish National Library's KBLabs Whisper models. We will also evaluate the WER. If we observe an improvement in Entity Recognition while maintaining WER, it indicates that synthetic data can indeed be used as training data for out-of-vocabulary words, without overfitting to synthetic data. Regardless of success or failure in improving these metrics, the findings will provide valuable insights on using synthetic data in the speech domain, thereby contributing to ongoing research.

To achieve this objective, we will perform three major tasks. First, we need to generate a dataset consisting of sentences containing entities, along with corresponding synthetic speech. The challenge here is to find alignment between the text and audio. We must find a speech synthesis engine that produces high-quality audio in the dataset, preferably one accessible via an API or with open-source weights. Machine learning with synthetic data presents challenges due to discrepancies between synthetic and real data distributions. Synthetic datasets can include artifacts absent in real data, such as structured noise, content inaccuracies, or unnatural speaking styles. For effective learning with synthetic data we need as small a gap as possible between the synthetic data and the true distribution. [3] The second task is to perform filtering on the generated audio. To increase the quality of the data we use an approach proposed by [4] which includes training a cross-modal embedding model based on a contrastive learning loss function to push similar pairs closer together, and dissimilar pairs further apart. The final task is to fine-tune the Whisper models on the dataset and evaluate the results. Since synthetic speech may introduce artifacts, such as unnatural speaking styles and the absence of background noise, fine-tuning only the decoder, similar to language model adaptation, allows the model to learn token representations of out-of-vocabulary words without overfitting to synthetic acoustic properties [1].

1.3 Purpose

Achieving the goals of this project will primarily benefit developers and practitioners in Automatic Speech Recognition (ASR), particularly those working with low-resource languages and specialized domains such as healthcare, forensics, and government. Improved accuracy in entity

recognition through synthetic data would enhance the reliability and effectiveness of ASR systems, enabling more precise transcription of critical or sensitive information.

1.4 Goals

The primary goal of this project is to investigate whether synthetic audio can improve transcription quality for named entities. This includes analyzing how the quality of training data impacts fine-tuning outcomes. The project is structured around the following three sub-goals:

1. **Industry Relevance and Academic Contribution:** This study investigates whether state-of-the-art TTS models can generate high-quality synthetic data suitable for fine-tuning base ASR models to create specialized ASR systems. Our analysis extends to other scenarios requiring the development of domain-specific ASR models.

2. **Educational Objective:** Successfully completing this project will fulfill the requirements of the course.

The main deliverables to achieve these goals are broken down into sub-tasks and further described in Chapter 4

1.5 Research Methodology

We begin by creating a relevant corpus to act as foundation for the synthetic dataset. Corresponding synthetic audio samples are produced using a state-of-the-art text-to-speech engine available through an API. To ensure dataset quality, a filtering framework based on the method proposed by [4] is applied. This framework leverages two pre-trained encoders with publicly available weights hosted on Hugging Face to discard low-quality text-audio pairs. The final dataset, used for ASR model fine-tuning, is entirely composed of publicly available resources. The base ASR model selected for fine-tuning is an open-source model available on Hugging Face. Fine-tuning is performed on the curated dataset to optimize entity recognition accuracy. Fine tuning is conducted utilizing parameter efficient fine-tuning (PEFT), described in [1] and [9]. The effectiveness of the approach is evaluated by measuring the ASR model's ability to correctly transcribe named entities, using standard accuracy metrics. This research aligns with prior studies

exploring ASR improvements using synthetic data [4] [1], leveraging publicly available models and datasets to ensure reproducibility. The use of open-source components allows other researchers to replicate and extend this work, contributing to ongoing advancements in ASR performance. Throughout the project, as we use several probabilistic machine learning models manual verification is conducted.

1.6 Delimitations

This project encompasses three distinct tasks: (1) generating a dataset of synthetic Swedish speech data, (2) training a filtering model to ensure alignment between text and audio, and (3) fine-tuning Whisper on this dataset to evaluate its effectiveness. Given this broad scope, one of the primary limitations is the need to control multiple factors, such as data quality, model configurations and hyperparameters, when interpreting the results. To further refine the scope, this study is strictly limited to the Swedish language. The dataset consists exclusively of Swedish text-audio pairs, the filtering model is trained on Swedish data, and the Whisper model selected for fine-tuning is pre-trained on Swedish speech. This linguistic constraint ensures a focused evaluation but may limit the generalizability of the findings to other languages.

1.7 Structure of the thesis

Chapter 2 presents an extensive overview of fundamental concepts and the development of Text2Speech, Automatic Speech Recognition, and Multimodal CLIP-models. Chapter 3 presents the methodology and method used to solve the problem. Chapter 4 describes the process of implementation along with all settings needed to reproduce the experiment. Chapter 5 walks through the results the experiment yielded and the last chapter discusses learnings and future work.

Chapter 2

Background

This chapter provides basic background information about the mechanisms of speech and hearing, Automatic Speech Recognition and Speech Generation models. Additionally, this chapter describes the history of ASR and the significant impact of deep learning approaches. The chapter also describes related work [4].

2.1 Sound and representations

We produce and perceive speech constantly without reflecting much over the mechanisms enabling us to do so, but in order to replicate these functions in computers we need to understand the underlying process. Many Automatic Speech Recognition approaches are more or less inspired by human speech perception [10]. As described in [11], speech is produced when air from the lungs flows through the open or vibrating vocal cords, forming the basic sound source. For voiced sounds, vocal cord vibrations create pulses of air. The vocal cords vibrate at different rates, ranging from 60 Hz for a large man to 300 Hz for a small woman or child, often referred to as the fundamental frequency. The lips, tongue, jaw, and velum then shape this airflow to produce various resonances. These rapid movements continuously blend to form smooth, intelligible speech which propagates through air as pressure waves. The ear captures these waves in the cochlea and converts into neural signals our brain processes and interprets.

Sound is fundamentally a wave phenomenon transmitted through a medium (air, liquid, or solid) as a vibration [12]. It can be characterized by its amplitude and frequency. The amplitude of the pressure wave corresponds to the perceived loudness, while the frequency corresponds to the perceived

pitch. Speech signals can be decomposed into sums of sinusoids [13], defined as

$$x(t) = A\cos(2\pi ft + \phi)\,, \tag{2.1}$$

where A is the amplitude, f is the frequency in hertz (Hz), and ϕ is the phase. To enable digital processing of an analog signal, the continuous analog waveform must be converted to a digital signal by sampling and quantization. In sampling, the waveform is measured at discrete time intervals (e.g., 8 kHz sampling rate for telephone applications or 44.1 kHz sampling rate for high fidelity audio applications), producing a sequence $x[n]$ of samples that approximates the continuous signal. This discrete-time representation makes the audio signal suitable for digital processing, [13].

2.1.1 Fourier Transform and Time-Frequency Analysis

A powerful tool for analyzing sound signals is the **Fourier transform**, which decomposes a time-domain signal into its constituent frequencies. The Fourier transform provides a frequency-domain representation $X(f)$ that indicates the amplitude and phase of each sinusoidal component in the signal. [14] defines the continuous-time Fourier transform (FT) of a signal $x(t)$ in mathematical terms as:

$$X(f) = \int_{-\infty}^{\infty} x(t)\, e^{-j2\pi ft}\, dt\,, \tag{2.2}$$

For implementation in computers, a discrete counterpart is used. The discrete Fourier transform (DFT) operates on a finite window of N samples $x[0], x[1], \ldots, x[N-1]$ and produces complex coefficients N. The k-th DFT coefficient is

$$X[k] = \sum_{n=0}^{N-1} x[n]\, e^{-j2\pi \frac{nk}{N}}\,, \quad k = 0, 1, \ldots, N-1\,, \tag{2.3}$$

which yields a spectrum $X[0 \ldots N-1]$ [14] [13], which show the amplitude at each frequency at a single time point. Simplified, the Fourier transform converts a signal into a representation that depends on frequency, rather than time and usually the frequency domain aligns best with a perceptual understanding of audio processing. [14] [13].

The Fourier Transformation assume stationarity, which is problematic as speech signals are non-stationary. To capture the evolution of the frequency content, a Short-Time Fourier Transform (STFT) is used. The STFT computes

the Fourier transform over short, typically overlapping, time frames, resulting in a time-frequency representation. In practice, the signal is divided into frames by applying a window function $w[n]$ of length N and a DFT is applied to each frame. Mathematically, for a discrete signal $x[n]$, the STFT at frame index h and frequency bin k is:

$$X(h,k) = \sum_{n=0}^{N-1} x[h+n]\, w[n]\, e^{-j2\pi \frac{nk}{N}}\,, \tag{2.4}$$

which yields $X(h,k)$ as the spectrum of the h-th frame. The collection of spectrums $|X(h,k)|$ plotted as a function of time h and frequency k produces a spectrogram. A spectrogram visualizes how spectral energy density evolves over time and time-varying resonances or patterns become visible [13].

2.1.2 Log-Mel Spectrograms in Speech Processing

Although a linear-frequency spectrogram contains comprehensive information, it can be redundant and not optimally aligned with human auditory perception. Based on experiments, humans perceive frequency on a roughly logarithmic scale; we are more sensitive to differences at lower frequencies than at higher frequencies. The mel scale is a perceptual frequency scale that reflects this property: it is linear in frequency below about 1 kHz and logarithmic above and is supposed to more accurately model the sensitivity of our ear [13].

Using this scale, a set of J bandpass filters (often triangular-shaped and overlapping) is designed so that their central frequencies and bandwidths are evenly spaced on the mel scale. This filter bank is applied to the linear power spectrum of each STFT frame to produce a mel spectrogram. Taking the logarithm of these energies yields the log-mel spectrogram for the frame and stacking them for all time frames gives the full log-mel spectrogram and features derived from log mel spectrograms are today standard practice when it comes to digital speech processing [13].

2.2 Representation Learning

Machine Learning algorithms can generally be divided into supervised learning and unsupervised learning. Supervised learning learns from labeled datasets, observing several examples of a random vector $X = [x_1, x_2, ..., x_n]$ learning to predict y from X, typically by estimating $P(y|x)$. Unsupervised

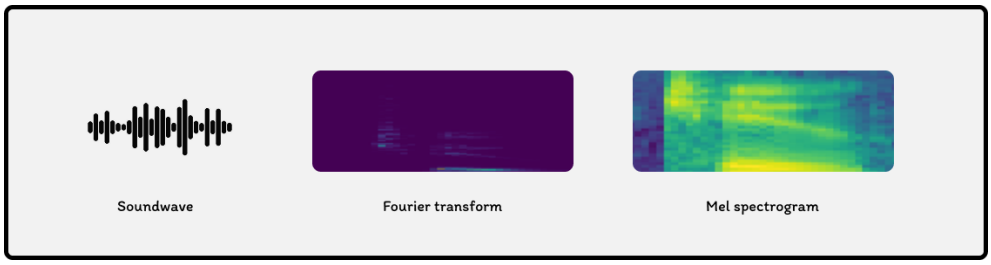

Figure 2.1: Soundwave, Fourier transformed and Mel-Spectrogram

learning however, learns from unlabeled datasets. It involves implicitly or explicitly learning the probability distribution that generated a dataset. Representation Learning facilitates the use of representations of the data learned from unsupervised learning to perform supervised tasks, such as regression and classification. It is common to have large amounts of unlabeled data compared to labeled data, which makes representation learning particularly useful [15].

2.2.1 Embeddings

As previously mentioned, to perform downstream tasks on data comprehensible to humans one must often extract representations of enabling computers to do so. Embeddings facilitate this by representing complex sequences as vectors within a multi-dimensional space. This representation allows computers to capture specific features and attributes, ranging from semantic meanings to acoustic properties. By mapping complex structures in this manner, similarities can be assessed based on spatial proximity. Such representations empower machine learning models to perform various tasks, from question answering to machine translation [4].

There are several different kinds of embeddings. Word embeddings are the process of encoding words as vectors, enabling machines to compare spatial proximity and perform downstream tasks. Initially, word embeddings treated words as independent entities, which led to an inability to capture the semantic relationships between words.

Audio embeddings encode audio as vectors based on the same principles as word embeddings. As discussed, several features can be extracted from a raw audio signal, represented as a sinusoidal wave in the time, frequency, or cepstral domain [13]. These features are handcrafted relying on psychoacustic experts and may not generalize well across diverse tasks, prompting the exploration of self-supervised learning to derive relevant features directly from

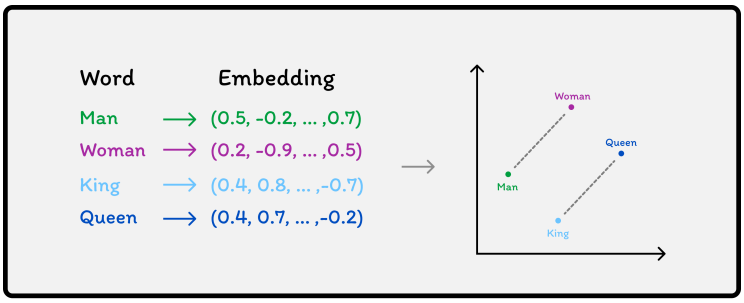

Figure 2.2: Illustration of word embeddings

raw data [4] [16] . Self-supervised learning learns meaningful representations from unlabeled data by generating its own labels, reducing the need for large amounts of labeled data. An example of this is Wav2Vec 2.0, developed by [5], which masks parts of the raw audio and trains the model to predict the masked parts based on the surrounding context. Wav2Vec 2.0 is based on the transformer architecture, as detailed in [17], and the transformer encoders ability to consider context marked a significant shift in the field.

2.2.2 Transformers

The transformer architecture relies on attention, a mechanism that enables the model to focus on the most relevant parts of the input sequence by assigning different weights to different elements. Introduced in 2017 by [17], in terms of speech technology, the transformer provided the capability to handle higher-dimensional data, and by utilizing self-attention layers, it could capture long-term dependencies in input speech, thereby outperforming previous models. The transformer led to the development of sophisticated models for embeddings, such as the Bidirectional Encoder Representations from Transformers (BERT) [18]. BERT is uniquely designed to analyze text by considering the bidirectional context, effectively integrating information from both the preceding and following text in a sentence and to enable easy fine-tuning, leading to BERT being the foundation of many state-of-the-art models [4].

2.2.3 Contrastive Learning

Given the scarcity of high-quality labeled data, models need to learn meaningful representations from unlabeled data. Building upon embeddings in a multi-dimensional latent space, contrastive learning (CL) achieves this

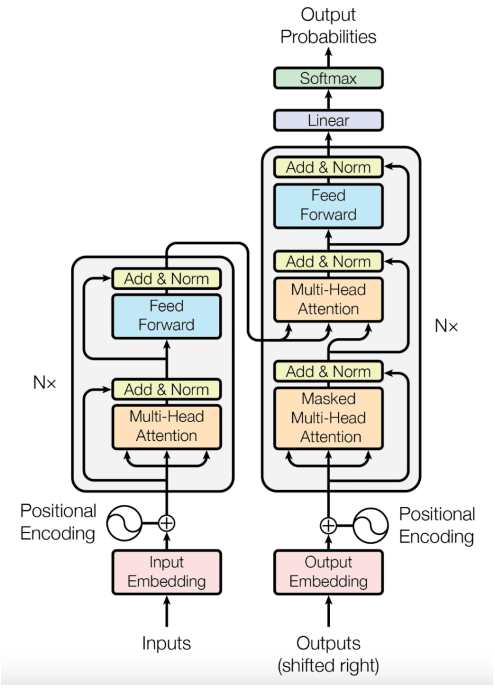

Figure 2.3: Transformers architecture. With permission to reproduce from the authors at Google [17]

by clustering similar samples together and distancing dissimilar ones. At its core contrastive learning construct different versions of the same data, aiming to push similar embeddings closer together and dissmilar embeddings further away to make the features more robust and effective for downstream tasks [19]. The latent space can organize data of various formats. This process is referred to as unimodal and multimodal contrastive learning. Unimodal CL is a representation learning technique in which the method learn the inherent structure in each modality to learn without external labels, altering the data slightly. In imaging it can for example crop or change colors. [4]. Multimodal contrastive learning leverage the fact that differences in the data occur naturally (e.g the difference between audio and text), making it an ideal setting for contrastive learning [19].

2.3 Speech technology applications

This sections walks through fundamentals and state of the art approaches in Text-to-Speech and a brief overview of the history of automatics speech recognition.

2.3.1 Text-to-speech

Text-to-Speech (TTS) technology has a wide range of applications, particularly in fields like medicine and education. In recent years, it has evolved significantly, moving beyond robotic-sounding voices to produce highly sophisticated, natural-sounding speech. In 2017, [20] introduced the Tacotron. Traditionally TTS systems require separate components for text analysis, acoustic modeling, and waveform synthesis, each demanding extensive domain expertise. Tacotron employs a sequence-to-sequence model with attention, allowing it to be trained from scratch with text-audio pairs. The encoder extracts robust text representations, while the decoder predicts spectrogram frames sequentially, which are then converted into audio waveforms. The model achieves a 3.82 mean opinion score (MOS), outperforming a production parametric system in naturalness. Experiments demonstrate that Tacotron effectively learns alignments between text and speech without requiring phoneme-level supervision.

In continuation of the work, [21] introduced Tacotron 2, replacing the Griffin-Lim algorithm with a WaveNet vocoder and prediciting a mel spectrogram rather than a magnitude spectrogram for feature representation, Tacotron 2 marked a significant breakthrough in text-to-speech and achieved a mean opinion score (MOS) of 4.53. Professionally recorded speech benchmarks at 4.58.

With today's advanced text-to-speech (TTS) capabilities, which enable the generation of speech that is nearly indistinguishable from human speech, and given the significant challenges and costs associated with creating high-quality text-audio paired datasets, a new application area has emerged. TTS is now being utilized to generate synthetic speech datasets for training speech recognition models [1] [4] [3].

2.3.2 Automatic speech recognition

Research into converting audio signals into other formats has a long history, beginning in 1952 when [22] developed a recognizer that accurately identified digits 0-9 with 97-99% accuracy. Their work focused on recognizing phonemes from a single speaker using formant frequency analysis, measuring the resonance frequencies of the vocal tract to characterize vowel sounds. This laid the groundwork for understanding phonetic elements of speech and their acoustic representations [23]. In the 1960s and 1970s technology aiding the recognition of isolated words and connected speech, like filter-bank analysis and progressed the field further.

2.3.2.1 From statistical approaches to deep-neural networks

In the 1980s, statistical methods such as Hidden Markov Models (HMMs), Gaussian Mixture Models (GMMs) and Linear Predictive Coding revolutionized the field by managing the inherent variability in speech [23]. While yielding impressive results, their inability to effectively model non-linear relationships prompted researchers to explore other approaches. This led to the adoption of methods capable of representing complex, non-linear functions, such as Deep Neural Networks (DNNs). Initially, DNNs were employed to extract features for HMMs and GMMs. Researchers used a variety of architectures ranging from Convolution Neural Networks (CNNs) to Long Short-term Memory networks hybrid modeling which significantly improved ASR models [4]. Recently, however, ASR models have transitioned from hybrid to end-to-end modeling, achieving state-of-the-art performance on nearly all accuracy benchmarks [24]. In this chapter, we will discuss several state-of-the-art ASR model architectures, ultimately leading to Whisper and Wav2Vec, for which we aim to improve accuracy.

2.3.2.2 End-to-End models

End-to-end models, in contrast to hybrid models, directly transform an input speech sequence into a token sequence via a singular network. This approach offers several significant advantages. Firstly, they employ a unified objective function, facilitating the optimization of the entire network rather than just individual components, as is common in hybrid architectures. These models produce character-level outputs, which present a more intuitive format for individuals lacking ASR expertise. Finally, E2E models are more compact and can be deployed on devices maintaining accuracy with low inference time [24].

End-to-end (E2E) models that utilize Recurrent Neural Networks (RNNs), which are specialized for processing sequential data [15], have achieved significant success in speech recognition. This success is largely attributed to their ability to model temporal dependencies [4]. However, RNNs encounter a notable limitation known as the vanishing gradient problem. This issue arises when the sequences being processed are lengthy, causing the gradients in backpropagation to diminish. Consequently, it becomes challenging for RNNs to maintain and retain information when sequences increase in length [15]. Thereby, the Long Short-term memory (LSTM) architecture was introduced, which significantly improved ASR performance [4]. The LSTM model is an advanced form of the gated recurrent unit, designed to facilitate information

flow over time without the derivatives vanishing or exploding. This is achieved through the use of gates that regulate the flow of information, thereby maintaining the stability of the gradient across extended sequences [15].
Interest in ASR intensified even further following the introduction of the transformer model by [17] in their groundbreaking paper, ”Attention is All You Need.” The introduction of the transformer provided the capability to handle higher-dimensional data, and by utilizing self-attention layers, it could capture long-term dependencies in input speech, thereby outperforming previous models [4]. However, as previously mentioned, collecting high quality data to train ASR models is expensive and time-consuming. Speech recognition furter progressed with the introduction of unsupervised pre-training techniques, enabling models to learn from raw audio data rather than labeled. In 2020, [5] introduced wav2vec 2.0, referred to as ”A framework for Self-Supervised Learning of Speech Representation”. The wav2vec 2.0 model encodes speech audio into a latent space using a multi-layer CNN, then masks portions of the resulting latent representation. This masked representation is subsequently fed into a transformer to build contextualized representations. These are trained using contrastive learning, wherein the correct latent representation must be distinguished from distractors. The wav2vec 2.0 outperformed state-of-the-art ASR models while using 1% as much labeled data [5]. The unsupervised pre-training of audio encoders has yielded high-quality representations of speech. However, these systems often lack a decoder of equivalent performance to effectively interpret these representations. Building on this insight, [6] introduced the Whisper model. Trained on 680,000 hours of labeled data, Whisper has achieved state-of-the-art performance across multiple languages and provides developers with the capability to fine-tune the model for domain-specific purposes [4]. In the following sections, we delve into domain-adaptation and explore the underlying architecture of whisper and Wav2Vec2.0.

2.3.2.3 Domain Adaptation

Domain adaptation refers to the situation where knowledge gained in one setting is exploited to improve generalization in another. For example, in computer vision, a model might initially learn to recognize cats and dogs, and subsequently, this learning is applied to identifying wasps. The training on cats and dogs could assist in generalizing to wasps because many visual categories share underlying low level features. While the task remains the same, the input varies [15]. [1] investigates domain-adaptation of E2E ASR

models using solely synthetic data, rendering WER improvements of between 2 and 30 points depending on the model architecture.

2.3.2.4 Wav2vec 2.0

The biological motivation behind Wav2Vec 2.0 stems from the observation that acquiring languages strictly from labeled examples does not mirror how humans naturally learn languages. Infants acquire language skills by listening to the people around them. Self-supervised learning, generally used to learn parts of the input by another part of the input has emerged as a methodology to learn general data representations from unlabeled examples and subsequently fine-tune the model using labeled data. Wav2Vec 2.0 processes raw audio data by first encoding it from raw input X to the latent space Z using a multi-layer feature encoder $f : X \rightarrow Z$. The resulting representations are then masked and fed into a transformer network $g : Z \rightarrow C$, which contextualizes the sequence and to a quantization module, $Z \rightarrow Q$, to discretize Z and form targets for the objective function. After pre-training, the model undergoes fine-tuning by on labeled data minimizing CTC loss [5].

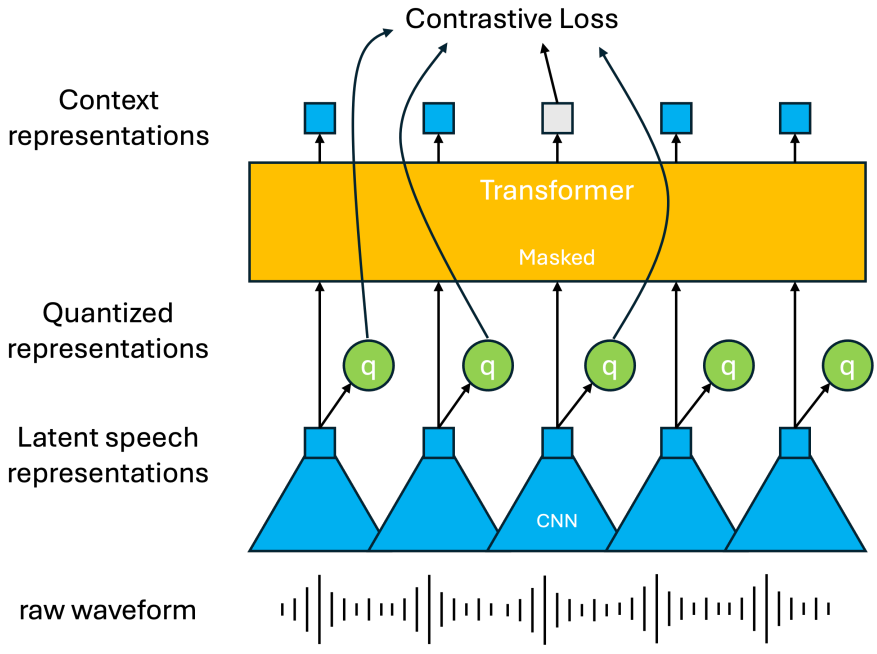

Figure 2.4: Wav2Vec 2.0

2.3.2.5 Whisper

Motivated by the discrepancy in capability between the encoder and the decoder in Wav2Vec2.0, whisper was born. [6] emphasizes that fine-tuning pre-trained audio encoders can be complex and risky, as there is a potential for models to overfit the new data. [25] demonstrates that ASR systems pre-trained using supervised learning on diverse data sets tend to generalize better across domains. Whisper is a sequence-to-sequence model designed for various speech recognition tasks, from voice activity detection to speech translation. This format is called multitask training and ultimately results in a vector containing both instructions on what task to perform and the actual text tokens being passed on to the decoder [6] .

It employs an encoder-decoder Transformer architecture and processes 30-second audio snippets. The system resamples these snippets to 16 kHz, converts them into log-mel spectrograms, and then passes them to the encoder. Both the encoder and decoder are configured with an equal number of transformer blocks and maintain identical widths. The encoder comprises two convolutional layers that process the spectrograms. Subsequently, sinusoidal position embeddings are added, and then the encoder's transformer blocks are applied. The output of this process is normalized. Following this, the decoder utilizes learned position embeddings and tied input-output token representations, along with the instructions embedded in the input vector [6].

Whisper was trained on 680.000 hours of labeled speech data. And the Whisper-Large model has 1550 million parameters. The training process did not involve and data augmentation or regularization. Instead they trained for few epochs and relied on the diversity within the audio snippets to prevent overfitting. The Whisper large-v3 model uses 128 Mel frequency bins, compared to 80, which enhances its performance [6]. .

2.3.2.6 Evaluation metrics for ASR-models

ASR models are commonly evaluated using Word Error Rate (WER). WER measures transcription accuracy by calculating the minimum edit distance between a model's output and the reference transcript. It is computed as:

$$WER = \frac{S + D + I}{N} \tag{2.5}$$

Where S is the number of substitutions, D is deletions, I is insertions, and N is the number of words in the reference transcript. A lower WER indicates better performance. Whisper achieves a Word Error Rate (WER) of 8.3 on the

Common Voice 15 dataset and 7.6 on FLEURS [6]. Another commonly used metric is character error rate (CER) which measure the ratio of characters that does not align with the reference transcript. It is caculated by:

$$CER = \frac{C}{N} \cdot 100 \tag{2.6}$$

Where C is the number of incorrect characters and N is the total number of characters in the reference text.

2.3.2.7 Evaluation metrics for binary classification

Binary classification metrics evaluate a model's performance in distinguishing between two classes, such as correctly identifying entities in ASR transcripts. Commonly used metrics include accuracy, precision, recall, and F1-score, derived from the confusion matrix

2.4 Related work

This section walks through work related to this study.

2.4.1 When whisper meets TTS

This study explored an approach to fine-tune Whisper-based models using only synthetic speech. They adapt the model's language component by freezing the encoder and only fine-tuning the decoder, allowing it to incorporate domain-specific vocabulary. Experimental evaluations across multiple languages and domains indicate that this method leads to notable reductions in word error rates, with improvements ranging from 2 to 30 percentage points, depending on the version of Whisper used [1]

2.4.2 Contrastive Audio-Language Learning for Music

Considering text and audio data are different ways of representing the same underlying information, there is a natural semantic correlation between text and audio embeddings encapsulating the same content. The authors explore a multi-modal approach using two encoders to learn the alignment between text and audio pairs, referred to as MusCALL, short for Music Contrastive Audio-Language Learning. The problem originates from the fact that music

infomration retrieval do not support search through free-form text. MusCALL is an attempt to bridge this semantic gap by learning alignment between music and text. This is achieved via multimodal contrastive learning. The authors experiment shows that their approach outperforms the baselines in retrieving audio matching a description in free-form text. The authors empaphizes that MusCALL can be transferred to perform any task involving text-based retrieval [19].

2.4.3 Enhancing Automatic Speech Recognition: Effects of Semantic Audio Filtering on Models Performance

Perez-Hohin et al. [4] propose a filtering framework aimed at enhancing the quality of synthetic data used for fine-tuning Automatic Speech Recognition (ASR) models. Inspired by the MusCALL approach [19], their framework employs contrastive learning to align synthetic audio with corresponding text representations effectively. Initially, the authors created a Portuguese synthetic dataset and fine-tuned the Whisper model, achieving suboptimal Word Error Rate (WER) performance. To address this issue, they integrated a semantic audio filtering approach utilizing a text encoder (768-dimensional embeddings) and an audio encoder (1024-dimensional embeddings). By training a projection layer to generate aligned 512-dimensional embeddings, the framework maximizes similarity between natural-sounding synthetic speech and its textual representation, while distancing unnatural audio samples from their corresponding texts.

The authors further evaluated various threshold settings for filtering synthetic audio, demonstrating that excluding low-quality synthetic audio-text pairs significantly enhances the performance of ASR models during fine-tuning. Their findings underscore the critical importance of aligning synthetic training data closely with the specific architectures and domain requirements of ASR models [4].This thesis is largely based on their work.

2.5 Summary

This chapter provided fundamental information relevant to ASR, including speech and hearing mechanisms, representation learning, and historical developments in the field. The Fourier Transform and Short-Time Fourier

Transform (STFT) were introduced as analytical methods for sound signals, explaining the typical preprocessing for speech data. The use of Log-Mel spectrograms and their alignment with human speech perception was described. Representation learning and embeddings was presented. Transformer architectures and their attention mechanisms were discussed, highlighting their capacity for modeling sequential data and Contrastive learning was described as an effective approach for extracting representations from unlabeled data.

State-of-the-art Text to Speech models were discusses and the evolution from statistical models to deep neural networks and end-to-end ASR systems, including Whisper and Wav2Vec, was described in detail. Overall, this chapter outlined essential methodologies and concepts that will serve as variables and reference points for subsequent analysis and comparison with proposed solutions.

Chapter 3

Pipeline overview

Figure 3.1 outlines the steps involved in conducting this experiment. It provides a high-level overview of all the components included, and in the following section, we will review each component individually, discussing their development and role in the experiment.

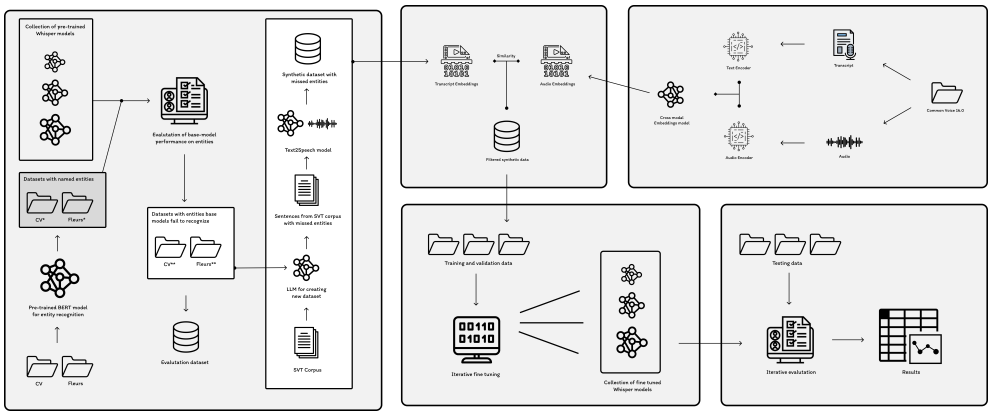

Figure 3.1: High level experiment overview

3.1 Experiment Design

This section will walk through all major tasks in the experiment. The entire pipeline is modeled below and the main checkpoints of the implementation was:

1. Creating benchmark datasets for entity recognition in ASR

2. Creating a corpus of sentences containing the missed entities
3. Creating a dataset with synthetic audio for every sentence in the corpus
4. Training the filtering model on real audio data
5. Applying filtering to remove low-quality samples from the dataset
6. Fine-tune whisper on the synthetic data
7. Evaluate WER, CER and entity recognition the benchmarks and fine-tuned whisper models
8. Compare results between the fine-tuned and benchmark models

3.1.1 Benchmarks

KBLab's Whisper models are evaluated using Word Error Rate (WER) on the Swedish subsets of the CommonVoice 16.0 and Fleurs datasets, where the smallest model scored a staggering 6.4 on Commonvoice and 7.3 on Fleurs, beating OpenAIs whisper-large-v3 in Swedish. However, WER alone does not provide insights into the models' accuracy regarding named entities. To address this gap, we preprocess the datasets using a BERT-based Named Entity Recognition (NER) model fine-tuned by KBLab. This process allows us to extract named entities along with their respective positions, indexes, and entity types from the transcripts. We then utilize this information to construct an entity-specific text corpus. Through this corpus, we evaluate Whisper's performance specifically on named entities, identifying entities the model frequently misrecognizes. These problematic entities guide the creation of targeted synthetic data, which is detailed in the following sections.

There are no publically available datasets that would allow us to evaluate ASR performance on entities, so we had to construct this ourselves. We evaluated the performance on entities extracted by an LLM. The datasets with extracted entities are 1526 and 2443 rows long, so manually evaluating the dataset is feasible. We do this to ensure the data quality as it is an integral part of this project. The entity extraction works fine, and we further refine the entities by running it through an LLM with the following prompt in Listing 3.1.

3.1.2 Corpus Collection

The foundation for our synthetic speech dataset is a text corpus provided by Språkbanken, consisting of approximately 500,000 sentences. These

Row	Text	Original Entities	Corrected Entities
9	Under 1980-talet arbetade han med TV-program som Taxi, Cheers, och The Tracy Ullman Show.	['Taxi', 'Che', 'ers', 'The Tracy', 'Ull', 'man Show']	['Taxi', 'Cheers', 'The Tracy Ullman Show']
46	Hur skulle Ios gravitation påverka mig? Om du stod på Ios yta skulle du väga mindre än du gör på jorden.	['I']	['Ios']
47	Kirby Muxloe-slottet är mer av ett förstärkt hus än ett äkta slott, vilket är typiskt för perioden.	['Kirby Muxloeslottet']	['Kirby Muxloe-slottet']
49	Den tillhör den amerikanska marinens sjunde flotta och är baserad i Sasebo, Nagasaki i Japan.	['SaseboNagasaki', 'Japan']	['Sasebo', 'Nagasaki', 'Japan']

Table 3.1: Caption

sentences are extracted from an XML file and processed with the same NER model described in the previous section, ensuring consistency in entity extraction methodology across our datasets.

Sentence	Entities	Types
Karlshamnsverket producerade 2022 dubbelt så mycket el som 2021	`Karlshamnsverket`	`ORG`
Inget gick att rädda efter den brand som startade under natten till i lördags i Restaurang Terrassen i Karlshamn	`Restaurang Terrassen Karlshamn`	`LOC`
Under lördagsförmiddagen kom ägaren Jwan Ali till platsen	`Jwan Ali`	`PRS`

Table 3.2: Example of the collected corpus with extracted entities

It is a difficult task to let an LLM generate the training data and still have great variability in sentences. Therefor we group the corpus from SVT partition of the språkbanken dataset in buckets depending on the entity it contains. For example in ”Under lördagsförmiddagen kom ägaren Jwan Ali till platsen” the entity would be labeled a person (PRS). After initial evaluation we keep track of the entities that whisper fails to accurately reconstruct. We then construct X sentences for each missed entity. To do this we simply sample 10 sentences where the entity is a person, and instruct the LLM to pick the most appropriate one and replace the entity with our entity. This prohibits weird context. Examples

Listing 3.1: Prompt to correct formatting on entities

```
prompt_template = """(
    "You are given a sentence (text), extracted entities, and their metadata.
        "
    "Your task is ONLY to fix obvious formatting errors in entities and
        metadata:\n"
    "- Merge entities mistakenly split.\n"
    "- Separate entities mistakenly joined.\n"
    "- Ensure entities EXACTLY match substrings from the original text.\n"
    "- DO NOT add additional information from the text about an entity that is
         not already present in the entity.\n"
    "- ALL text in the reformatted entity MUST be present in the original
        entity.\n"
    "- ALL entities must EXACTLY match the substrings as they appear in the
        original text, even if grammatically incorrect.\n"
    "\n"
    "Return ONLY corrected entities and metadata in plain JSON:\n"
    "{{\"entities\": [\"entity1\", \"entity2\"], \"metadata\": {{\"entity\":
        [\"entity1\", \"entity2\"], \"entity_type\": [\"type1\", \"type2
        \"]}}}}\n\n"
    "Text: {text}\n"
    "Entities: {entities}\n"
    "Metadata: {metadata}\n\n"
    "Corrected JSON:"
)"""
```

Sentence	Entities	Types
Ringhals producerade 2022 dubbelt så mycket el som 2021	`Ringhals`	`ORG`
Inget gick att rädda efter den brand som startade under natten till i lördags Lilla Ego i Stockholm	`Lilla Ego, Stockholm`	`LOC`
Under lördagsförmiddagen kom ägaren Lebowski till platsen	`Lebowski`	`PRS`

Table 3.3: Example of the collected corpus with replaced entities

3.1.3 Text to Speech model

For generating synthetic speech data, we considered two state-of-the-art models: an open-source model provided by Facebook and a commercial text-to-speech model developed by Elevenlabs, available via a paid API. Given that data quality significantly impacts model performance [4], we selected the Elevenlabs model due to its superior audio fidelity and consistency. We used three different voices, a woman, a young man and a middle aged man.

3.1.4 Filtering Framework

The filtering model, trained on audio-caption pairs from the Swedish subset of the publically available Commonvoice 16.0 dataset consists of two pre-

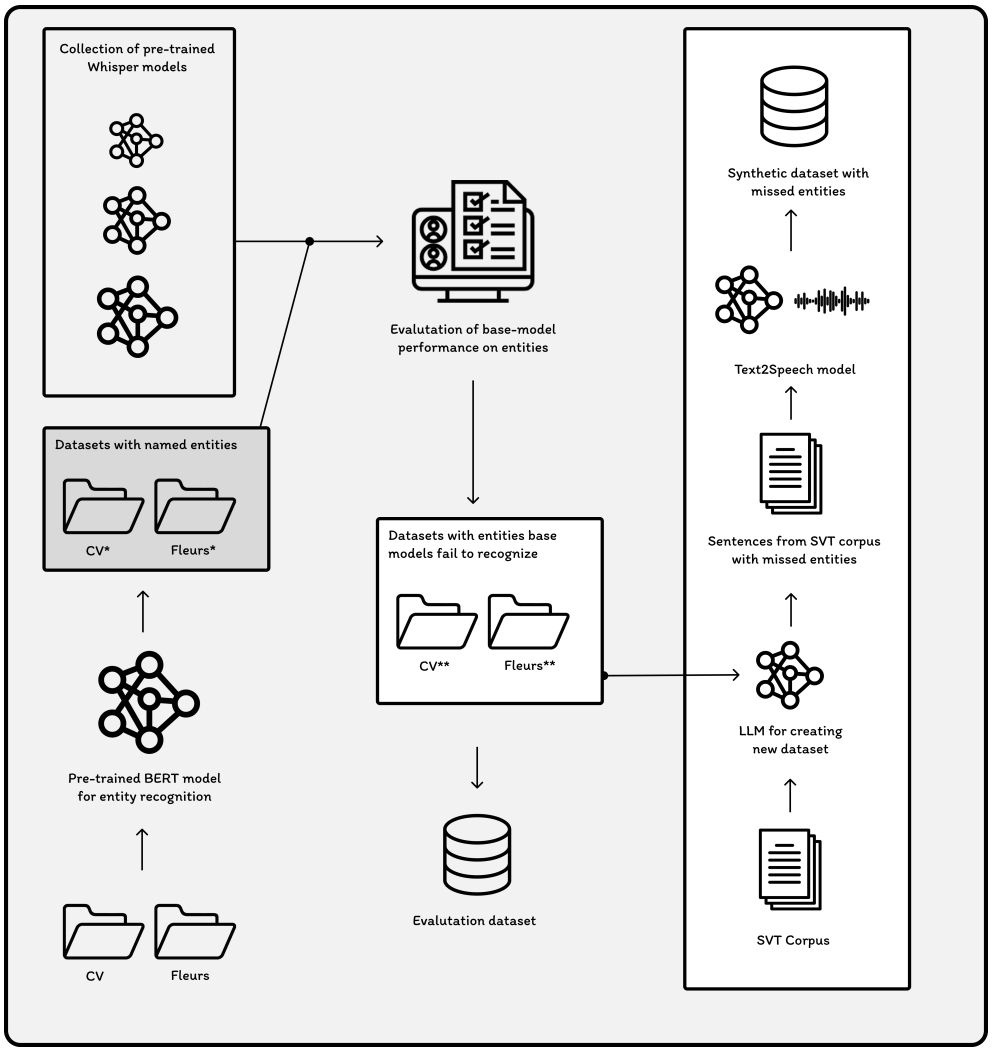

Figure 3.2: Pipeline for collecting corpus, extracting entities and creating the synthetic dataset

trained encoders, one that produces 768 dimensional text embeddings and one that produces 1024 dimensional audio embeddings. Text-embeddings are produced by extracting embeddings from a Biderectional Encoder Representations from Transformers model (BERT) and the audio embeddings are extracted from a Whisper model. The embeddings are mapped to 512 dimensional tensors in a joint space and trained using a contrastive loss function described below.

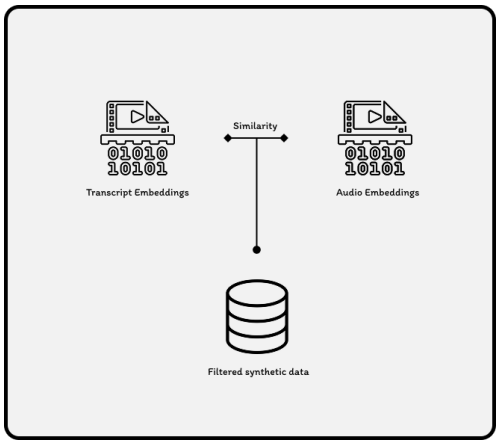

Figure 3.3: The Filtering framework

3.1.4.1 Loss function

The filtering model constructs audio and text embeddings in the same joint 512-dimensional space. In this space, where they are comparable, we can measure the cosine similarity between the audio and text representation of the same information. Ultimately, we want to keep the synthetic audio and text pairs that are similar to human speech and throw away pairs that are less similar to human speech. Therefor, the projection layers are trained to maximize the similarity between corresponding audio and text pairs using a loss function based on contrastive learning. This is accomplished by constructing positive and negative pairs through what is known as instance discrimination. In instance discrimination each data sample has its own unique class and the model learns that this class is different from all others in the batch. In this case, an audio clip A_i has a matching caption T_i, this is called a positive pair. A_i also has several non-matching captions T_j (negative pairs), i.e T_j $\forall\, j \neq i$. We denote our embedded positive pairs $Z(T)^+$ and negative $Z(T)^-$. Intuitively, this makes sense, and as defined in [19] we construct the following loss function:

$$L_{a\to t} = \frac{1}{N}\sum_{i=1}^{N} \log\left(\frac{\exp\left(Z(A_i)\cdot Z(T_i)^+/\tau\right)}{\sum_{Z\in\langle Z(T)^+, Z(T)^-\rangle}\exp\left(Z(A_i)\cdot Z(T_i)/\tau\right)}\right) \tag{3.1}$$

Where τ is the temperature parameter to scaled similarities and N is the batch size. The idea is that the projected embeddings for an audio text pair

(A_i, T_i) should lie close in the joint embedding space. Traditionally this loss function assumes that each pair is the most semantically relevant match. However this is not always the case. As described in [19], in real world datasets, within a batch some samples will share similarities with samples. Therefor we implement relevance-based weighting in the loss function taking similarities between captions (T_i, T_j) into consideration. We use a pre trained sentence-BERT encoder to extract embeddings of all captions within a batch and assign their similarities to a weight w_i

$$w_i = \exp\left(\frac{\frac{1}{N}\sum_j^N sim(t_i, t_j)}{\kappa}\right) \tag{3.2}$$

where κ is a temperature parameter and $sim(t_i, t_j)$ is the similarity score between text samples i and j. This yields the final loss function:

$$L_{a\to t} = \frac{1}{N}\sum_{i=1}^{N} w_i \log\left(\frac{\exp\left(Z(A_i)\cdot Z(T_i)^+/\tau\right)}{\sum_{Z\in\langle Z(T)^+, Z(T)^-\rangle}\exp\left(Z(A_i)\cdot Z(T_i)/\tau\right)}\right) \tag{3.3}$$

The total loss function is obtained by summing $L_{a\to t}$ to its counterpart $L_{t\to a}$. This process is called content-aware loss weighting [19].

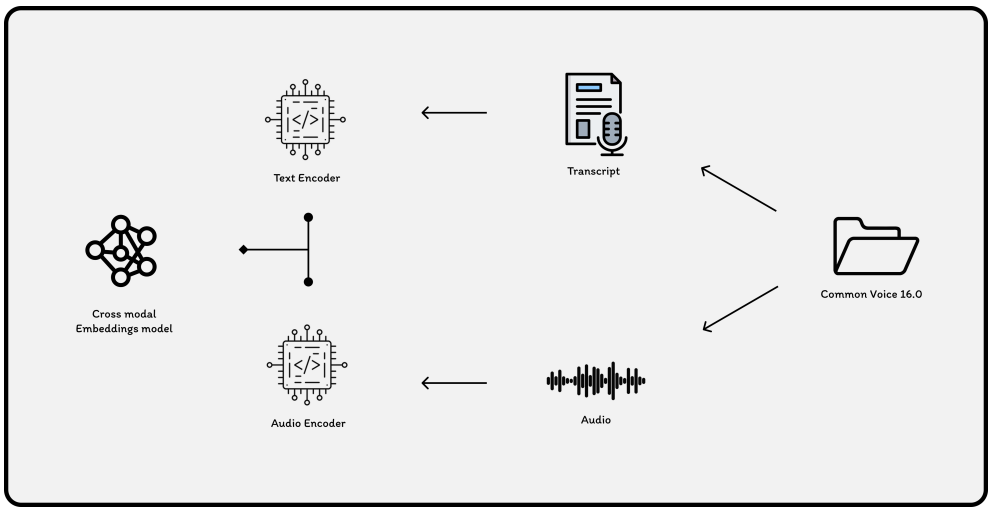

Figure 3.4: Training the filtering framework

3.1.5 Fine tuning

Using fine tuning to adapt pre-trained models to particular tasks or domains is an important part of natural language processing. Generally this is achieved by initializing a weight matrix W from a pre-trained model and continue training it on domain or task specific data, yielding $W = W_{base} + W_{tuned}$. This process is referred to as fully fine tuning a model, retraining all model parameters [9]. As pre-trained base models, we utilize three recently released foundation models from KBLab at the National Library of Sweden, unprecedented in performance on swedish ASR. These models are based on OpenAI's Whisper architecture [6] but have been trained on over 50,000 hours of transcribed Swedish audio, significantly more than the 2,119 hours used to train OpenAI's models. KBLab's best-performing model achieved a 47 percent reduction in word error rate compared to OpenAI's Whisper when transcribing Swedish. Whisper Large-v3, the largest variant, consists of 1.6 billion parameters, while Whisper Medium and Whisper Small contain 817 million and 282 million parameters, respectively.

Fine-tuning has proven to increase model performance on specific tasks, however, as pre-trained models increase in size, fine-tuning all parameters become less feasible. To tackle this we adopt a framework called Low-Rank Adaptation (LoRA). Normally, the equation for a weight update of training a machine learning is $W' = W + \Delta W$. The same equation holds for fully fine-tuning pre-trained models. The downside with this process is that we learn a new set of parameters of the same dimension as the pre-trained weights for each downsteam task, i.e the new model contains as many parameters as the old one, which is unfeasible as pre-trained models grow [9]. LoRA decomposes the weight matrix $W \in R^{d \times k}$ to new matrices $B \in R^{d \times r}$ and $A \in R^{r \times k}$, where $r < min(d, k)$ is the rank of the matrix. With this decomposition $\Delta W = BA$ and when deployed we can store $W' = W + BA$.

Among many other advantages LoRA makes training more efficient and reduce storage requirements significantly. LoRA freezes pre-trained weights and reduce the number of trainable parameters by 10.000 times for downstream tasks. Studies also show that LoRA can reduce GPU memory requirements by 3 times while maintaining on par performance [9]. Worth noting is that when using LoRA, the learned solutions are not equivalent. In [26] the authors compare spectral properties of pre-trained models weight matrices before and after fine-tuning. Their results suggest that models fine-tuned with LoRA,

despite performing on par with fully fine-tuned models on the specific fine-tuning task, may not generalize to well to other tasks or adapt as effectively when learning multiple tasks.

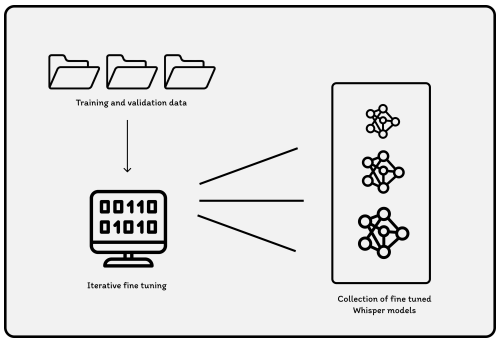

Figure 3.5: Fine-tuning the models

3.1.6 Evaluation

To account for potential mismatches in entities we track the entities we missed. The accuracy on missed entities is calculated by dividing missed entities with the total amount of entities present in the dataset. We measure this error, and the word error rate for both normalized and non-normalized sentences. We also evaluate the character error rate (CER) of the normalized and non-normalized candidate transcripts at both the sentence level and the entity level, allowing for a more granular assessment of entity-specific errors.

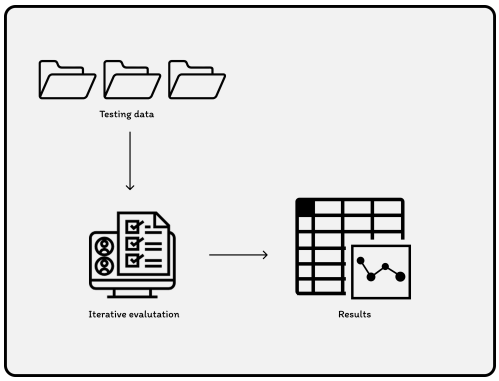

Figure 3.6: Evaluating our collecting or fine-tuned models

3.2 Validity and Reliability

This chapter argues for the reliability and validity of method and data. We anchor the arguments in specific parts of the experiments incorporated for the purpose of both validity and reliability, e.g manual verification of data.

3.2.1 Validity of method

The validity of the method depends mainly on the accuracy of measuring improvements in ASR models through synthetic data. To ensure methodological validity, the experiment is conducted with a controlled fine-tuning process across different models. The use of established evaluation metrics such as WER, CER, and entity recognition accuracy provides widely accepted measures to assess the accuracy and effectiveness of synthetic data fine-tuning. The validity is also strengthened by cross-evaluating models trained on combined real and synthetic datasets.

3.2.2 Reliability of method

The reliability of the method depends on its repeatability and consistency under the same experimental conditions. To ensure reliability, all steps, including dataset generation, entity extraction, synthetic audio creation, and model fine-tuning are thoroughly documented, with clear scripts and settings available for reproduction. Reliability can potentially be improved by employing larger and more diverse synthetic datasets and increasing the number of evaluation runs to minimize stochastic variability.

3.2.3 Data validity

The validity of the data used in this project is assessed through manual and automated checks. Manual verification was performed on all steps involving probabilistic models. Synthetic audio validity is maintained by employing a high-quality commercial TTS system, but also through manual verification. The alignment between synthetic text-audio pairs is further verified through embedding-based filtering, although in this particular experiment, no significant filtering was necessary.

3.2.4 Reliability of data

The reliability of the data depends on its consistency and robustness of the results in different evaluation scenarios. Synthetic data generation utilized multiple voices and varied sentence structures to ensure comprehensive coverage and reduce potential biases. The inclusion of real-audio datasets (CommonVoice, Fleurs) and synthetic datasets allows for robust validation of findings across different contexts. For the filtering model, the nature of the Commonvoice data in portuguese from the reference project can be different that the Swedish subset, rendering different results.

Chapter 4

Implementation details

This chapter details every step of the implementation. We cover the hardware and software used walk through the project from start to finish. The source code for this project, largely inspired by [4]:s repository on link to github, is publically available here: link to github. As my workflow required frequent commits, there is no use in trying to make any sense of the commits or commit-messages, as they are mostly nonsense and minimal changes.

4.1 Implementation pipeline

In this section we describe the implementation pipeline

4.1.1 Benchmark

Benchmark runs confirm that Whisper struggles with entity recognition. Accuracy is calculated as the proportion of entities incorrectly transcribed relative to the total number of entities in the dataset. Some transcription errors are minor and do not significantly impact interpretation, for example:

> **Ground Truth:** Han gjorde 2 mål och 2 assists i Washingtons 5-3-seger över Atlanta Thrashers.
> **Transcription:** Han gjorde två mål och två assist i Washingtons 5-3-seger över Atlanta Trashers.

However, other errors are more severe, substantially altering meaning, such as:

> **Ground Truth:** Det är det största köpet i eBays historia.
> **Transcription:** Det är det största köpet i Bergs historia.

Model	CV	CV (Norm.)	Fleurs	Fleurs (Norm.)
KBLab/whisper-small	0.718	0.754	0.593	0.640
KBLab/whisper-medium	0.737	0.772	0.624	0.669
KBLab/whisper-large-v3	0.786	0.810	0.666	0.700

Table 4.1: Entity accuracy for Common Voice and Fleurs

4.1.2 Corpus

In our implementation, detailed in Chapter 3, we encountered greater challenges than anticipated in maintaining contextual correctness. First we tried generating sentences using solely an LLM. As this did not provide contextually reasonable sentences we had to reconsider our approach. We utilized sentences from SVT Nyheter as our primary sentence pool and categorized these sentences into distinct buckets based on the entities they contained (LOC, ORG, or PER). For each entity that the Whisper model failed to recognize, we randomly sampled ten sentences from the corresponding entity-type bucket. The sentence pool contained around 250k sentences. An LLM was then tasked with selecting the most contextually appropriate sentence from these candidates, substituting the original entity with our unrecognized entity, and returning the adjusted sentence, ensuring both grammatical accuracy and contextual relevance. The contextual correctness final dataset was manually verified to account for the stochasticity of this process. The prompt is displayed in listing: 4.1

4.1.3 Creating synthetic dataset

With our newly curated corpus containing extracted entities, we can begin synthesizing audio snippets. Since Whisper is trained on 30-second audio windows, we aim to generate audio clips shorter than 30 seconds, varying in both length and speaker. We do not modify any parameters to control pitch or pace. For each entity, we generate five contextually relevant sentences, which are then synthesized. In total, we use five distinct voices, ensuring that each entity is read aloud in a unique voice for each sentence. A code snippet from the script creating the dataset is displayed in listing:4.2.

Listing 4.1: Prompt to correct formatting on entities

```
def run_correction(client, buckets, total_missed_entities):
    PROMPT_TEMPLATE = """(
        "You are given candidate sentences, each containing an entity of
            type '{entity_type}'"
        "Your task is to select the sentence that best fits incorporating the
             missed entity '{missed_entity}'"
        "Then, replace the original entity with '{missed_entity}' to
            maintain grammatical correctness"
        "If replacing the entity results in a contextually incorrect
            sentence, carefully alter the sentence to ensure contextual
            correctness, while closely following the original style,
            structure, and tone"
        "\n"
        "Avoid unrealistic or illogical combinations of organizations,
            locations, and events (e.g., political parties associating
            power transfer with unrelated companies, animal welfare
            incidents attributed to inappropriate organizations like online
             marketplaces, or historical figures managing farms). Ensure
            logical coherence, organizational plausibility, geographical
            accuracy, and factual correctness"
        "\n"
        "NEVER change the missed entity '{missed_entity}' to a different
            entity."
        "\n"
        "ONLY RETURN the modified, contextually and grammatically correct
            sentence"
        "\n"
        "Candidate sentences:"
        "{candidate_sentences}"
    )"""
```

Listing 4.2: The main method for our dataset generation

```
async def main():
    utterances = pd.read_csv("../text_generation/dataset/final_sentences.csv
        ")
    speakers = [
        "speaker1",
        "speaker2",
        "speaker3",
        "speaker4",
        "speaker5"
    ]

    await generate_audio_by_sentences_async(utterances, speakers)

if __name__ == '__main__':
    asyncio.run(main())
```

4.1.4 Applying filtering model

For the filtering model, we observe that projecting high-dimensional embeddings into a two-dimensional space effectively maps both text and audio into the same multidimensional space. However, applying the filtering does not reveal any clear distinction between synthetic and real audio. This may be because [4] used an open-source text-to-speech model, which produces lower-quality samples compared to those generated by ElevenLabs.

Additionally, we employed a BERT model as encoder, whereas [4] used an Albertina model, which could impact model performance. Differences in encoder architecture may contribute to the observed results and reduce overall performance. Given these findings, we proceed with the experiment while retaining all 15 hours of synthetically generated data.

4.1.5 Fine-tuning

Initially we fined tuned 3 models from KBLab, Whisper small, Whisper medium and Whisper large-v3 on various datasets with different configurations. First we trained them solely on 15 hours of synthetic data. Then we constructed two new datasets combining the synthetic data with real data. For the small model we accumulate gradients for 8 steps before updating the weights. We use a batch size of 4 and a learning rate of e^{-5}. For Whisper medium we used accumulated gradients for 32 steps before updating the weights, used a learning rate of e^{-6} with a batch size of one for training and 32 for evaluation. For the large model we accumulated gradients for 4 steps with a training batch size of two, an evaluation batch size of 4 and the same learning rate as for medium. For all models we train for three epochs and enable half-precision floating-point training to reduce memory usage. For training the large models, we utilze the library ”peft” from huggingface fine-tune with LoRA. Since the trainable parameters are significantly less with LoRA we have to increase the learning rate in order for the model to actually learn. For whisper large we increase it to e^{-4}. For the LoRA configuration for matrix refactorization we use a rank $r = 32$ and re-scaling factor $\alpha = 64$, in line with experiments performed by [1]. After applying LoRA for whisper large the number of trainable parameters were about 1.008%.

4.1.6 Dataset

The final dataset consisted of 6320 audio examples based on 1264 distinct entities, where each sentence has an audio representation by a distinct voice.

Figure 4.1: Distribution of entities in terms of categories

So each entity is put in 5 different sentences where each sentence is read by a distinct voice. The skewed distribution of categories is a result of the models current capabilities.

4.1.7 Hardware and software used

Various hardware and software was used to complete this project. The tasks that could be performed locally on a computer was. For the every task that required a lot of compute (i.e the tasks involving running machine learning models) AWS Sagemaker Notebook on a ml.g5.xlarge instace with 1000GBs of allocated memory was used. The filtering model was trained on the Swedish Commonvoice 16.0 dataset retrieved via the huggingface api and processed locally. The synthetic dataset was generated using a TTS available by API from elevenlabs. For the tasks that required GPT models we used GPT-4o-mini from OpenAI via their asynchronous API. For version control we used github.

Chapter 5

Results and Analysis

In this chapter, we present and analyze the results of our experiments. We primarily focus on training loss, validation loss, and training runtime across multiple models, along with entity accuracy and word error rate (WER) on various datasets for the fine-tuned models. Our best medium-sized model improved entity accuracy from 62.4% to 63.3%, while reducing WER from 0.163 to 0.161. For large models, the best-performing model achieved a 1.2% increase in entity accuracy, improving from 66.6% to 67.8%, and reduced WER by 0.01, from 0.134 to 0.124, on the Fleurs dataset with extracted entities. Overall, the medium-sized models exhibit the most significant improvement, whereas the small and large models do not perform as well.

5.1 Fine-tuning

In this section, we present and analyze the data obtained during the training and evaluation of multiple models in various combinations of datasets. To fine-tune the large models we utilized parameter efficient fine tuning by Low rank adaptation of large language models (LoRA) to enable fine-tuning on one a single GPU.

5.1.1 Training

Initially, we fine-tuned three models from KBLab: Whisper Small, Whisper Medium, and Whisper Large-v3 on 15 hours of synthetic data. The first training run (Whisper Small) took approximately 30 minutes and exhibited a consistently decreasing training and validation loss, indicating that the models were fitting well to the training data without overfitting. The second training

run, using Whisper Medium as base model took about 2.5 hours, showing similar decrease in loss. As fine-tuning the large model was done with LoRA, only 1.008 percent of the parameters were tuned, yielding a shorter training time. This training took about 1.17 hours. Both training and validation loss showed steadily decreasing loss during training. Given these loss plots the fine-tuning on synthetic data with our configuration seems successful. To decrease training time we logged the validation loss less frequently for the large model.

Figure 5.1: Training and validation loss curves for the three models. From left to right, small, medium, large.

Fine-tuning KBlabs Whisper small on the combined dataset of the Synthetic data and Fleurs took 1 hour and 3 minutes. Fine-tuning of the medium sized model took around 4 hours and fine-tuning the large model, despite utilizing LoRA for parameter efficient fine-tuning took over 16 hours. Across all models we see that training and validation loss decrease steadily throughout the entire training.

The fine-tuning on the combined dataset of Synthetic data and Common Voice the small model took 4 hours, and for the medium model it took 12.5 hours. For both models we see a steadily decreasing validation loss indicating that the model generalizes well. The training loss is also decreasing throughout training. The gap between training and validation loss is small for both models. However, the gap narrows as training continues indicating slight overfitting. We see slight oscillations in the training loss, possibly due to variations in

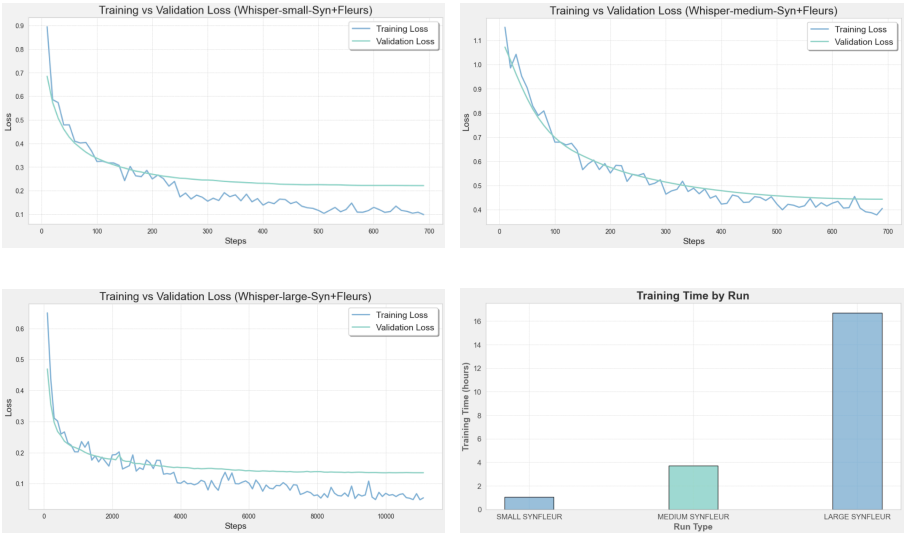

Figure 5.2: Training and validation loss curves for the three models trained on both the synthetic dataset and fleurs. From left to right, small, medium, large.

data distribution across batches, given that synthetic and real audio might have slightly different characteristics. The large model took over 14 hours and also showed stable training. To decrease training time we logged the validation loss less frequently for the large model here aswell. We also observe that training loss is more stable in fine-tuning runs where we combine synthetic and real audio data.

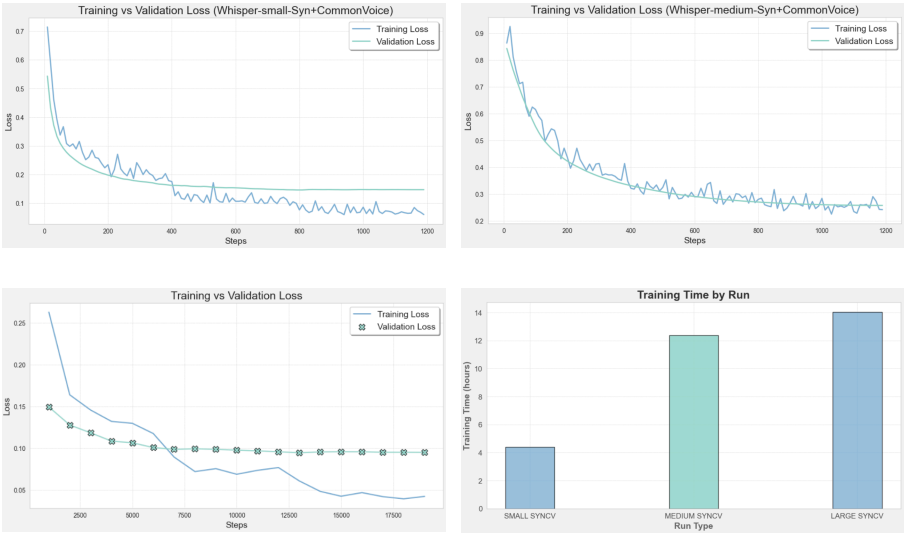

Figure 5.3: Training and validation loss curves for the three models trained on both the synthetic dataset and commonvoice.

5.1.2 Evaluation

We evaluate three distinct fine-tuning strategies: one trained exclusively on our synthetic dataset, another trained on the combination of our synthetic dataset and Common Voice, and the third trained on our synthetic dataset combined with Fleurs. One thing to keep in mind with this approach is that while the model might not overfit to the actual training data, it might overfit to synthetic data, increasing word error rate on real data. Worth noting is that our evaluation is not aligned with the scores produced on huggingface by KBLab, where they show lower word error rate. This is because we only evaluated the models on a the subsets of Fleurs and Commonvoice that contained entities the models struggeled to recognize, naturally yielding a higher word error rate. The overall trend is an increasing accuracy on entities and varying changes in WER depending on model. Normalized refers to where all sentences are converted to lower-case, while non-normalized sentences are the ground truth transcript.

For the datasets comprising a combination of real and synthetic data (Synthetic + CommonVoice and Synthetic + Fleurs), the results deviated from initial expectations. Notably, a significant decline in performance was observed for the model trained on the Synthetic + Fleurs dataset. Specifically, for non-normalized entities, the performance metric dropped from 0.718 to

0.536, while the word error rate (WER) increased from 0.149 to 0.213.

Base Models								
Model	**Common Voice**				**Fleurs**			
	Non-Normalized		**Normalized**		**Non-Normalized**		**Normalized**	
	Accuracy	**WER**	**Accuracy**	**WER**	**Accuracy**	**WER**	**Accuracy**	**WER**
small	0.718	0.149	0.753	0.096	0.593	0.172	0.640	0.099
medium	0.737	0.141	0.772	0.083	0.624	0.163	0.669	0.089
large-v3	0.786	0.104	0.810	0.062	0.666	0.134	0.700	0.071
Fine-Tuned Models								
Model	**Common Voice**				**Fleurs**			
	Non-Normalized		**Normalized**		**Non-Normalized**		**Normalized**	
	Accuracy	**WER**	**Accuracy**	**WER**	**Accuracy**	**WER**	**Accuracy**	**WER**
small	0.752	0.174	0.779	0.131	0.640	0.218	0.671	0.157
medium	**0.752**	**0.125**	**0.782**	**0.080**	**0.633**	**0.161**	0.678	0.093
large-v3	0.781	0.153	0.799	0.115	**0.678**	**0.124**	0.703	0.153

Table 5.1: Entity accuracy and Word Error Rate (WER) for Common Voice and Fleurs, with separate groupings for un-normalized and normalized values. The models are fine-tuned solely on synthetic data. Models that show improvement in comparison to the base-models are marked with **bold**. Datsasets contain sentences in commonvoice and fleus that the transcription model struggle to recognize.

Model trained on synthetic + fleurs, evaluated on CV				
	Non-Norm Acc	**Non-Norm WER**	**Norm Acc**	**Norm WER**
small	0.536	0.213	0.736	0.139
medium	0.719	0.137	**0.776**	0.088
large-v3	0.386	0.223	0.751	0.121
Model trained on synthetic + commonvoice, evaluated on Fleurs				
	Non-Norm Acc	**Non-Norm WER**	**Norm Acc**	**Norm WER**
small	**0.645**	0.219	**0.676**	0.161
medium	**0.637**	0.180	**0.677**	0.115
large-v3	**0.687**	0.204	**0.706**	0.146

Table 5.2: Entity accuracy and Word Error Rate for models trained on datasets consisting of synthetic and real data. Comparing non-normalized and normalized results. Models fine-tuned on training data containing data from the fleurs dataset was evaluated on the commonvoice dataset and vice versa. **Bold** indicates improvement in comparison to the base-models

In contrast, the model trained on the Synthetic + CommonVoice dataset exhibited performance comparable to that of the model trained exclusively on synthetic data. Apart from the small model trained on synthetic + fleurs, we observe a general increase in entity accuracy along with a higher word error rate for the small models. We also evaluate the CER for every candidate sentence:

Base model				
	Common Voice		**Fleurs**	
	Non-Norm CER	**Norm CER**	**Non-Norm CER**	**Norm CER**
small	0.044	0.032	0.053	0.035
medium	0.040	0.028	0.051	0.032
large-v3	0.027	0.019	0.040	0.025
Models fine-tuned solely on synthetic data				
	Common Voice		**Fleurs**	
	Non-Norm CER	**Norm CER**	**Non-Norm CER**	**Norm CER**
small	0.052	0.043	0.098	0.084
medium	**0.035**	**0.025**	0.052	0.035
large-v3	0.044	0.037	0.097	0.083
Models fine-tuned on synthetic + fleurs, evaluated on CV				
	Common Voice		**Fleurs**	
	Non-Norm CER	**Norm CER**	**Non-Norm CER**	**Norm CER**
small	0.063	0.046	-	-
medium	**0.038**	**0.027**	-	-
large-v3	0.061	0.039	-	-
Models fine-tuned on synthetic + commonvoice, evaluated on Fleurs				
	Common Voice		**Fleurs**	
	Non-Norm CER	**Norm CER**	**Non-Norm CER**	**Norm CER**
small	-	-	0.100	0.086
medium	-	-	0.061	0.045
large-v3	-	-	0.092	0.079

Table 5.3: Normalized and non-normalized character error rate (CER) for models trained on datasets consisting of varuous combinations synthetic and real data, evaluated on the subsets of CommonVoice and Fleurs.

For an even more granular approach to evaluation we calculate the CER within each entity.

Base model				
	Common Voice		**Fleurs**	
	Non-Norm CER	**Norm CER**	**Non-Norm CER**	**Norm CER**
small	0.372	0.334	0.302	0.290
medium	0.346	0.316	0.285	0.260
large-v3	0.303	0.277	0.261	0.239
Models fine-tuned solely on synthetic data				
	Common Voice		**Fleurs**	
	Non-Norm CER	**Norm CER**	**Non-Norm CER**	**Norm CER**
small	**0.360**	0.338	0.311	0.303
medium	**0.327**	**0.301**	**0.280**	**0.258**
large-v3	0.309	0.299	0.304	0.293
Models fine-tuned on synthetic + fleurs, evaluated on CV				
	Common Voice		**Fleurs**	
	Non-Norm CER	**Norm CER**	**Non-Norm CER**	**Norm CER**
small	**0.307**	0.343	-	-
medium	**0.323**	**0.311**	-	-
large-v3	**0.277**	0.308	-	-
Models fine-tuned on synthetic + commonvoice, evaluated on Fleurs				
	Common Voice		**Fleurs**	
	Non-Norm CER	**Norm CER**	**Non-Norm CER**	**Norm CER**
small	-	-	0.318	0.309
medium	-	-	**0.279**	0.266
large-v3	-	-	0.309	0.291

Table 5.4: Normalized and non-normalized character error rate (CER) within the extracted entities from the sentences for models trained on datasets consisting of variuous combinations synthetic and real data, evaluated on the subsets of CommonVoice and Fleurs. The fine-tuned models that showed improvement in comparison to the base-models are marked in **bold**

To get a more accurate measurment of the models performance in terms of WER and CER we do an additional run on the full, non-altered commonvoice and fleurs dataset, allowing us to compare WER with KBLabs benchmarks.

Model Size	CV (Base)	CV (Tuned)	Fleurs (Base)	Fleurs (Tuned)
Small	0.087	0.148	0.091	0.126
Medium	0.054	**0.052**	0.066	0.069
Large	0.041	0.097	0.054	0.121

Table 5.5: WER comparison of base and tuned models on Common Voice (CV) and Fleurs. Lower scores indicate better performance. Bold indicates best result within a row.

We see that in comparison to the base-models, the fine-tuned models perform worse in terms of WER, apart from the medium-sized model on the commonvoice dataset. The fine-tuned models may be ovefitting to the properties of synthetic data.

5.2 Reliability Analysis

Reliability refers to whether we measure in a consistent and dependable manner. We see consistent performance of similar models across evaluations of the fine-tuned models. By conducting experiments with several different models and comparing results across multiple datasets reliability was systematically assessed. Reliability could be enhanced by employing the same fine-tuning strategy (LoRA) for all model-sizes and by doing additional repetitions of fine-tuning for a stronger statistical robustness of results. However, we deal with limited resources and conduct many runs within the experiment, limiting the possibility for such measures.

5.3 Validity Analysis

Validity refers to whether we measure what is relevant in the given context. In terms of validity the chose mentrics WER, CER and accuracy are relevant and widely accepted in terms of evaluating ASR capability. The real-audio datasets chosen are also widely recognized for ASR evaluation, as they are used by both OpenAI in [6] and for evaluation of KBLabs models. Furthermore addressing entities that were previously difficult for the models to recognize enhance relevance of the experiments.

5.4 Discussion

The results demonstrate that fine-tuning Whisper models with high-quality synthetic data effectively improves model performance on transcribing out-of-vocabulary words, particularly entities. However, in terms of general transcription accuracy WER is significantly increased for small and large models, while remaining almost uneffected for the medium models. While the improvements in accuracy are clear, they are not groundbreaking, likely due to the already extensive fine-tuning performed on KBLab's Whisper models using large volumes of Swedish speech data. Measuring CER and specifically evaluating CER within entities further confirms the validity of these results by illustrating the targeted impact on transcription accuracy.

An interesting direction for future research would be to apply similar fine-tuning experiments using OpenAI's Whisper models as base models. Investigating performance differences in such scenarios might yield greater improvements due to the less extensive prior fine-tuning compared to KBLab's models. Integrating synthetic data not only addresses accuracy issues related to rare and out-of-vocabulary words but also mitigates privacy concerns, as synthetic speech does not compromise individual privacy.

Chapter 6

Conclusions and Future work

This chapter summarizes the key findings of this project. It also highlights that modest results might be due unprecedentedly good base-models. We also suggest future work, including testing different model architectures, exploring semantic accuracy evaluation methods, investigating alternative fine-tuning strategies, and enhancing synthetic data validation methods for cost-effectiveness.

6.1 Conclusions

Our results indicate that fine-tuning Whisper models using high-quality synthetic data can enhance model performance on transcribing out-of-vocabulary words, specifically entities, without significantly compromising the overall WER or CER. Although the improvements observed are incremental rather than groundbreaking, this outcome underscores the viability of using synthetic speech data to address challenges associated with rare and domain-specific vocabulary.

The relatively modest improvements could partly be due to KBLab's Whisper models already being extensively fine-tuned on large volumes of Swedish data. Future work should involve investigating if greater performance improvements are achievable when starting from less extensively fine-tuned, more multilingual base models. This proof-of-concept suggests a clear potential to scale the approach by building larger datasets and perform comprehensive evaluations across broader scenarios.

6.2 Future work

Due to the breadth of the problem, only some of the initial goals have been met. In this section, we focus on some remaining issues that should be addressed in future work. Regarding the CLIP-model and Filtering Framework, future work could explore several different text and audio encoders, preferably varying in architecture. For instance, employing Albertina instead of BERT. Although the filtering model did not function optimally in this experiment, alternative encoders might yield different outcomes and should be tested thoroughly.

We employed LoRA for fine-tuning large models due to practical constraints. Given that small and medium models were fully fine-tuned, an interesting future approach could involve either fully tuning the large model or applying LoRA to smaller models, allowing for direct comparisons both in terms of model weights and overall results on synthetic data. Furthermore trying out different base-models to investigate generalizability to other languagues would be interesting, as KBLabs models aren't applicable to other languages than swedish.

Another aspect worth investigating is the semantic meaning of the entities. While CER provides a measure of similarity between the candidate and the actual entity, future research could develop metrics or methods that more precisely capture semantic accuracy, contributing to a deeper understanding of model performance.

Also, it's worth considering that the fine-tuning process might overfit to the specific voices included in the synthetic speech dataset. An evaluation of a synthetic version of the eval dataset would provide insight into this. If that would be the case, it would indicate that you can specialize ASR models to individual voices, which is also an interesting area to explore.

Lastly, future work could further explore automated approaches to synthetic data validation, enhancing the scalability and robustness of these techniques across diverse languages and domains.

6.2.1 Cost analysis

The current text-to-speech generation approach functions effectively but remains expensive. A fully operational filtering framework could enable the use of open-source, cheaper, and lower-quality text-to-speech solutions, significantly reducing costs while maintaining acceptable data quality.

6.2.2 Next obvious things to be done

In particular we wish to highlight the unresolved challenge of effectively validating synthetic data quality using automated methods. Solving this problem is essential to fully exploit the potential of synthetic data in ASR model training, ensuring robustness, efficiency, and cost-effectiveness of future research and applications.

References

[1] J. C. Vásquez-Correa, H. Arzelus, J. M. Martin-Doñas, J. Arellano, A. Gonzalez-Docasal, and A. Álvarez, "When whisper meets tts: Domain adaptation using only synthetic speech data," in *International Conference on Text, Speech, and Dialogue*. Springer, 2023, pp. 226–238. [Pages 1, 2, 3, 4, 5, 13, 15, 18, and 36.]

[2] E. Pusateri, A. Walia, A. Kashi, B. Bandyopadhyay, N. Hyder, S. Mahinder, R. Anantha, D. Liu, and S. Gondala, "Retrieval augmented correction of named entity speech recognition errors," *arXiv preprint arXiv:2409.06062*, 2024. [Page 1.]

[3] T.-Y. Hu, M. Armandpour, A. Shrivastava, J.-H. R. Chang, H. Koppula, and O. Tuzel, "Utilizing imperfect synthetic data to improve speech recognition," in *ICASSP*, 2022. [Online]. Available: https://arxiv.org/abs/2110.11479 [Pages 1, 3, and 13.]

[4] Y. Perezhohin, T. Santos, V. Costa, F. Peres, and M. Castelli, "Enhancing automatic speech recognition: Effects of semantic audio filtering on models performance," *IEEE Access*, 2024. [Pages 1, 3, 4, 5, 7, 10, 11, 12, 13, 14, 15, 19, 24, 33, and 36.]

[5] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," *Advances in neural information processing systems*, vol. 33, pp. 12 449–12 460, 2020. [Pages 1, 11, 15, and 16.]

[6] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, "Robust speech recognition via large-scale weak supervision," in *International conference on machine learning*. PMLR, 2023, pp. 28 492–28 518. [Pages 2, 15, 17, 18, 28, and 47.]

[7] U. B. Maulik, P. Mitra, and S. Sarkar, "Enhancing domain-specific asr performance using finetuning and zero-shot prompting: A study in the medical domain," 2025. [Page 2.]

[8] C. Van Gysel, "Modeling spoken information queries for virtual assistants: Open problems, challenges and opportunities," in *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2023, pp. 3335–3338. [Page 2.]

[9] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen *et al.*, "Lora: Low-rank adaptation of large language models." *ICLR*, vol. 1, no. 2, p. 3, 2022. [Pages 4 and 28.]

[10] S. Dusan and L. R. Rabiner, "Can automatic speech recognition learn more from human speech perception?" *Trends in Speech Technology*, pp. 21–36, 2005. [Page 7.]

[11] M. Honda, "Human speech production mechanisms," *NTT Technical Review*, vol. 1, no. 2, pp. 24–29, 2003. [Page 7.]

[12] A. V. Oppenheim, *Discrete-time signal processing*. Pearson Education India, 1999. [Page 7.]

[13] X. Huang, A. Acero, H.-W. Hon, and R. Reddy, *Spoken Language Processing: A guide to theory, algorithm, and system development*. Prentice hall PTR, 2001. [Pages 8, 9, and 10.]

[14] J. O. Smith, *Mathematics of the discrete Fourier transform (DFT): with audio applications*. Julius Smith, 2007. [Page 8.]

[15] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org. [Pages 10, 14, and 15.]

[16] X. Favory, K. Drossos, T. Virtanen, and X. Serra, "Coala: Co-aligned autoencoders for learning semantically enriched audio representations," *arXiv preprint arXiv:2006.08386*, 2020. [Page 11.]

[17] A. Vaswani, "Attention is all you need," *Advances in Neural Information Processing Systems*, 2017. [Pages xi, 11, 12, and 15.]

[18] J. Lee and K. Toutanova, "Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, vol. 3, no. 8, 2018. [Page 11.]

[19] I. Manco, E. Benetos, E. Quinton, and G. Fazekas, "Contrastive audio-language learning for music," *arXiv preprint arXiv:2208.12208*, 2022. [Pages 12, 19, 26, and 27.]

[20] 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. [Page 13.]

[21] 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. [Page 13.]

[22] K. H. Davis, R. Biddulph, and S. Balashek, "Automatic recognition of spoken digits," *The Journal of the Acoustical Society of America*, vol. 24, no. 6, pp. 637–642, 1952. [Page 13.]

[23] B.-H. Juang and L. R. Rabiner, "Automatic speech recognition–a brief history of the technology development," *Georgia Institute of Technology. Atlanta Rutgers University and the University of California. Santa Barbara*, vol. 1, no. 67, p. 1, 2005. [Pages 13 and 14.]

[24] J. Li *et al.*, "Recent advances in end-to-end automatic speech recognition," *APSIPA Transactions on Signal and Information Processing*, vol. 11, no. 1, 2022. [Page 14.]

[25] W. Chan, D. Park, C. Lee, Y. Zhang, Q. Le, and M. Norouzi, "Speechstew: Simply mix all available speech recognition data to train one large neural network," *arXiv preprint arXiv:2104.02133*, 2021. [Page 17.]

[26] R. Shuttleworth, J. Andreas, A. Torralba, and P. Sharma, "Lora vs full fine-tuning: An illusion of equivalence," *arXiv preprint arXiv:2410.21228*, 2024. [Page 28.]

TRITA-EECS-EX-2025:585
Stockholm, Sverige 2025

www.kth.se