CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Automatic Emergency Detection in Naval VHF Transmissions

Investigating the feasibility of self-supervised speech-to-text models for complex domains with large amounts of unlabelled data

Master's thesis in Mathematical Sciences

Jonathan Gildevall & Niclas Johansson

Department of Mathematical Sciences
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2021

Master's thesis 2021

Automatic Emergency Detection in Naval VHF Transmissions

Investigating the feasibility of self-supervised speech-to-text models for complex domains with large amounts of unlabelled data

Jonathan Gildevall & Niclas Johansson

UNIVERSITY OF GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Mathematical Sciences
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2021

Automatic Emergency Detection in Naval VHF Transmissions
Investigating the feasibility of self-supervised speech-to-text models for complex domains with large amounts of unlabelled data
Jonathan Gildevall & Niclas Johansson

Supervisor: Marina Axelson-Fisk, Department of Mathematical Sciences
Advisor: Kristoffer Röshammar, Tenfifty AB
Examiner: Johan Jonasson, Department of Mathematical Sciences

Master's Thesis 2021
Department of Mathematical Sciences
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Picture of a sinking ship image: Freepik.com. This cover has been designed using resources from Freepik.com

Typeset in LaTeX
Gothenburg, Sweden 2021

Automatic Emergency Detection in Naval VHF Transmissions
Investigating the feasibility of self-supervised speech-to-text models for complex domains with large amounts of unlabelled data
Jonathan Gildevall & Niclas Johansson
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

As the proficiency of Speech-To-Text (STT) services increases, so does the possible applications. This thesis explores the use of such services in a very special domain, naval VHF transmissions. It explores STT service performance and details the development of a domain-specific Speech-To-Text model based on the self-supervised wav2vec 2.0 architecture. This enabled the recognition of emergency messages using keyword detection and also created a foundation for more advanced intent analysis in the future. The developed model outperforms Google on the naval domain and achieves good classification results using keyword detection, managing to discern most messages containing one or more keywords. This performance meant that the model could be used as an aid for actual emergency message detection by Sjöfartsverket. The research also shows that many of the pre-trained models do not have adequate performance on the intended domain, but it was noted that using semi-supervised methods such pre-trained models can be tuned to reach acceptable performance levels. This can be done with smaller sets of domain-specific data to achieve good results on the specific domains without the need for a completely new model for each domain.

Keywords: Automatic Speech Recognition, Speech-To-Text, Intent Analysis, Self-supervised, wav2vec 2.0, Naval Environment, Emergency Messages

Acknowledgements

First of all, we would like to thank our supervisor Marina Axelson-Fisk for helping us throughout the spring with our various questions. We also want to thank our examiner Johan Jonasson for his contribution and attention to detail. Then we would like to thank everyone at Tenfifty who made us feel like we were part of the company and provided invaluable guidance. AI-Sweden supported us with hardware that allowed us to complete the thesis, so a big thanks to them as well. The final thank you has to go to the researchers at Facebook AI because without the progress they made and their strive for availability, this thesis would have looked a lot different.

Jonathan Gildevall & Niclas Johansson, Gothenburg, May 2021

Contents

Glossary

API Application Programming Interface 2, 3, 32, 33, 38, 42, 44, 45, 57–60
ASR Automatic Speech Recognition 2, 4, 5, 69
CER Character Error Rate 4, 9, 38, 47, 49–51, 60
CNN Convolutional Neural Network 11, 18
CTC Connectionist Temporal Classification 20, 36
FFT Fast Fourier Transform 15, 16
GAN Generative Adversarial Network 2, 3, 11, 16, 31, 32, 42, 58
GELU Gaussian Error Linear Unit 18
GRU Gated Recurrent Unit 13
JRCC Joint Rescue Co-ordination Centre .. 1, 7, 25–27, 29, 31, 32, 34–37, 39, 45, 46, 50, 52, 53, 57–61, 63, 64, 68
LCSS Longest Common Substring Similarity 39, 56
LM Language Model 20, 21, 29, 37, 38, 47, 49, 50, 58–60, 67, 69
LSTM Long Short-Term Memory 13
MLP MultiLayer Perceptron 11, 12
MOS Mean Opinion Score 3, 9, 28, 32
MRA Match Rating Approach 23, 39, 56, 62
NLP Natural Language Processing 2, 22, 39
PReLU Parametric Rectified Linear Unit 17
ReLU Rectified Linear Unit 17, 18
RNN Recurrent Neural Network 2, 11, 13, 14, 31, 32, 69
SE Speech Enhancement 2, 3, 15, 31–33, 41, 45, 57–59, 68
SEGAN Speech Enhancement Generative Adversarial Network 16, 29, 32, 41, 42, 58, 68
SNR Signal to Noise Ratio 9, 33, 45
SSNR Segmental Signal to Noise Ratio 3, 9, 32, 41, 42
STT Speech-To-Text 2–4, 8, 17, 25, 30–34, 38, 39, 41, 42, 56–63, 67, 68
VHF Very High Frequency 1–3, 7, 8, 25, 31, 45, 67, 68
WER Word Error Rate 4, 8, 9, 20, 33, 37, 38, 43–45, 47, 49, 50, 52, 58–61

List of Figures

List of Tables

1

Introduction

The Joint Rescue Co-ordination Centre (JRCC), otherwise known as "Sjö- och flygräddningscentralen", is responsible for handling emergency responses in Swedish waters and airspace [1]. In 2020 the JRCC handled 1893 emergencies involving commercial and recreational vessels as well as other coastal incidents, that is an increase over the 1799 emergencies in 2019 [2]. The JRCC also coordinates the maritime assistance service, which includes the handling of environmental disasters and non-emergency assistance for vessels. These emergency messages are mainly received by Very High Frequency (VHF) radio and phone [3].

The VHF system is set up in such a way that it covers the entire Swedish coastline, as well as the three largest lakes [4]. It consists of 56 masts and receivers which send their incoming transmissions to the JRCC headquarters in Gothenburg and there every tower is treated as a separate source. The VHF transmissions are usually monitored by a single rescue leader at all times and due to the current configuration, the operator listens to every message as they are received [5]. This presents challenges in the form of simultaneous transmissions and difficulties in receiving new messages while coordinating an active emergency. The effects of this are the possibility of missed emergencies which in turn could lead to the loss of life. The work of the rescue leaders is complicated further by the fact that the VHF channel for emergencies, channel 16, is used for a lot more than asking for help. This means that they actively have to discern which messages are directed at them and which to simply ignore.

The thesis is part of a larger project, where the ambition is to create a tool for aiding the operators with identifying emergency transmissions and the results of this research will play a key part. A successful outcome will pave the way for a more complete system that will, according to a recently published article by Sjöfartverket, bring enormous benefit to the rescue leaders and help them save lives [6]. The project is the first application of artificial intelligence on VHF transmissions ever attempted and will lead the way forward.

1.1 Background

The use of voice-controlled applications has increased massively during the last years and so has the interest in Automatic Speech Recognition (ASR) systems. ASR is the technology of turning speech into text, which is why it is also commonly known as Speech-To-Text (STT), and it is used in meeting transcription software, virtual assistants, and video-captioning to name a few examples of applications [7].

ASR systems tasked with converting spoken language to textual representations are traditionally built using large amounts of transcribed, studio-grade, recordings, making the development of such systems very data-intense [8]. These requirements have created a situation where the higher-performing systems are available as paid services, the Google Cloud Speech-To-Text Application Programming Interface (API)[1], or as pre-trained versions, where Mozilla DeepSpeech[2] is one of the best performing, to enable the use of complex and generalising models. Without this availability, the creation of ASR systems is simply out of reach for many organisations, as these requirements for large amounts of high-quality data is something that limits the creation of systems for domains with limited access to transcribed recordings [9]. Fortunately, this has led to the rise of self-supervised STT models, building on the success of such techniques in areas like Natural Language Processing (NLP) [10, 11, 12] and image recognition [13, 14, 15, 16, 17]. At the time of writing one of the most prominent self-supervised systems is the evolution of wav2vec by Facebook AI, wav2vec 2.0 [18, 9]. It has achieved state-of-the-art results on the LibriSpeech dataset[3], detailed in Section 3.3, and has been the centre of several ground-breaking research papers on the topic [19, 9, 20, 21]. Self-supervised means that the model is able to learn without needing a large, annotated dataset. wav2vec 2.0 is implemented in such a way that it first attempts to learn speech representations from the recorded audio. That is a dataset without the need for transcriptions. This is then followed by a simpler step that requires some transcriptions in order to map the representations to letters.

As the use of ASR systems increases so does the interest in Speech Enhancement (SE) and noise reduction technologies. SE is the process of removing unwanted sound such as noise created by the environment in order to make the speech easier to understand for the STT models. Different kinds of audio processing have been used for a long time, ranging from purely algorithmic solutions to neural network-based approaches. A recently presented technique using Generative Adversarial Networks (GANs) is particularly interesting [22], but there are also some based on the more conventional neural network architectures known as Recurrent Neural Networks (RNNs). An example of this is the Conv-TasNet [23] model. These have all very different approaches to audio processing, but the main difference lies in the data requirements, where the key point of interest in the GAN approach is the possible use of the domain-specific VHF data.

[1] `https://cloud.google.com/speech-to-text`
[2] `https://github.com/mozilla/DeepSpeech`
[3] `https://www.openslr.org/12`

1.2 Problem

As presented in the introduction, the biggest problem for the rescue leaders is that they might miss a message and fail to respond to the emergency. The goal is thus to create a system that can understand which messages contain emergencies and the ability to notify the operators about them.

The thesis contains two attempts at solving this fundamental problem, the first approach was to build a model for Speech Enhancement (SE) that would allow the Google STT API to perform well enough for some basic intent analysis on the transcribed messages. The reason for this was that there had already been some research done on the topic of STTs in this specific domain, where Google's API was used to transcribe the messages [5]. This research concluded that the Google algorithm did not have the required performance to be usable by the operator but that it performed well on higher-quality messages. It could not be validated until the work had started due to data availability issues; this approach is detailed in the first part of the method.

The second approach was based on what was learnt from the initial approach and switched focus entirely. The use of SE was postponed and all the effort was put into the development of a new STT model specifically for the domain. A decision was made to use a self-supervised model since it suited the data availability and to base it on available solutions to enable quicker progress. This would then be used to create transcriptions for the same basic intent analysis.

Both approaches are researching the feasibility of using neural networks for understanding the intent of transmissions broadcasted on VHF radio in the naval domain. The intent analysis on the transcribed recordings had to be limited to robust keyword detection using word similarity measures due to the time spent on the first approach. Suggestions for more thorough intent analysis techniques are given in the section about future work, Section 7.1.

1.2.1 Objectives

The goal of the thesis described in Section 1.2 requires multiple components that will perform their respective tasks. In order to build these components so that they fulfil their tasks and requirements, a few objectives are needed to evaluate them so that the final system can work well as a whole. The Speech Enhancement section refers to the initial approach only and the use of the Google STT API is left out entirely.

Speech Enhancement: Evaluate the impact of different kinds of SE methods for STT performance. Also includes the creation of a GAN for this purpose based on the research of Pascual *et al.* [22].
Metrics: Segmental Signal to Noise Ratio (SSNR), Mean Opinion Score (MOS)

Audio transcription: Create domain-specific Speech-To-Text models and evaluate the performance of similar models to get a good comparative result. The use of an open-source base model enables very fair comparisons.
Metrics: Word Error Rate (WER), Character Error Rate (CER)

Intent analysis: Detecting whether or not a message is regarding an emergency. This will be done using a keyword detector to be able to find words in the transcribed text. Steps are taken so that they are found even if they do not match the real word perfectly by using word similarity measures.
Metrics: Accuracy, F-score

1.3 Limitations

The thesis project is inherently limited by the time allocation. 20 weeks means that steps need to be taken to ensure timely results. Due to this, limitations had to be placed on the technology used, where the very active research environment around STT might encourage one to try the latest releases, but only techniques with proven proficiency were considered during this thesis.

In addition to the scope limits put in place, there exists some limitations in form of accessible hardware and available hours with said hardware. The main computations were made on Google Cloud[4] with access to their A2-level machines with up to 8 NVIDIA A100 GPUs[5]. Due to the significant cost of these configurations, only 30 hours were allocated to the project, specifics about Google Cloud usage can be found in Appendix A. A smaller part of the computation was performed on AI-Sweden hardware, with access to a single A100 GPU for almost two weeks at the end of the project. Information about this setup can also be found in Appendix A.

The final large limitation comes from the sensitive data used, which means that very little data can be presented in the thesis and next to none made available afterwards. This also had some impact on the thesis, causing delays due to uncertainties about what was allowed and where the data could be used.

1.4 Contributions

Since the thesis is done on a very specific domain, the traditional comparisons to state-of-the-art models on bench-marking datasets do not make sense, thus the contributions are limited to showing the potential of self-supervised approaches for ASR systems in domains with domain-specific or limited data. It also highlights the benefit of open-source models for the community at large, which makes the fact that none of the models produced during the thesis can be released very unfortunate. This is due to the presence of a confidentiality agreement.

[4] `https://cloud.google.com`
[5] `https://www.nvidia.com/en-gb/data-center/a100/`

2

Theory

In this chapter, theory required to understand the thesis is presented. The reader is assumed to have a background in computer science, statistics or similar, and a sound understanding of basic machine learning concepts.

2.1 Audio

The most essential part of any Automatic Speech Recognition system is the audio. In its most basic form sounds are just vibrations, that create areas of higher and lower pressures in mediums such as gases, liquids, or solids. The sound that humans are most frequently exposed to comes from sound waves in the air. These areas of pressure differences make up a sinus wave that has an amplitude and a frequency. The frequency is measured in Hz and it is commonly referred to as pitch, which is how different tones are perceived. Amplitude on the other hand is the power of the audio signal and it is measured in decibel.

Humans have stored sounds using different methods for millennia, but older methods use some form of object where the vibrations can be recorded directly and for many years vinyl disks were used. These methods resulted in a completely lossless recording where the true vibrations are printed into the plastic, but today the physical discs are almost gone, and all the audio is stored digitally.

The digital audio format revolves around the storing of amplitude values for each time step, typically using 16-bit resolution as a way of keeping an approximation of the generated sinus waveform. This is done at different sample rates, where the rate indicates the number of samples per second. Some common sample rates are 16 kHz, 44.1 kHz, or 48 kHz [24]. The digital format also incorporates channels, that store multiple tracks of the same recording from different sources. The most common is stereo, which allows the listener to distinguish sounds that comes from the left or the right speaker to get a sense of direction. To store audio data digitally mainly two types of formats are used, compressed versions, such as MP3, and uncompressed formats like WAVE.

2.1.1 Waveform audio file format

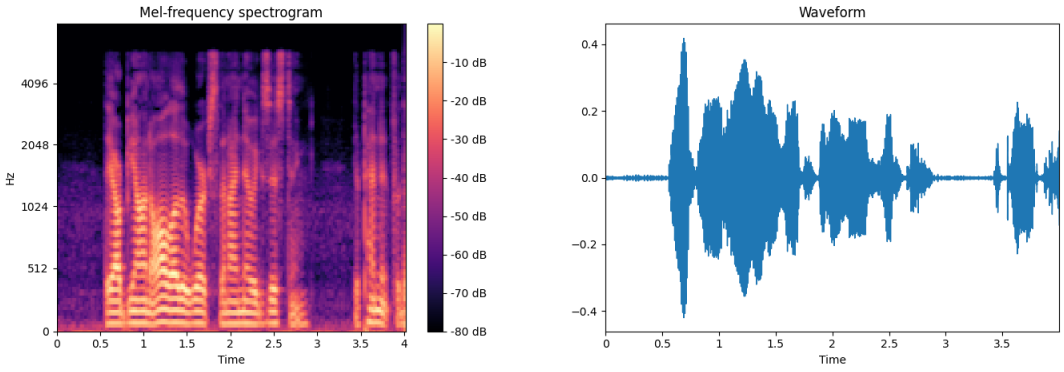
WAVE files consist of two mandatory parts, a format chunk with the file information such as sampling rate and the number of channels, and the audio data itself with amplitude values for each sampling point [25]. This results in lists of amplitude data that represent the waveform of the stored audio.

2.1.2 Free Lossless Audio Codec

FLAC[1] is both a codec and a container. It differs from the WAVE format, Section 2.1.1, by applying mathematically lossless compression rather than storing the data as-is. This means that files stored as FLAC will be smaller than their WAVE counterpart.

2.1.3 Spectrum

One way to visualise audio is to use spectrums. A spectrum shows the audio as a plot of the time, frequency, and amplitude. The x-axis represents the time and the y-axis the frequency. The amplitude of the signal at a given frequency is shown by the colour. This can be seen in Figure 2.1a where the lighter coloured areas represent a higher amplitude compared to the darker areas. This kind of spectrum shows all the important information of the audio as well as visualising where unwanted noise is present, something that the waveform plot in Figure 2.1b simply does not show.



(a) Log-mel spectrum **(b)** Waveform plot

Figure 2.1: Audio spectrum and waveform plot of a studio-grade recording of an American man talking about the style of a certain group of colourists.
"They are, beyond all other works that I know existing, dependent for their effect on low, subdued tones; their favorite choice in time of day being either dawn or twilight, and even their brightest sunsets produced chiefly out of gray paper."

2.1.4 Logarithmic Mel scale

The most common way of visualising sound is by a Logarithmic Mel spectrum [26]. Human hearing is better at differentiating the lower frequencies compared to higher

[1] `https://xiph.org/flac/format.html`

ones and human speech is generally located between 85-255 Hz [27]. Traditional frequency values do not account for this, which is why the Mel scale was introduced in 1937 to better visualise how audio pitch is perceived [28]. This is achieved by creating a function that have a lower growth rate below 1000 Hz than a regular linear function would, but the increase rate is slower above the threshold. The Mel value for a given frequency is calculated according to Equation 2.1. The frequency f is used in combination with empirical constants to give a good representation of how the audio is perceived.

$$logmel = 2595 * log_{10}(1 + \frac{f}{700}) \tag{2.1}$$

The computed Mel values in Figure 2.2 show how the Mel scale affects the frequency values.

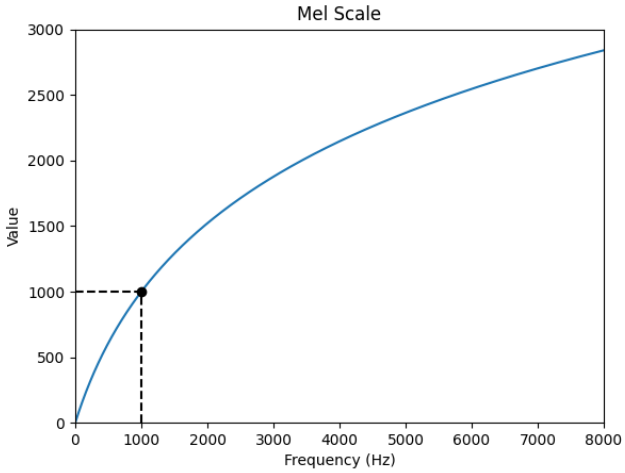


Figure 2.2: A plot showing the Mel values for frequencies in the 0-8000 Hz range.

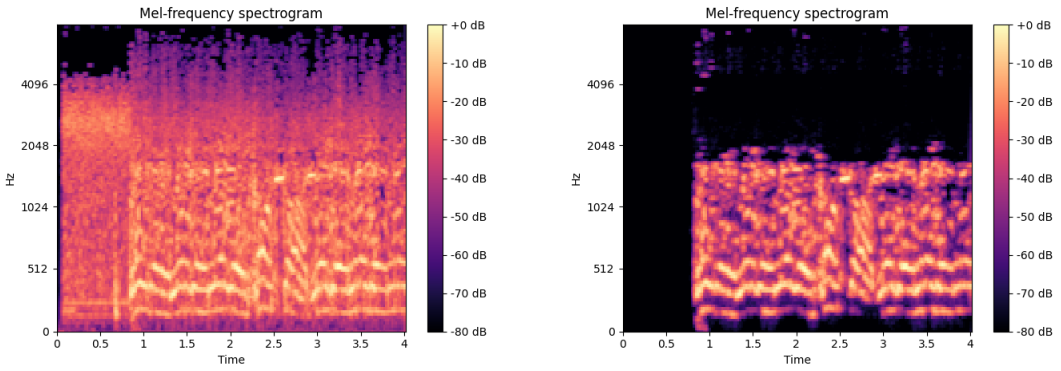
2.2 VHF radio

Very High Frequency is the name of the radio frequencies in the electromagnetic spectrum from 30 to 300 MHz. This range is split into subparts designated for different usages. One of those subparts is the naval VHF frequency range that is located between 156-174 MHz and has a range of about 60 nautical miles (110 km). To make this spectrum easier to use it is further split into 91 channels [29]. These channels have distinct purposes and the most important one is channel 16 located at 156.8 MHz. Channel 16 is the international emergency and contact channel and is monitored by all vessels, ports and the JRCC [30]. As this is an internationally recognised channel there are some rules on how it is used. For instance, when sending an emergency message one of two words should be included, either “Mayday” or “Pan” for declaring an emergency [29]. The rules also state that when an emergency message has been sent all other non-emergency traffic have to be suspended. Channel

16 is also used for contacting another vessel or a port as well as for information messages regarding weather and other dangers in the maritime environment [4]. Therefore, all vessels are required by law to monitor channel 16.

2.2.1 Issues

According to the paper by Ralston *et al.* [31] there are many sources of interference for the VHF signals, causing noise in the received audio. This noise can emerge from both technical and environmental factors, many of which are out of user control. Some examples are solar and cosmic radiation and many of these factors cannot be prevented. An example of these complexities can be seen in Figure 2.3, which contains the original and de-noised versions of the mayday call from M/V Summit Venture as it collided with the Sunshine Skyway Bridge in Tampa Bay, Florida, on May 9, 1980 [32]. Figure 2.3a clearly shows higher noise levels than the spectrum in Figure 2.3b where the speech is more prevalent. The noise makes it hard for algorithms to differentiate between the speech and the noise.



(a) Audio spectrum of a noisy sample **(b)** Audio spectrum of a clean sample

Figure 2.3: Audio spectrum of a Mayday call made by the M/V Summit Venture to the United States Coast Guard on May 9, 1980 as it struck the Sunshine Skyway Bridge, causing the bridge to collapse.

2.3 Metrics

To evaluate the different objectives of this thesis, described in Section 1.2.1, a set of metrics will be used. Since the objectives all have very different data and goals, metrics for transcription proficiency, noise measurements and common classification validation methods will be used.

2.3.1 Word Error Rate

WER is a well-established metric for measuring performance in STT systems and works by measuring how much the output text differs from the desired output [33]. This is done by aligning the two texts word by word and counting the number of

substitutions (S), deletions (D), insertions (I) and correct (C) words. The WER is then computed according to Equation 2.2.

$$WER = \frac{S + D + I}{S + D + C} \tag{2.2}$$

2.3.2 Character Error Rate

CER is very similar to WER and it is computed in much the same way. S, D, and I represent substitutions, deletions, and insertions on a character level instead of on a word level. This combined value is then divided by the number (N) of characters, resulting in Equation 2.3.

$$CER = \frac{S + D + I}{N} \tag{2.3}$$

2.3.3 Segmental Signal to Noise Ratio

One of the most common objective metrics for signal noise processing is the Signal to Noise Ratio (SNR). This way of measuring is not perfect, and a more effective metric has been defined, Segmental Signal to Noise Ratio (SSNR), that gives better results for waveform audio [34, 35]. Here segments of audio, usually around 20 ms, are used to calculate a mean, compared to the entire audio sequence as done in SNR. The SSNR is calculated as shown in Equation 2.4 where $x(i)$ is the unprocessed audio for sample i and $y(i)$ is the processed audio. When performing SSNR there is a defined segment length N and a number of segments M respectively.

$$SSNR = \frac{10}{M} \sum_{m=0}^{M-1} log_{10}(\frac{\sum_{i=1}^{N} x_m^2(i)}{\sum_{i=1}^{N}(x_m(i) - y_m(i))^2}) \tag{2.4}$$

2.3.4 Mean Opinion Score

MOS is a measure of quality where each item is given a subjective score by a reviewer (R) and the score for each item is then the mean of all the given scores by the number (N) of reviewers. There exist several different ways to score an item, but the most common way is to use a numeric score and a label. An example can be seen in Table 2.1.

To increase the robustness of the inherently inconsistent measure, several parameters should be controlled according to recommendations specified by the telecommunications industry [36]. These parameters include audio levels, speaker placement and room noise level. The computation of MOS can be found in Equation 2.5.

$$MOS = \frac{\sum_{n=1}^{N} R_n}{N} \tag{2.5}$$

Score	Label
5	Excellent
4	Good
3	Fair
2	Poor
1	Bad

Table 2.1: An example of a MOS scoring table.

2.3.5 Accuracy metrics

To get a deeper understanding of the outcome in a classification task, a common way is to evaluate the results by putting them in a truth table, as can be seen in Table 2.2. With this table it is easy to calculate many of the common accuracy metrics for classification as well as getting an understanding of where the errors come from.

Total	Condition positive	Condition negative
Predicted condition positive	True Positive (TP)	False Positive (FP)
Predicted condition negative	False Negative (FN)	True Negative (TN)

Table 2.2: Definition of a binary truth table.

Accuracy is the most used metric for classification tasks and gives the fraction of correct classifications made and is calculated as shown in Equation 2.6 [37]. This metric gives a good overview of how well a model performs on a particular dataset but might hide useful information about how the model classifies the messages.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.6}$$

From the truth table, a lot of other metrics have also been defined, some of the most common are Precision, Recall and F-score.

Precision is the metric of the fraction of the items that are classified as positive that are relevant and is calculated as shown in Equation 2.7.

$$Precision = \frac{TP}{TP + FP} \tag{2.7}$$

Recall is the metric of how many of the positive results are predicted as positive by the model. Recall is calculated as shown in Equation 2.8.

$$Recall = \frac{TP}{TP + FN} \tag{2.8}$$

F-score is a measure of the test accuracy for a classification task. It is a combined metric made up of Precision and Recall with a resulting score between 0 and 1, where

1 is when both metrics are perfect. The F-Score is popular because it takes both the evaluation methods into account and thus it calculates a more comprehensive classification score. It is shown in Equation 2.9.

$$F\text{-}score = \frac{TP}{TP + \frac{1}{2}(FP + FN)} \tag{2.9}$$

2.4 Neural network fundamentals

This section will describe the architectures used in the thesis. It will not explain MultiLayer Perceptrons (MLPs), Convolutional Neural Networks (CNNs) or Recurrent Neural Networks (RNNs) as these concepts are seen as prerequisites for understanding the extent of the content [38, 39, 40].

2.4.1 GAN

A recent development in generative algorithms that was developed in 2014 are Generative Adversarial Networks [41, 42]. These networks are mainly used to generate new samples of a given dataset, that resembles the original data enough to fool a classifier. One recent development is text-to-speech where the goal is to generate audio from a text input [43]. To achieve this the GAN architecture works by trying to fool itself by winning a zero-sum game. A zero-sum game is a game where there is one winner and one looser and where the loser loses exactly as much as the winner wins in every round. In the GAN architecture there are two parts as seen in Figure 2.4, the generator and the discriminator. These two parts compete in the zero-sum game.

Discriminator: The classifier part is called the discriminator and it is trained to separate data from noise, or separate true examples from fake but could also be trained on distinct classes. This means that the discriminator learns the general characteristic of the dataset.

Generator: The discriminator's opponent in the zero-sum game is called the generator. The task of the generators is to fool the discriminator by updating itself until it can generate data that resembles the real data enough to fool the discriminator. This allows the GAN to, for example, generate images from some input seed so that the generated image is close enough to reality so that it fools the discriminator.

2.4.1.1 Training

To train a GAN the training is split into two parts. First, the discriminator is trained on a dataset Y containing target images, noise and images containing other objects than the target images. This is done using a standard loss function for classification [44].

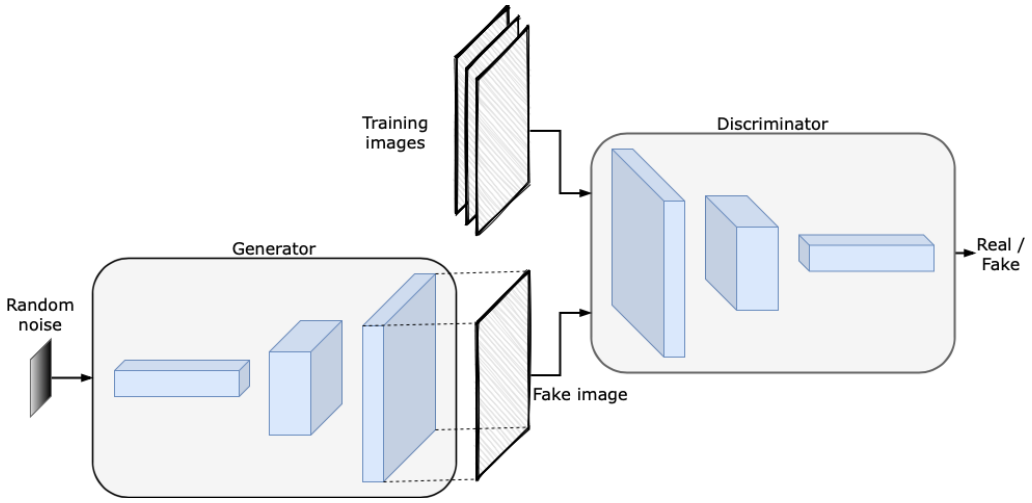


Figure 2.4: The inner workings of a GAN as well as the architecture and setup of the model.

The generator maps between a random distribution X and another distribution Y. To make the resulting image in Y differ x in X is a random seed given to the generator as can be seen in Figure 2.4. The generator iteratively learns how to turn the seed x in X into an image y in Y that is close enough to a real image to fool the discriminator. This means that when trained, one can simply change the seed x to generate a different image y. The training is defined as a min-max game between the generator (G) and the discriminator (D) according to Equation 2.10, where E is the Error function used for calculating the error for the dataset distribution P_{data} and the generators distribution P_y.

$$min_G max_D(D, G) = E_{x \sim P_{data}}[log(D(x))] + E_{y \sim P_y(Y)}[log(1 - D(G(y)))] \quad (2.10)$$

The goal is to minimise the score for the generator and maximise the score for the discriminator. This loss will cause the generator to get better and better at fooling the discriminator, resulting in more realistic images as the discriminator loss increases.

2.4.2 Auto-encoder

Commonly, most MLPs have a much larger input dimension than output dimension, this is especially true for classification networks. An example of a neural network with equal-sized inputs and outputs is the auto-encoder architecture [45]. It has two parts, an encoder and a decoder shown in Figure 2.5. The encoder is like an MLP where an input x is reduced in size to an output z with a smaller dimension than the size of x. In the auto-encoding context this value z is called an encoding vector and contains information about the original data but compressed and with reduced dimensions so that only the most important features are encoded. The other part is the decoder, and it is like the encoder in reverse so that each layer makes the

encoding vector larger and scales it to the final size. This output is called y and has the same size as the original input x.

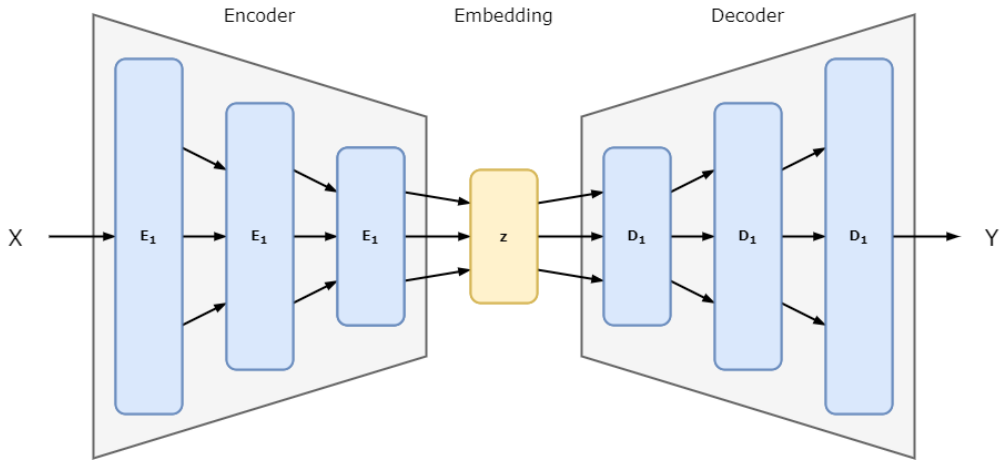


Figure 2.5: The structure of an auto-encoder, where the input is encoded into the embedding z and then decoded into an output of the same size.

Auto-encoders are used for data processing with loss, and an example of this is image compression [46]. To train a model to do data compression the network is trained on the same images as inputs X and outputs Y and learns to compress the information down to the encoding vector z and then recreate the image from that encoding without losing too much information. The network can then be used where the encoder is used to compress an image x to z, the encoding vector z is then sent to the receiver who uses the decoder to decode the encoding vector into an approximation of the original image.

2.4.3 Transformers

The previously established state-of-the-art approach for language modelling and machine translation, RNNs, and by extension Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) based networks, are inherently limited by their sequential nature [47]. Because of this limitation, several efforts have been made to push the boundaries of recurrent language models before the introduction of the significantly more parallelisable Transformer architecture.

2.4.3.1 RNN Encoder-Decoder

The Transformer is based on the idea of a two-part architecture, with an encoder and a decoder, initially presented in the paper "Learning Phrase Representations using Recurrent Neural Network Encoder–Decoder for Statistical Machine Translation" by Cho *et al.* [48]. They presented the concept of using an RNN to encode a sequence of symbols into a fixed length vector representation which could then be decoded by an additional RNN into a new sequence of symbols. This is formalised as an input sequence of symbols $(x_1, ..., x_n)$ encoded into a continuous representation $\mathbf{z}$ $= (z_1, ..., z_n)$. $\mathbf{z}$ will then be used by the decoder to generate the output $(y_1, ..., y_m)$ [47].

2.4.3.2 Architecture

The proposed architecture for the Transformers follows that of an RNN Encoder-Decoder, using stacked attention- and fully connected layers for both the encoder and decoder. This is visualised in the left and right halves of Figure 2.6.

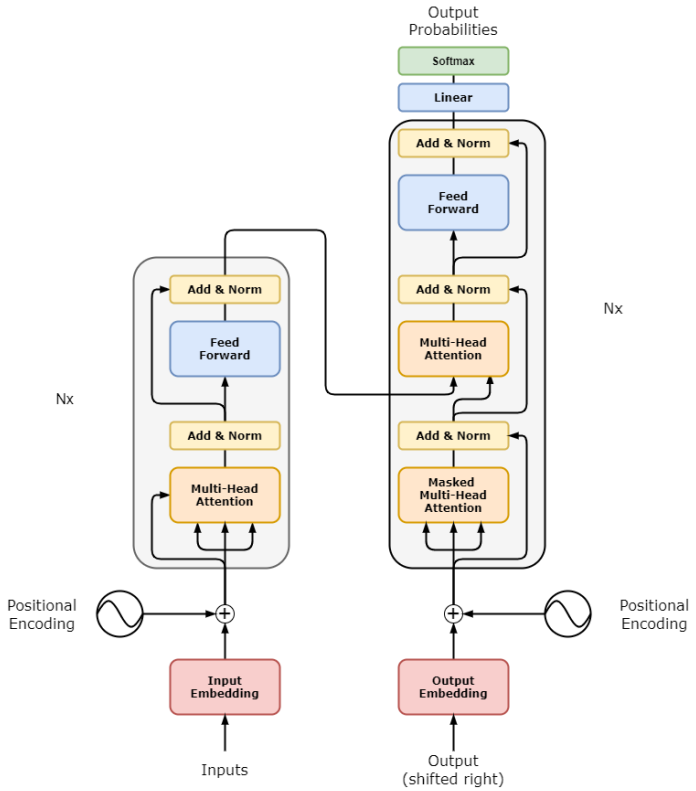


Figure 2.6: The Transformer architecture, with the encoder to the left and the decoder to the right.

Encoder: The encoder blocks consist of two sub-layers, a multi-head self-attention mechanism followed by a fully connected feed-forward layer. Vaswani *et al.* [47] also used residual connections, visualised by the arrows along the left side in Figure 2.6, taking inspiration from the paper "Deep Residual Learning for Image Recognition" by He *et al.* [49]. The output of each sub-layer is then normalised, an evolution of the more common batch normalisation [50].

Decoder: The decoder block is very similar to the encoder block, but an extra layer is added to perform multi-head attention over encoder stack output. Another addition is made as well, limiting the attention to positions before the current index i, so as not to rely on future outputs to be able to output sequentially [47].

Similarly, to other architectures for sequences, the input and output tokens are converted using learned embeddings. The output is computed using a linear layer followed by a SoftMax, to predict the next-token probabilities. Due to the success Press *et al.* [51] achieved in their paper "Using the Output Embedding to Improve Language Models" the weight matrix is shared between the two embedding layers and the linear layer.

2.4.3.3 Attention is all you need

A key to the success of the Transformer is the use of attention, which is described very concretely in the paper as "...mapping a query and a set of key-value pairs to an output. The output is computed as a weighted sum of the values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding key." [47]. In the encoder-decoder blocks, the queries come from the previous decoder block, and the memory keys and values come from the output of the encoder. The dot-product for the attention function was chosen over additive attention due to the efficiency of available matrix multiplication code run on GPUs and the attention is improved by adding a scaling factor which increases the performance when using more key values.

The concept of multi-head attention was also introduced, to allow the model to jointly attend to information from different sub-spaces. This is done by linearly projecting the input to each of the attention heads before concatenating and projecting the result. By using smaller heads compared to a single all-encompassing one the computation overhead becomes negligible.

2.4.3.4 Training

The Transformer by Vaswani *et al.* [47] uses the Adam optimiser with varied learning rate and something called warm-up steps. A step corresponds to a running of a single batch update. Warm-up steps is a method where the learning rate is increased during the initial n steps, 4000 in this case, and thereafter decreasing it by some function. Here it was decreased proportionally to the square root of the number of steps taken.

To reduce over-fitting three types of regularisation was applied, two kinds of residual dropouts and a label smoothing [47]. The term dropout refers to the dropping of units (hidden and visible) in a neural network and label smoothing prevents the largest logit in the SoftMax from becoming much larger than the others [52, 53]. The residual dropout was added to the output of each sub-layer and to the sums of the embeddings.

2.5 Speech Enhancement

SE is the process of making spoken language stand out compared to the other sounds that may be present in the audio recording. Many of the approaches and techniques for doing this are quite old and thus based on some fundamental mathematical concepts, but there are some newer additions as well. These approaches are mostly based on neural networks.

2.5.1 Fourier based noise reduction

A common method for noise reduction is the use of Fourier transforms, more specifically using Fast Fourier Transforms (FFTs) to transform the frequency domain into

the time domain [54]. FFT is an efficient algorithm for performing Fourier transformations much faster. This is initially done on a noisy sample without content to create a calibrated Fourier transform representation of the noise, this is then applied to the audio to filter out the unwanted sounds.

2.5.1.1 Audacity noise reduction algorithm

The algorithm is developed by the team behind Audacity and is based on Fourier analysis, it works by processing a piece of audio just containing noise to identify which sounds to suppress. This process is called spectral noise gating[2]. The proprietary set of statistics sampled from the pure noise is then used in a windowed fashion when processing the frequencies of the audio sample to be reduced. In total 2040 Fouriers are used which results in 1025 frequency bands being processed. This means that the audio is de-noised on a frequency level, which is why time and frequency smoothing is applied to change each frequency less drastically.

2.5.2 SEGAN

Speech Enhancement Generative Adversarial Networks (SEGANs) are an application of GAN for noise cancellation created in 2017 [22]. What differentiates the SEGAN architecture from a more traditional GAN is that the generator in the SEGAN has equal-sized inputs and outputs and uses a structure similar to an autoencoder, Section 2.4.2. The generator is implemented using one-dimensional convolutional layers and something called skips to process the audio and generate a clean audio output. Skips are connections between the layers as shown in Figure 2.7, as the output from the encoding layers is concatenated with the result from the corresponding decoding layer and then decoded by the next layer. This means that the random seed explained in Section 2.4.1 is replaced by the encoding vector from the encoder. This encoded vector is then decoded in combination with skip connections between the encoding layers and the decoding layers to get the original audio back so it can be processed by the decoder. These skip connections are used to get more information back from the original file without getting the noise as it is processed by the convolutional layers. Compared to the compression described in Section 2.4.2 the SEGAN architecture does not need to compress the file but rather attempts to keep as much detail as possible.

The discriminator part of the network is trained to classify between clean and noisy audio files. This is then used to measure how clean a sample is. The model uses strided one-dimensional convolutions as its pooling approach as the inventors of SEGAN found that the strides resulted in a more stable GAN training [22].

The model described in "SEGAN: Speech Enhancement Generative Adversarial Network" [22] is built using only convolutional layers in the encoder, decoder and in the discriminator. The reason for this is due to the sequential nature of speech and that such convolutions prohibit the interference of information from widely separated

[2] `https://wiki.audacityteam.org/wiki/How_Audacity_Noise_Reduction_Works`

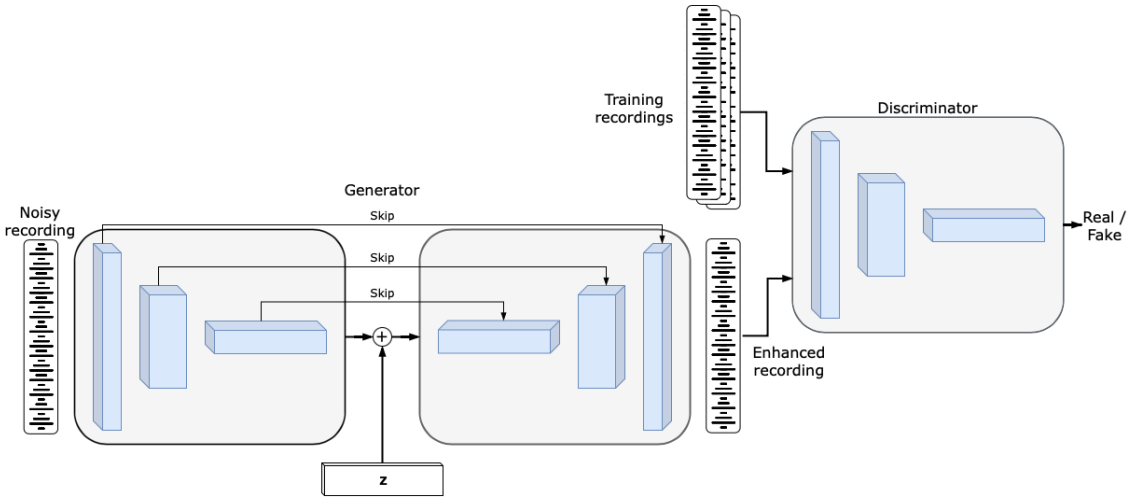


Figure 2.7: The architecture of the SEGAN. Note the skip connections between the layers of the encoder and decoder in the generator.

speech segments, thereby keeping the time-based relations intact. The convolutional layers use a Parametric Rectified Linear Unit (PReLU) activation function, PReLU is an expansion of the Rectified Linear Unit (ReLU) activation function. The difference is that PReLU has a trainable parameter a which is applied to the sub-zero-part of the ReLU shown in Equation 2.11 [55].

$$f(y_i) = max(0, y_i) + a_i min(0, y_i) \tag{2.11}$$

The parameter a is trained using back-propagation for a negligible added cost to the total training time as it can be optimised simultaneously with the other layers [55]. This results in faster convergence since the training controls how the negative activations are used.

2.6 Speech-To-Text

There exist several different solutions for turning spoken language into text and as it is an active research topic new STT architectures are released frequently. Some of these new releases include pre-trained models to make the progress available to more people, but many of the groundbreaking releases remain proprietary.

2.6.1 wav2vec 2.0

The paper "wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations" from the researchers at Facebook AI [9] presents a model that manages to achieve outstanding results using very few transcriptions by relying on a large amount of un-transcribed recordings to learn speech representations. These were then adapted to text, requiring a lot less transcribed data than traditional supervised models would need to achieve similar results. The network manages to do this

by using a strided CNN to create latent speech representations from pieces of the input waveform. The set of representations are then fed into a Transformer to create contextual representations from the set. This is done via a contrastive learning task where parts of the input representations are masked, and the Transformer tries to distinguish their true representation. To improve the training further the network learns discrete speech units to represent the CNN outputs.

2.6.1.1 Architecture

The model is composed of a single, multi-layer convolutional feature encoder $f : \mathcal{X} \mapsto \mathcal{Z}$ which takes raw audio $\mathcal{X}$ and outputs latent speech representations $\mathbf{z}_1, ..., \mathbf{z}_T$ for T time-steps. These are then sent to the Transformer $g : \mathcal{Z} \mapsto \mathcal{C}$ to create representations $\mathbf{c}_1, ..., \mathbf{c}_T$ from the entire sequence. The output of the feature encoder f is also discretized using a quantization module $\mathcal{Z} \mapsto \mathcal{Q}$ to create the targets $\mathbf{q}_t$ to be represented in the self-supervised objective [9]. A graphical representation of the structure can be found in Figure 2.8.

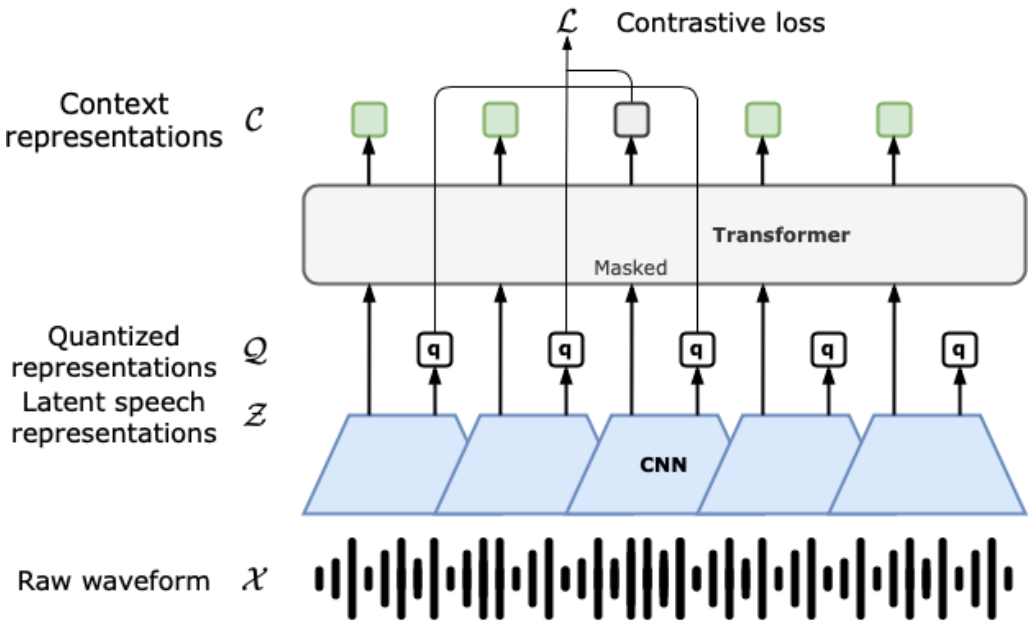


Figure 2.8: The structure of the wav2vec 2.0 architecture, showing the convolutional feature encoder, the Transformer and the different representations.

The feature encoder part of the model consists of blocks containing a temporal convolution, which means one dimensional layer, with layer normalisation and a Gaussian Error Linear Unit (GELU) activation function [50, 9, 56]. The GELU activation function is an alternative to the more traditional ReLU function and is defined in Equation 2.12, where $\Phi(x)$ is the standard Gaussian cumulative distribution function.

$$\mathbf{GELU}(x) = x\Phi(x) \tag{2.12}$$

The Transformer follows the architecture explained in Section 2.4.3 but with a convolutional layer instead of fixed positional embeddings, similar to the implementation in "Transformers with convolutional context for ASR" by Mohamed and his team at Facebook AI [57].

Due to the prior success in using discrete units a quantization module was implemented, discretizing the output of the feature encoder $\mathbf{z}$ using product quantization. This amounts to choosing representations from codebooks and concatenating them [58]. These G codebooks, or groups, with V entries each, are then implemented using a Gumbel SoftMax, which is a fully differentiable way to choose and learn the discrete representations [59]. This means that $\mathbf{z}$ is mapped to $\mathbf{l} \in \mathbb{R}^{G \times V}$. The equation for doing this quantization is shown in Equation 2.13, where the probabilities for choosing representation v from group g are computed, τ is a non-negative temperature, $n = -\log(-\log(u))$ and u are uniform samples from $\mathcal{U}(0, 1)$.

$$p_{g,v} = \frac{\exp(l_{g,v} + n_v)/\tau}{\Sigma_{k=1}^{V} \exp(l_{g,k} + n_k)/\tau} \tag{2.13}$$

2.6.1.2 Pre-training

The pre-training is done by masking a subset of the outputs from the feature encoder before feeding them to the Transformer, which is optimised using the objective function in Equation 2.14 where α is a hyperparameter.

$$\mathcal{L} = \mathcal{L}_m + \alpha \mathcal{L}_d \tag{2.14}$$

The first part, $\mathcal{L}_m$ is a contrastive loss where the model needs to identify the true representation $\mathbf{q}_t$ in a set of $K + 1$ distractors $\tilde{\mathbf{q}} \in \mathbf{Q}_t$. The set of distractors are uniformly sampled from other masked time steps of the same utterance.

The loss is calculated according to Equation 2.15, where $sim(\mathbf{a}, \mathbf{b}) = \mathbf{a}^T\mathbf{b}/\|\mathbf{a}\|\|\mathbf{b}\|$ and κ is the temperature.

$$\mathcal{L}_m = -\log \frac{\exp(sim(\mathbf{c}_t, \mathbf{q}_t)/\kappa}{\Sigma_{\tilde{\mathbf{q}} \sim \mathbf{Q_t}} \exp(sim(\mathbf{c}_t), \tilde{\mathbf{q}})/\kappa} \tag{2.15}$$

The second part of the loss, $\mathcal{L}_d$, is a diversity loss built upon the Gumbel SoftMax in Equation 2.13 where the loss is designed to make use of the codebook representations. The loss is computed according to Equation 2.16.

$$\mathcal{L}_d = \frac{1}{GV} \sum_{g=1}^{G} -H(\bar{p}_g) = \frac{1}{GV} \sum_{g=1}^{G} \sum_{v=1}^{V} \bar{p}_{g,v} \log \bar{p}_{g,v} \tag{2.16}$$

2.6.1.3 Fine-tuning

After the network has been pre-trained on the available audio data it needs to be fine-tuned using transcribed data in order to perform well. This fine-tuning is done by adding a linear projection on top of the network to output the vocabulary, $\mathcal{Q} \mapsto \mathcal{Y}$. The training of this layer is done using a Connectionist Temporal Classification (CTC) loss, which is a function that produces a label for every time step of the recording. These labels are, in the case of wav2vec 2.0, letters or blank for indicating silence or lack of speech which are then used to compute the cross-entropy loss [60]. In order to improve the final error rates and reduce overfitting for small, labelled datasets a version of SpecAugment is applied to the audio, which masks some time-steps and channels during the training [61].

2.7 Language Models

The key concept of a Language Model (LM) is the ability to determine the probability of a given sentence or generate probable words given a set of preceding ones [62]. The ability to assign probabilities to sentences allows such a model to select the more probably alternative given a couple of inputs from a noisy source, such as speech recognition. The addition of an LM gives the system the ability to realise that "I will be back soonish" is a much more probable sentence than "I will be bassoon dish" [62]. For this reason, the use of LMs are very common to decrease the WER of different systems, an example of this is given in the paper by Baevski *et al.* [9].

2.7.1 N-Grams

One of the most common and intuitive ways to create an LM is to use an N-gram. The model works by computing word frequencies from a sample of text and then use those to compute the probability of a sentence [62]. Due to the creative nature of language, this approach needs some modification to get around the vast number of sentences that would appear with zero probability. The solution is to make a Markov assumption, which is what allows the simplification in Equation 2.17, where the probability is approximated by the N previous words rather than all the words.

$$P(w_n|w_{1:n-1}) \approx P(w_n|w_{n-N+1:n-1}) \tag{2.17}$$

A bigram model, N=2, looks at the current word and one preceding word instead of the complete sentence. A trigram model, N=3, looks at two preceding words and so on. This means that the function for computing the probability of the next word in a bigram model is approximated as in Equation 2.18.

$$P(w_n|w_{1:n-1}) \approx P(w_n|w_{n-1}) \tag{2.18}$$

The probability of a given sentence is computed as the product of the word probabilities and can be seen in Equation 2.19, which uses the bigram approximation.

$$P(w_{1:n}) = \prod_{k=1}^{n} P(w_k|w_{k-1}) \tag{2.19}$$

The previously mentioned word frequencies that were computed from the sample of text are used to compute the probabilities for bigrams, which can be seen in Equation 2.20 where the counts are used.

$$P(w_n|w_{n-1}) = \frac{C(w_{n-1}w_n)}{C(w_{n-1})} \tag{2.20}$$

2.7.1.1 Interpolated Kneser-Ney smoothing

One problem with N-gram LMs is that they do not know what to do when a known word appears after a word that it has never appeared after in the sample of text used to create the N-gram. In order to avoid that these unseen word combinations automatically are assigned a zero probability, a way of smoothing needs to be applied to them. This can be done in many ways by adding fake occurrences [62].

One of the most used and best performing N-gram smoothing algorithms is Kneser-Ney smoothing [62]. The smoothing is based on absolute discounting and it works by redistributing the probability mass by subtracting a number of discounts from each count greater than 0 and changing the N-gram distribution mass by adding these removed counts to zero count N-grams [63]. The absolute discount for a bigram is calculated by Equation 2.21 where a discount d is removed from all cases where the count $- d$ is higher than 0.

$$P(w_i|w_{i-1}) = \frac{C(w_{i-1}w_i) - d}{\sum_{w_i} C(w_{i-1}w_i)} + (1 - \lambda_{w_{i-1}})P_{abs}(w_i) \tag{2.21}$$

The equation for Interpolated Kneser-Ney smoothing is shown in Equation 2.21 and uses the normalising constant λ to re-distribute the probability mass, discounted by the absolute discounting algorithm shown in Equation 2.22.

$$P_{KN}(w_i|w_{i-1}) = \begin{cases} \frac{max(C(w_{i-1}w_i)-d,0)}{C(w_{i-1})}, & \text{if } C(w_{i-1}w_i) > 0 \\ \alpha(w_{i-1})P_{KN}(w_i), & \text{otherwise} \end{cases} \tag{2.22}$$

This gives the ability to make guesses on word combinations never encountered without having zero probabilities that can hurt the system performance. Equation 2.21 is presented for bigrams but can easily be extended to account for more previous words using Equation 2.17, allowing this rule to generalise for all N.

2.8 Word similarity measures

One of the simplest forms of NLP is keyword detection, which is the process of figuring out if a particular word is present in the text. This is quite fragile, which is why there exists several algorithms to compute word similarity. This can be used to create a more robust keyword detection system.

2.8.1 Hamming distance

One of the simplest ways of measuring the similarity between two words is to use the Hamming distance function. It simply compares the strings character by character to check if they are equal, if the characters are not equal the distance is increased by one [64]. This is often used with normalisation as the Hamming distance between a string and an empty string results in the length of the longest string, therefore normalisation is applied to bring this value down to a value between 0 and 1 as the distance otherwise is based on the word length. Equation 2.23 shows how to compute the Hamming distance between two strings, where $tail(a)$ is everything but the first character in the string and $|a|$ is the length or number of characters.

$$ham(a,b) = min \begin{cases} |b| & \text{if } |a| = 0 \\ |a| & \text{if } |b| = 0 \\ ham(tail(a), tail(b)) + 0 & \text{if } a[0] = b[0] \\ ham(tail(a), tail(b)) + 1 & \text{otherwise} \end{cases} \tag{2.23}$$

2.8.2 Levenshtein distance

An extension of the basic Hamming distance is the Levenshtein distance [65]. The Levenshtein distance for two strings a and b is given recursively by $lev(a,b)$, shown in Equation 2.24 where $tail(a)$ is all of a except the first character and $|a|$ is the number of characters in a.

$$lev(a,b) = \begin{cases} |b| & \text{if } |a| = 0 \\ |a| & \text{if } |b| = 0 \\ lev(tail(a), tail(b)) & \text{if } a[0] = b[0] \\ 1 + min \begin{cases} lev(tail(a), b) \\ lev(a, tail(b)) \\ lev(tail(a), tail(b)) \end{cases} & \text{otherwise} \end{cases} \tag{2.24}$$

2.8.2.1 Damerau–Levenshtein

The Levenshtein distance has a problem, which is when two letters have switched positions and therefore the Damerau–Levenshtein distance was invented [66]. The Damerau–Levenshtein distance uses an extended recurrent algorithm where a new

option for substitution is added as shown in Equation 2.25 where $dtail(a)$ is added to represent a double tail, that is all but the two first characters. The notable difference is the last line, that handles the transposition:

$$dlev(dtail(a), dtail(b)) + 1 \quad \text{if } |a|, |b| > 1 \text{ and } a[0] = b[1] \text{ and } a[1] = b[0]$$

$$dlev(a,b) = min \begin{cases} 0 & \text{if } |a| = |b| = 0 \\ dlev(tail(a), b) + 1 & \text{if } |a| > 0 \\ dlev(a, tail(b)) + 1 & \text{if } |b| > 0 \\ dlev(tail(a), tail(b)) + 1 & \text{if } |a| > 0 \text{ and } |b| > 0 \\ dlev(dtail(a), dtail(b)) + 1 & \text{if } |a|, |b| > 1 \text{ and} \\ & a[0] = b[1] \text{ and } a[1] = b[0] \end{cases} \tag{2.25}$$

2.8.3 Longest common substring similarity

One way of figuring out how well a word matches another word is to find the longest substring that is present in both strings [67]. This can be done by using a general suffix tree. This tree allows the algorithm to find all substrings between two strings. An example is ABCCD and BCCAD, here BCC is the longest substring but there are also many substrings of shorter lengths such as CC, BC, A, B and more. The goal is to compare the words by finding how large the longest substring is compared to the total word length.

2.8.4 Match Rating Approach

The MRA algorithm differs from the other word similarity algorithms by being phonetic [68]. It was developed by Western Airlines in 1977 to handle homophonous names, that is names that are pronounced the same but spelt differently. An example of a set of homophonous words are *rain*, *reign* and *rein*. The algorithm builds on three distinct parts, rules for encoding and comparison, and predefined minimum ratings decided by the length of the string. All comparisons are done on encoded strings.

Encoding

1. Delete all vowels unless the vowel is the first character.
2. Delete the second consonant of all double consonants.
3. Create codex using first three and last three characters at most.

Comparison

1. Only compare if length difference is less than three.
2. Get minimum rating of combined length from the table.
3. Remove identical characters, left to right.
4. Remove characters found in both words, right to left.
5. Subtract the number of unmatched characters in the longer string from six.
6. This value, the similarity rating, is considered good if greater than or equal to the minimum rating value from the table.

Table

Sum of Lengths	Minimum Rating
≤ 4	5
$4 < sum \leq 7$	4
$7 < sum \leq 11$	3
$= 12$	2

Following these rules when comparing the two names Smith and Smyth result in the encoded strings SMTH and SMYTH with a minimum rating of 3. By applying the rules from the comparison table one gets a rating of 5, which indicates that the names are very similar.

3

datasets

This chapter contains information about the datasets that were used during the model implementation and evaluation. The thesis is built around the large VHF transmission dataset from the JRCC but had to be extended with two open-source speech datasets to facilitate the early development and comparison of available models. The main reason for the use of DataShare, a dataset with parallel clean/noisy samples, was due to availability issues with the JRCC data in the early stages of implementation, whereas the LibriSpeech dataset is the standard for bench-marking STT models.

3.1 The JRCC recordings

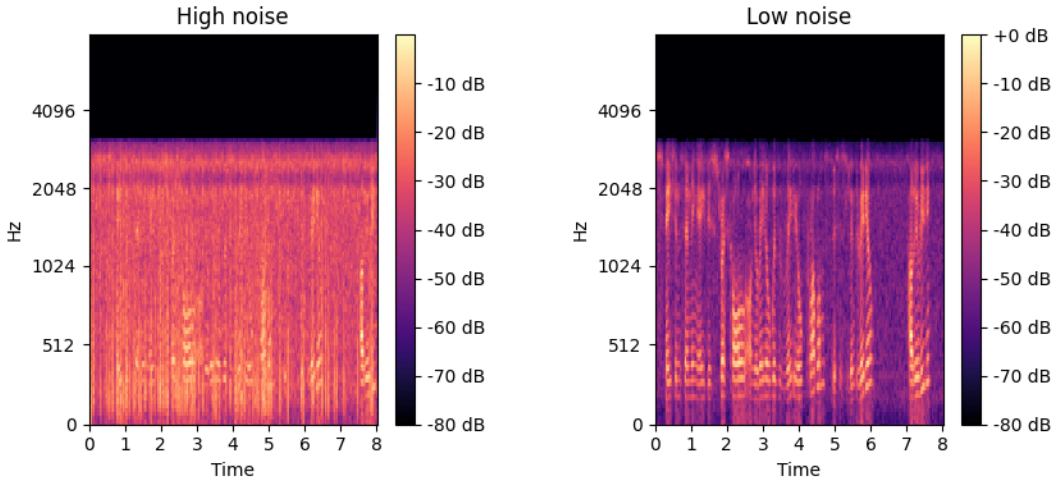
The main dataset used in this thesis is the VHF transmissions recorded by the JRCC, containing about one year's worth of data from the masts in the coastal radio network [5, 4]. The dataset is made up of 2.7 million recordings, which results in roughly 11 000 hours and a total size of around 1.3 TB. The recordings are in the uncompressed WAV file format described in Section 2.1.1 sampled at 16 kHz with one channel, summarised in Table 3.1.

Sample rate:	16 000 Hz
Resolution:	16 bit
Channels:	1 (mono)
Encoding:	WAV

Table 3.1: Properties of the sound files in the JRCC dataset.

To create a coherent result visualisation, a pair of recordings of the same message but with different audio quality will be used throughout the report, Figure 3.1. This message is selected to provide meaningful insights and highlight the impact of the quality of the recording. The pair is created due to the multiple mast setup mentioned in the introduction.

Due to the sheer size of the dataset, it is difficult to provide details about the composition, but the key points are the skewed nature of it, with very few interesting messages mixed up among thousands of weather forecasts and ship to ship commu-



(a) Noisy version, MOS: 1.5 **(b)** Clean version, MOS: 4

Figure 3.1: Two versions of the same recording from the JRCC dataset with their Mean Opinion Score.
"station sending a distress call on your radio with radio problems this is lyngby radio how do you read me"

nication. It also contains several copies of the same message picked up from different masts, which should yield good comparisons but also plenty of unhearable messages.

A couple of subsets were created in order to develop the different models. They are mutually exclusive unless stated otherwise.

3.1.1 The categorised recordings

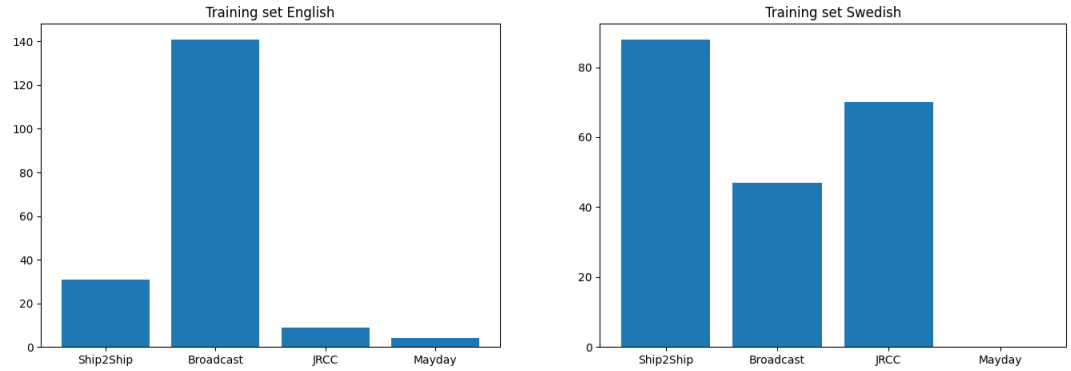
A subset of the JRCC recordings was created, where the recordings were categorised according to the set of classes in Table 3.2. This was done at an early stage intended to serve as a basis for the direct classification mentioned in Section 7.1.4 but mainly served as a tuning set for the keyword detection.

Mayday	All messages containing some form of call for aid.
JRCC	Messages containing any mention of "Sweden Rescue", "JRCC" or similar.
Broadcast	General broadcasts, for example weather reports.
Ship2Ship	Communication between ships and other stations

Table 3.2: A list of the classes used to categorise the JRCC recordings and their definitions.

The dataset consists of 389 recordings in total with 205 in Swedish and 184 in English. These datasets, especially the English subset, have an imbalanced distribution with a low frequency of messages regarding the JRCC or emergencies.

This is visualised by the histogram in Figure 3.2 and presents some challenges during the keyword detector training.



(a) Class distribution English **(b)** Class distribution Swedish

Figure 3.2: The class distribution for the English data.

3.1.2 Transcriptions

The dataset was also extended with transcriptions, which were done in collaboration with the JRCC to enable a higher quantity of messages but with retained accuracy. The transcriptions were done through a proprietary tool, where the recordings were transcribed and tagged to the classes in Table 3.2.

To get as much consistency in the transcription process as possible, a few guidelines was put in place:

- Write what you heard word by word. you can add punctuation marks such as period, comma and question marks as you see fit.
- Write numbers as words , "twelve" instead of "12". Be careful if it is twelve or one two.
- No special characters. Write "e" instead of "é" and "dollar" instead of "$".
- Don't use acronyms if it is not said explicitly.
- If you can't make out a word or part of a sentence, replace it with a question mark "?".
- If you write a word that you think you hear but are unsure about, mark that file as unsure.
- If it is impossible to hear anyone talking in the file mark it as only noise.
- Mark the audio quality, where 1 is unhearable and 5 is very good audio quality.

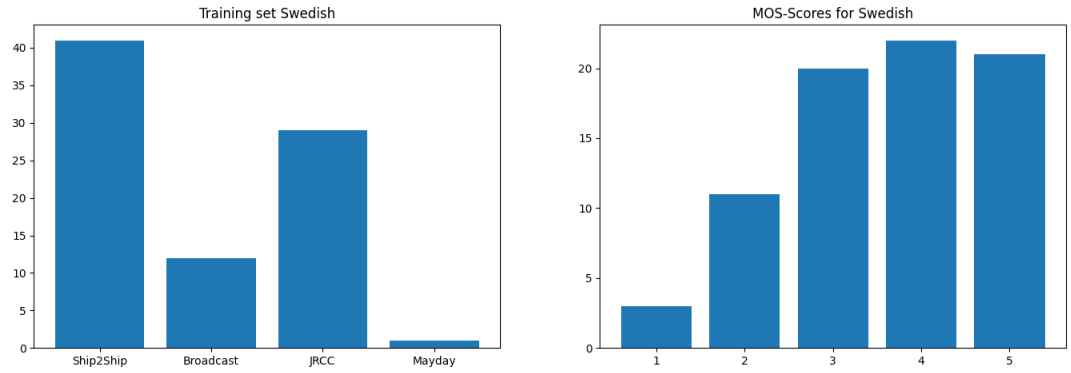
To evaluate the transcribers and to find areas where the above-mentioned rules needed to be extended, all transcribers started with the same set of 20 samples.

From these 20 samples a few conclusions were drawn, where the biggest problem was the identification of ship names. Many other words that may not have been so clear could still be transcribed as the brain fills in the blank spots as described in a recent paper by L. Gwilliams *et al.* [69].

Due to limited time with the tool before the hardware access deadline was up, the number of transcriptions ended up on the smaller side of the spectrum. 1493 recordings were transcribed, but over one thousand of those could not be used. The main reason for this was that they could not be completely transcribed, meaning that one or a couple of words were too hard to hear for proper transcription, other reasons were that they contained other languages than Swedish or English. This resulted in three sets of data, with 79 recordings in the Swedish set, 259 in the English one and 338 in the combined dataset with both languages. The validation sets consisted of 10 recordings for each language sampled from the previously mentioned set of 20 to ensure the highest accuracy.

3.1.2.1 Swedish

The Swedish training set contains 79 recordings with a combined MOS score of 3.56. Half of it is made up of Ship2Ship recordings and the JRCC class makes up most of the other half as seen in Figure 3.3. This makes the dataset very imbalanced as a classification set and results in problems with accurate classification training.

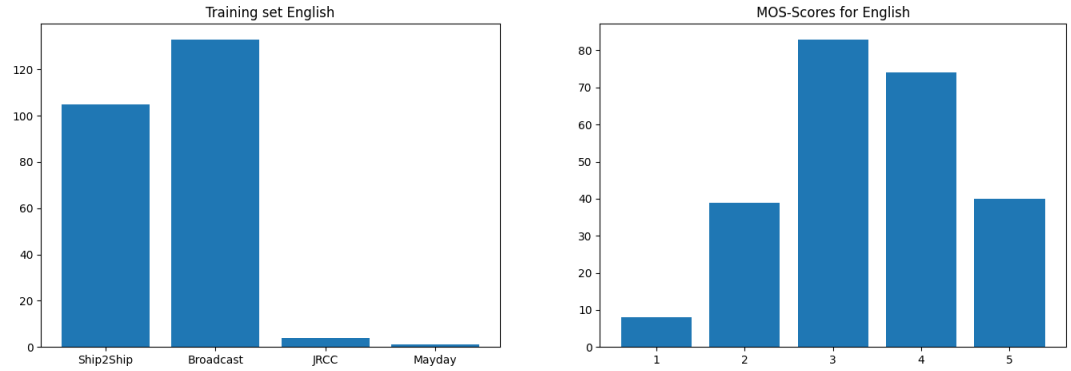


(a) Class distribution **(b)** MOS-Score

Figure 3.3: The class distribution and the MOS distribution for the Swedish training dataset.

3.1.2.2 English

The English set is a bit larger than the Swedish with 259 samples and has a bit lower MOS score, 3.324, but the most noticeable aspect is the imbalanced representation of data classes. This is similar to the Swedish part of the dataset and can be seen in Figure 3.4.



(a) Class distribution **(b)** MOS-Score

Figure 3.4: The class distribution and the MOS distribution for the English training dataset.

3.1.3 The small sample

At the start of the thesis, some of the JRCC data were available. It consisted of 18 samples were 15 could be properly transcribed and were in either English or Swedish. These files have a total size of 8 MB with a total transcription length of 350 words. The dataset had 10 English and 5 Swedish recordings.

3.1.4 Text corpora

The sample of text that will serve as the corpus for generating LMs is simply the transcriptions, formatted in such a way that each recording gets treated as one line. The Swedish corpus consists of 999 words, the English one 6069 and the combined set consists of 7068 words.

3.2 Clean-noisy DataShare

Due to limited access to the JRCC data in the early stages of the thesis, another audio set had to be used during the development of the SEGAN. This parallel clean/noisy data[1] comes from the University of Edinburgh and the small test set was used to validate the initial SEGAN model [70]. This data is collected from 28 speakers, sampled at 48 kHz with different types of background noise added to the noisy version of the clips, the specifics can be found in Table 3.3. The total size for these two parallel sets is roughly 310Mb.

[1] `https://datashare.ed.ac.uk/handle/10283/2791`

Sample rate:	48 000 Hz
Resolution:	16 bit
Channels:	1 (mono)
Encoding:	WAV

Table 3.3: Properties of the sound files in the DataShare dataset.

3.3 LibriSpeech

The dataset consists of roughly 1000 hours of 16 kHz read English speech, it is derived from the LibriVox project [19]. LibriVox is a free public domain of audiobooks. LibriSpeech or some of the subsets in other languages serve as the typical benchmark for newly developed STT models, the sound file properties can be found in Table 3.4 and the list of subsets in Table 3.5.

Sample rate:	16 000 Hz
Resolution:	16 bit
Channels:	1 (mono)
Encoding:	FLAC

Table 3.4: Properties of the sound files in the LibriSpeech dataset.

dev-clean	337Mb	development set, "clean" speech
dev-other	314Mb	development set, "other", more challenging, speech
test-clean	346Mb	test set, "clean" speech
test-other	328Mb	test set, "other" speech
train-clean-100	6.3Gb	training set of 100 hours "clean" speech
train-clean-360	23Gb	training set of 360 hours "clean" speech
train-other-500	30Gb	training set of 500 hours "other" speech

Table 3.5: The different subsets of the LibriSpeech dataset.

4

Methods

The following sections aim to describe the methods used. The sections are in chronological order and each contains two steps, first the implementation with the details around it followed by a section about how the method was evaluated to prepare for the results chapter.

This paragraph is intended to help the reader understand the workflow of the following sections and ensure that their purpose and succeeding implications are clearly stated. Due to the initial hypothesis, where Speech Enhancement would be used to enhance the performance of the Speech-To-Text service, the method needs to be seen as three parts. The first part details the attempted solution and validation of said hypothesis, that is the Speech Enhancement of Section 4.1 and the Google STT validation details in Section 4.2. The second part marks the start of the new approach with the development of a domain-specific STT model rather than improving an established one. This can be found in Section 4.3. The third and final part contains the intent analysis, which due to the time allocation is limited to robust keyword detection. These word similarity algorithms are detailed in Section 4.4. While the first part is intended to be seen as separate, the other two parts are highly connected, with the evaluation of the algorithms performed on the transcriptions from part two.

The final and definite approach can be summarised as follows, the self-supervised Speech-To-Text model wav2vec 2.0 was used to enable intent analysis on the VHF transmissions. The intent analysis was then done using robust keyword detection, where five different word similarity metrics were evaluated.

4.1 Exploring Generative Adversarial Networks for Speech Enhancement

Initially, the main problem was assumed to be the level of noise in the recordings, which is why a Speech Enhancement model was developed using GANs. The main reason for choosing this network architecture instead of a mathematical algorithm like the one Audacity has developed, more details in Section 2.5.1.1, or a more traditional structure like a RNN was the possibility to use the unprocessed JRCC recordings during development. Both the Audacity algorithm and a RNN require

additional data or non-trivial data processing; the RNN needs a noisy and a clean signal during training and the Audacity needs an input with pure noise resembling the one in the recording at run-time. The internals of a GAN means that it is possible to train the generator and discriminator using a dataset where both audio versions are available and then improve the generator by training it on the noisy data from the JRCC. This can be done because the discriminator and generator are separate, allowing the discriminator to be frozen when using the noisy data. This benefit in combination with the good results achieved in several research papers, the recentness of said papers and the fact that the technology is still developing all serve as good reasons for the use of SEGAN in the application [22, 71, 72, 73].

The implementation of the SEGAN was modelled on both the framework and structure presented in the paper by Pascual *et al.* [22], to allow for a quicker initial evaluation of the concept. This version of SEGAN is implemented in TensorFlow. As mentioned in Section 3.2, this initial development could not be done on the domain data, but it would be enough to create the model and verify it as a proof of concept. It was implemented in such a way that it would easily be integrated into a system with the Google API at its core and allow for easy extension and modification when the actual data became available. The first working version had just been completed and trained on the small test sample of the DataShare dataset when the focus had to be shifted to the Google STT service evaluation due to the arrival and nature of the initial results from that research. Thus, the results presented in Section 5.1 are very brief but present an accurate view of the current state of development.

4.1.1 Evaluation

To evaluate the SEGANs processing quality the generated output was evaluated using both objective and subjective metrics. As described in Section 2.3 there are two main metrics for SE efficiency. The MOS score was evaluated by the authors of this thesis. Normally MOS needs to be evaluated in a quiet room using standard equipment, something that is further explained in Section 2.3.4. To make the use of MOS feasible, a few limitations are needed. First and foremost, many of the proposed recommendations require standardised equipment and will thus have to be ignored. In addition to that, the number of reviewers will be very small due to the lack of time and funds for proper evaluation. The SSNR was calculated using the evaluation script written in MATLAB provided by H. Pan[1] and MOS in combination with SSNR were used to evaluate the SEGAN performance. Both methods were used as they are perceived in different ways and together give a comprehensive picture of how clear one sample is compared to another. Both these measures are used to compare two samples, so a clean sample without the noise was used for the SSNR calculation. As SE is all about enhancing audio, the important thing is the difference between a raw sample and a processed one. Therefore the MOS score was collected for both samples and then the relative MOS score was calculated.

[1] `https://github.com/pquochuy/sasegan`

4.2 Using Speech-To-Text services

Based on the research presented in the introduction, the STT service of choice was the one provided by Google [5]. The key would therefore be the application of SE to improve the performance of this state-of-the-art solution on lower quality recordings according to the hypothesis. This meant creating a baseline by just using the API on the small sample set of recordings to generate transcriptions and WERs.

4.2.1 Speech Enhancement for Google's STT API

To improve the initial performance of the STT service, different kinds of SE were used. The first one was audio level-based sequence cutting to try and split the audio file into several words and sentences that could be sent to the API. The second was to evaluate the limits of the API using Gaussian noise. This was added to a clean audio file that had received a perfect transcription from the API previously. The noise was added with increasing SNR until the transcription started to miss words or classify them incorrectly. The results should show how much noise the system can handle before starting to fail and thus validate the hypothesis.

Audio splitting: To enhance the spoken parts and remove unwanted sounds an audio level splitter was used to split the samples into smaller subsections. The audio level setting was used in two ways to achieve different outcomes, one to split the samples into sentences and one to split them into words.

Audacity noise reduction: To perform more standard noise reduction on the audio before being processed by the STT API the Audacity noise reduction algorithm described in Section 2.5.1.1 was applied to the audio.

4.2.2 Evaluation

The key performance metric of STT implementations is Word Error Rate. This was used in conjunction with manual observations of the transcriptions, to see how well the context was preserved. As previously mentioned, the assumptions were that the API would perform very well on recordings with limited noise, indicating the need for SE to provide good results on noisier recordings. This assumption did not align with the results received during testing and despite significant efforts to use features like speech contexts[2] to ease the transcription of certain words or phrases the performance did not reach acceptable levels. Detailed configuration parameters are available together with the results in Section 5.2.

4.3 Creating a domain-specific STT model

After realising that the approach based on the Google STT would not provide sufficient performance to be a usable system, the logical next step became to research

[2] `https://cloud.google.com/speech-to-text/docs/speech-adaptation`

the possibility to develop a domain-specific STT model that could handle both the peculiar language and the noisy environment. The biggest challenge with this approach was the lack of transcriptions, which severely limited the available options, but it was quickly discovered that there had been incredible progress made in this domain recently. The best performing STT model at the time of writing was created using a limited amount of transcribed data. The model, called wav2vec 2.0, was developed by Alexei Baevski and his team at Facebook AI and made available to the public in both pre-trained and simple architecture versions [9].

Thus, the choice of trying an approach with wav2vec 2.0 was mainly based on the performance achieved using limited transcriptions and the possibility to use all the available data even without transcribing it. The model comes in two versions, referred to as the base size and the large size respectively. The models share encoder setup, seven blocks of the encoder layer described in Section 2.6.1, which results in chunks of 25ms. They then differ in their Transformer setup, where the base version has a smaller configuration with 12 blocks and an inner dimension of 768, which results in around 95 million parameters. The large configuration is comprised of 24 such blocks with a bigger dimension, 1024, and ends up with 317 million parameters. Since the JRCC dataset is comprised of multiple languages, the large model was selected as the base for further training. Mainly because that version has shown good results on multilingual data in previous research [20].

4.3.1 Setup

One of the keys to the successful implementation of wav2vec 2.0 is the toolkit named Fairseq, which is intended to simplify the training of models for text generation tasks [74]. This meant that the implementation came with a lot of infrastructure which quickened the setup of the model and allowed the use of more advanced optimization techniques such as the one found in Hydra[3]. Hydra is a framework that allows for easy configuration of hardware, thanks to its integration in Fairseq. The wav2vec 2.0 pre-trained models can be found on Fairseq's GitHub[4], or by using the Hugging Face hub[5]. Hugging Face supplies the Transformers[6] library that contains pre-trained models for easy integration in applications.

The Fairseq toolkit comes with a set of scripts to perform the necessary data processing and preparation, these scripts can be found in the GitHub repository[7], where the edited version of the transcription formatting script can also be found. This was necessary since the structure of the transcribed JRCC data differed from that of the transcribed LibriSpeech files.

With the setup and data processing complete, the model would then be pre-trained

[3] https://github.com/facebookresearch/hydra

[4] https://github.com/pytorch/fairseq/tree/master/examples/wav2vec

[5] https://huggingface.co/models?filter=wav2vec2

[6] https://github.com/huggingface/transformers

[7] https://github.com/JonathanGildevall/Automatic-Emergency-Detection-in-Naval-VHF-Transmissions

on all the available data to learn the speech representations, this model would then have to be fine-tuned using the transcription datasets created in collaboration with the JRCC operators.

4.3.2 Pre-training

The initial plan was to pre-train the large model in a similar fashion to the one done by Baevski *et al.* [9], that is, running the training for roughly two days or 250 000 updates, but after experimenting with different amounts of computing power using Google Cloud[8] it was deemed infeasible within the limits of the thesis. A detailed calculation for the cost of training can be found in Appendix A, but the summary is that such a training would end up costing around $10 000.

4.3.2.1 Transfer learning on Google Compute Infrastructure

The cost factor meant that a different approach was needed, where the results from the thesis "Transfer learning for domain-specific automatic speech recognition in Swedish" indicated that transfer learning could achieve good results [75]. The thesis also showed good results for English models extended to support Swedish, which is why the large model was selected for this transfer learning experiment. The concept has been widely researched and discussed in detail, where Conneau *et al.* [20] does so with the same model in mind. They discuss cross-lingual transfer learning, showing that it can produce very good results when extending a high-resource model for a low-resource language. The paper also touches upon *inference*, where the extension to include more languages can decrease the performance for the high resource language but explains that this can be alleviated by increasing the model capacity. There is a pre-trained model available from this paper, the XLSR-53, which has been trained on 53 different languages available on the Fairseq GitHub[9] together with the other versions of wav2vec 2.0. This could have served as a baseline for the transfer learning, but the cost factor and the uncertainty connected with extending a multilingual model meant that the XLSR-53 would only be tested using fine-tuning as a reference.

Training configuration: The training of the large model was done on the cloud instance with eight Nvidia A100-GPUs set up in such a way that the batch size would be equivalent to the one used for the initial training, i.e., 2.7 hours. The large model had been pre-trained for 600 000 updates, or batches, before being extended with another 7 530 updates on the JRCC dataset. This corresponds to roughly two epochs, meaning that the data is run through the model twice. More information about the specific configuration can be found in Appendix A.

4.3.3 Fine-tuning

The process of fine-tuning a wav2vec 2.0 is the part where the learned speech representations are adapted to represent letters, this is done by mainly training an added

[8] `https://cloud.google.com`

[9] `https://github.com/pytorch/fairseq/tree/master/examples/wav2vec`

linear layer to output one of the letter tokens using a SoftMax. The exact configuration varies between the languages but the size correlates to the number of needed tokens, where a token is needed for each letter or symbol as well as tokens for silence. The key to the fine-tuning is the use of the CTC loss mentioned in Section 2.6.1, where the added layer determines what the learned representations mean in a letter context. The fine-tuning was done using the three transcription datasets mentioned in Section 3.1.2, with 79, 259 and 338 recordings respectively. To compare these to the numbers found in the paper about wav2vec 2.0 [9], the amounts need to be converted to minutes. The Swedish set consists of roughly 22 minutes, the English 96 and the combined set consists of 118 minutes, in the paper they had transcription sets in the order of 10 minutes, 1 hour, 10 hours and 960 hours.

Every fine-tuning was done over 13 000 updates, but with a slightly reduced learning rate for the Swedish version due to the smaller size. The complete configuration can be found in Appendix B. This was done on AI-Sweden compute resources, where a single A100 had been allocated for the duration of the training, the specifics can be seen in Appendix A.

4.3.3.1 Swedish

Initially, two models were scheduled for fine-tuning on the Swedish dataset: the large model with transfer learning on the JRCC data and the XLSR-53 model. The XLSR-53 was selected since it had shown good performance in other applications, with excellent results on the Swedish part of LibriSpeech. This model has also been made available[10] and will be used for comparison of the other results.

These plans were then extended when Facebook AI released a new set of pre-trained models[11], this time on the VoxPopuli dataset [76]. The raw data for this set was collected from 2009-2020 European Parliament event recordings. The large model pre-trained on the Swedish subset and the model pre-trained on the entire dataset was fine-tuned on Swedish to see how a model purely trained on Swedish would compare to a multilingual model containing the same data.

4.3.3.2 English

On the English dataset, the JRCC pre-trained model was fine-tuned together with the large model without transfer learning using the JRCC data to allow for a direct comparison between it and the other large implementations with fine-tuning on LibriSpeech.

The English fine-tuning was also extended with a VoxPopuli model, mainly due to the good results the larger model got from the Swedish fine-tuning but also to get a point of comparison separated from LibriSpeech.

[10] `https://huggingface.co/KBLab/wav2vec2-large-xlsr-53-swedish`
[11] `https://github.com/facebookresearch/voxpopuli`

4.3.3.3 Swedish-English

For the combined dataset a single model was fine-tuned, the XLSR-53, since the JRCC pre-trained model showed issues with handling a single language it was skipped. The Swedish-English dataset tuning was mainly done as an experiment to show how the performance is impacted by attempting to transcribe multiple languages with the same model rather than two separate. The benefit of such a model is obvious, as it eliminates the need for language detection.

Due to the poor results and a lack of time the combined fine-tuning was not done using the VoxPopuli model.

4.3.4 Language Models

To reach state-of-the-art performance on the LibriSpeech dataset the wav2vec 2.0 model had to incorporate a Language Model (LM), which was used to improve the letter selection during decoding [9]. The decoding with an LM uses the wav2letter++ beam search decoder that allows the letter selection to use the probability information of the selected LM [77]. Two different types of LMs were tested, a 4-gram model and a Transformer, both trained on the transcriptions from LibriSpeech. Because of the substantial performance improvement that the addition of an LM generated, it was decided to attempt something similar for the domain-specific version as well. The biggest difference is the size of the corpora, where Baevski *et al.* had 960 hours of transcribed recordings as the base for their LMs. The biggest of the three used here are based on less than two hours. The impact of the small size is discussed at length in Section 6.3.2.

4.3.4.1 Transformers

The best results achieved with the wav2vec 2.0 model was using a Transformer based LM. This is a huge model with over 562 million parameters and training requirements in the hundreds of GPUs and large corpora [78]. Due to this, the use of a similar model is left as future work, as the already limited GPU allocation was needed for the fundamental wav2vec 2.0 training.

4.3.4.2 N-grams

Since the addition of even a simple 4-gram LM decreased the Word Error Rates for the wav2vec 2.0 model, attempts were made to create a similar model for the domain. Two models were created from each corpus, the 4-gram as well as a simpler 3-gram. The addition of the 3-gram comes from conclusions drawn in a prior thesis in a similar domain, where the 3-gram was deemed sufficient and more space-efficient than the comparable 5-gram [75].

The N-grams were built using KenLM[12], which is an efficient implementation of the Kneser-Ney algorithm described in Section 2.7.1.1. These LMs were then used as an

[12] `https://kheafield.com/code/kenlm/`

aid for the wav2vec 2.0 model, to see if the addition of proper language knowledge could improve the transcription quality.

4.3.5 System implementation

The result from the training is a PyTorch checkpoint file, which needed to be converted to be easily integrated into a system not reliant on Fairseq. This process is eased by the fact that the previously mentioned Hugging Face library supports and delivers several of the complete wav2vec 2.0 models, allowing them to be easily integrated. The collaboration means that there exists a script for converting the checkpoint to the correct format as this is the intended workflow. By using the Transformer library the model could then easily be incorporated into the rest of the system like a regular model would have been. The script for converting and an example implementation using the transformer library can be found in the GitHub repository[13].

4.3.6 Evaluation

The initial evaluation of the models is straightforward, where the fine-tuned versions are used to transcribe the validation recordings, first without an LM and then with the two different versions of the N-grams. This was done for each of the models on the corresponding language version of the sets, e.g., the large model fine-tuned on Swedish was validated on the Swedish dataset. The WER and CER for this were then used as basic performance measures, but due to the intended usage, it is equally important to review the transcriptions themselves, to make sure that the intent is not lost. This is especially important when the LMs are used. The effect of the LMs are especially interesting in comparison to the Google API. Additionally, the difference between the WER and CER will be highly interesting to see, since wav2vec 2.0 is letter-based rather than word or word-piece.

4.4 Keyword detection

When working with letter-based Speech-To-Text models the possibility of misspelt words needs to be accounted for to perform successful keyword detection. Especially in this case since the model is based on the concept of speech representations and the languages are full of similar sounds and even silent phonemes. To handle these cases the detection will be performed using word similarity measures so that words that are similar enough will be detected. This should in theory allow for the detection of "maida" as "mayday" with the correct implementation.

To achieve the best final classification score, a few text distance algorithms will be tested. All these algorithms are implemented in a library called textdistance[14]. These five algorithms in total all have different advantages and disadvantages as

[13] `https://github.com/JonathanGildevall/Automatic-Emergency-Detection-in-Naval-VHF-Transmissions`

[14] `https://pypi.org/project/textdistance/`

they approach the problem of text comparison a bit differently. Three of the algorithms are based on letter-by-letter comparisons where the baseline is the Hamming distance as explained in Section 2.8.1. The Hamming distance gives a quick letter-by-letter difference between the two words. This algorithm is then extended in two steps by the Levenshtein difference and further by the Damerau–Levenshtein difference. These two extensions of the Hamming distance takes more than character by character comparisons where the Damerau takes letter switching into account for example. The last two algorithms are the Match Rating Approach (MRA) and the Longest Common Substring Similarity (LCSS) measure. The MRA is a phonetic measure developed to find phonetic differences between spoken surnames. As the problems faced by the keyword detector are mainly phonetic since the wav2vec model is based on speech embedding recognising phonemes, MRA should perform well on the task.

To make these algorithms easier to use a normalisation based on the word length is used to get a similarity score between 0 and 1 which makes the classification easier. In order to use these text similarity measures, a threshold has needed to be set to determine when a word is similar enough to the keyword. To find these thresholds a grid search was implemented, where a threshold a was tested for values between 0 and 1 for each word in the list of words. This tuning was performed on the categorised recordings explained in Section 3.1.1 as this is a set that was not used for tuning the STT models. This linear search was performed from the highest threshold to the lowest to try and use the highest threshold possible as many thresholds could have the same performance.

4.4.1 Evaluation

To evaluate the total performance of each algorithm every word was tuned individually, and the final classification scores were evaluated where a sample was classified as an emergency. This was done with an OR operation for all keyword classifications. To find the best threshold the accuracy described in Section 2.3.5 was used.

All the algorithms were then evaluated for every STT model, where the F-score, Equation 2.9 and the accuracy, Equation 2.6, were recorded. The accuracy only takes the final classification score into consideration as this is the result that is interesting for the operators at JRCC. This leaves a problem where many of the emergency messages do not contain any of the keywords in the list used by the detector and therefore are impossible to classify correctly but no more advanced NLP is implemented in this thesis as it is outside the scope.

5

Results

This chapter contains the results and performance figures for the tested models. It is important to note that the division of the sections are made to reflect that of the methods used. This means that the two initial sections, 5.1 and 5.2, contain the results of the initial approach and are thus completely disconnected from the rest of the chapter. The final sections, 5.3 and 5.4, are however very tightly connected. This comes from the fact that the keyword detection needs to be tuned to the kind of text it receives, which comes from the wav2vec 2.0 models in this case.

5.1 SEGAN

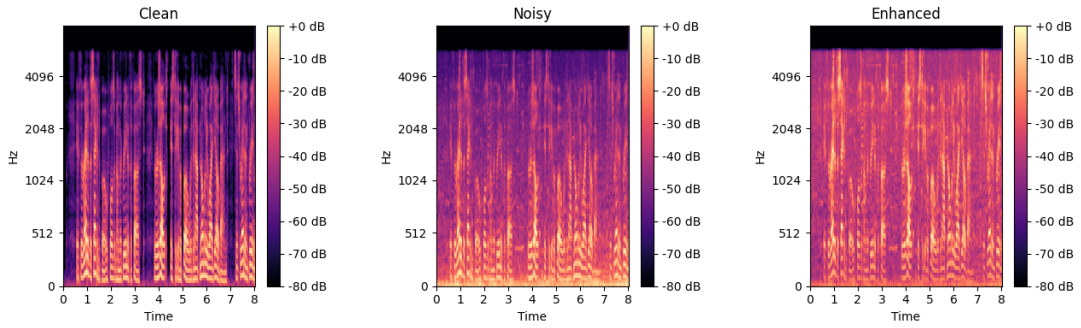
Due to the quick shift in focus from SE development to STT the SEGAN architecture was only trained and evaluated on the clean-noisy Datashare dataset described in Section 3.2. The goal was to show the state of development as the focus shifted. The results presented in Table 5.1 show a small improvement for the enhanced audio over the noisy data.

Samples	SSNR
Noisy-Clean	6.30
Noisy-Processed	0.44
Processed-Clean	6.57

Table 5.1: The SSNR of the audio samples from the SEGAN test.

This is quite hard to see in the spectrograms in Figure 5.1. This is because while the speech is more prevalent in the enhanced sample, Figure 5.1c, the volume is reduced and the overall quality worse. Hence the washed out picture. The speech in the noisy sample, Figure 5.1b, is less prevalent but the quality of the recording is higher. This is why the picture is a closer resemblance to Figure 5.1a despite the worse result. Thus the SSNR is better for the enhanced sample without generating a better looking spectrum. A more detailed explanation to how to interpret spectrograms can be found in Section 2.1.3.

The evaluation shows a small improvement for the SEGAN enhanced audio sample but nothing close to the results presented in the paper by Pascual [22]. Results of



(a) Clean sample **(b)** Noisy sample **(c)** Enhanced sample

Figure 5.1: Spectrums showing the frequency distribution and amplitudes for three version of a sample from the DataShare dataset. The three samples are a clean version, a noisy and one enhanced by the SEGAN. The enhanced sample shows how the frequency distribution has changed and the peaks have been smoothed over the entire spectrum.

a five point SSNR increase for the SEGAN architecture, which in combination with our findings show that the concept of using GANs to perform speech enhancement is promising.

5.2 The Google STT service performance

The evaluation of the Google Speech-To-Text API was performed on the only available data at the time, the small sample mentioned in Section 3.1.3. It contains two parts, firstly, the results of the evaluation of different API configurations, and secondly, the results of the tests to see how the audio quality affected the performance. The complete transcriptions and spectograms for each sample are available in Appendix C.

A set of different configurations were evaluated. The default configuration just uses the API[1] in its simplest form with the language specified. The Swedish (SV) configurations uses language code *sv-SE* and the English (EN) configurations uses *en-US*. Note that the *sv-SE* configuration also manages English and can automatically process parts of the input as Swedish or English, as can be seen in the appendix where some transcriptions contain both languages. The Command configuration uses a parameter called *model*, where the *command_and_search* choice is "Best for short queries such as voice commands or voice search" according to the API description. There is also a *phone_call* model that is tailored to lower sampling rates and lower quality, but this model was not available for Swedish. The final configuration uses *speech_context* which allows for the specification of a set of words and phrases to aid the transcription of these, so that the model gets a hint on what to look for. For these tests the phrases and words found in the table on the next page were used:

[1] `https://cloud.google.com/speech-to-text/docs/reference/rest/v1/RecognitionConfig`

speech_context
mayday
pan
allmänt anrop
sweden rescue
sjöräddningsenhet
sunningesundet
all stations
alla båtar

These were tailored to the dataset and should thus help the model to achieve optimal performance. Table 5.2 contains the configurations in a summarised manner.

Configuration	Parameters
Default SV	`{language_code="sv-SE"}`
Default EN	`{language_code="en-US"}`
Command SV	`{language_code="sv-SE", model="command_and_search"}`
Command EN	`{language_code="en-US", model="command_and_search"}`
Speech Context SV	`{language_code="sv-SE", speech_contexts=[`*speech_context*`]}`
Speech Context EN	`{language_code="en-US", speech_contexts=[`*speech_context*`]}`

Table 5.2: Table containing the parameters for the different Google STT API configurations.

All Word Error Rate (WER) results were run on a normalised version of the received transcriptions, where numbers are written in text-form since Google automatically converts some of the given numbers from the text representation. The letters are all lower cased. Table 5.3 contains the WERs for each configuration when applied to every recording.

Configuration	WER	
	SV	EN
Default	66.57	73.43
Command	69.14	67.14
Speech Context	62.86	69.71

Table 5.3: WER for the different Google configurations by language.

In table 5.4 the WERs are adjusted so that the configurations are correctly applied by language, meaning that the EN versions were used for English and the SV versions were used for Swedish.

Configuration	WER	
	SV	EN
Default	53.40	66.41
Command	71.59	57.25
Speech Context	54.55	62.21

Table 5.4: WER for the different Google configurations, grouped by correct language.

In order to get a sense of what the best case performance of the API could be, the best WER for each transcription was selected and the final result of that was 48.57%, but this includes non-trivial combinations of results from all the different configurations.

5.2.1 Transcription examples

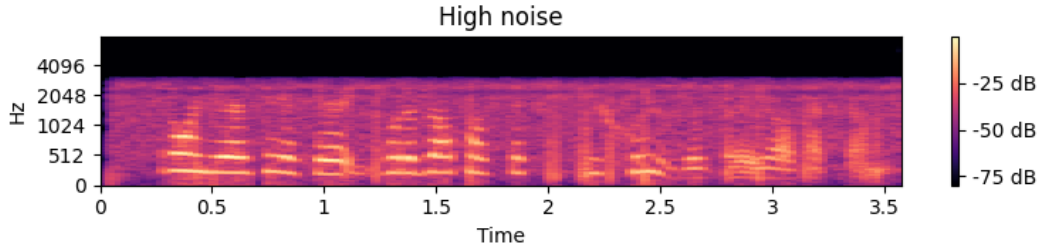
The WERs presented in the tables above convey two things, that the results are way worse than the Google API can manage on more familiar speech recordings and that it is very difficult to interpret the combined WER numbers. Therefore the two examples below are presented to show the underlying issues discovered through the testing, giving a different look at the performance compared to the WERs.

The clearest example is the result of the transcription with the lowest WER, Table 5.5, where the WER of just 14% is competitive in a lot of scenarios and that the number can probably serve as a bench-mark for the other approaches. But when looking beyond the WER, analysing the actual transcription, it quickly becomes clear that the WER is not enough to judge the performance. When looking at the result from this angle, the transcription falls flat when it comes to the issue of context and the conveying of intent. Because the information about both the port and ferry "visby" has been scoured from the recording it is impossible to understand or benefit from the transcription, therefor defeating the purpose of enabling intent analysis.

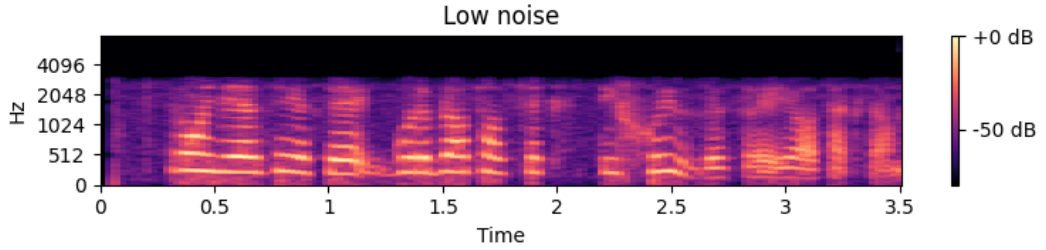
Configuration	Transcription
Reference	all stations all stations all stations information concerning the port of visby information concerning the port of visby the ferry visby will enter the port within ten minutes
Speech Context English	all stations all stations all stations information concerning the port of little information concerning the port of the state will enter the port within 10 minutes

Table 5.5: The transcription with the lowest WER from the small JRCC sample.

The other example is intended to show the impact of VHF noise on the transcription quality and is made possible by the fact that the small sample contained a message recorded through different masts. The spectrograms and the best transcriptions can be seen in Figure 5.2. The result from the noisier version, Figure 5.2a, shows a significantly worse WER than the API managed to achieve on the cleaner sample. The lower WER of the sample, Figure 5.2b, does not mean that it is good enough, as a closer look reveals that it is not enough to convey the intended message.



(a) Log-mel spectrum of a JRCC recording with high noise



(b) Log-mel spectrum of a JRCC recording with low noise

Figure 5.2: Spectrums of two JRCC recordings containing the same message but with different noise levels. Note the brighter colours in the top, high noise, spectrum.

	Transcription	WER
Reference	från jrcc går det bra att ta sextiosju så är jag tacksam kom	
High noise	Ronja GPS Går det bra på en 67 därför jag kom	54
Low noise	från tncc Går det bra 67 så är jag tacksam	31

5.2.2 The effects of Speech Enhancement

Some simple SE tests were performed as well, to investigate the impact of the noise on the transcription proficiency. The results of a test where various levels of Gaussian noise was added to a clean recording can be seen in Table 5.6. This was done to see at which level the added noise caused the Google API to fail. The table shows that when Gaussian noise with an SNR of five was added the issues start to appear, which is a lot of background noise that severely impacts the listening experience. Therefor it was concluded that cyclic noise was not the sole reason for the poor performance as high levels of noise could be present in the audio before the API started to make mistakes.

SNR	Transcription
Clean	Ericsson var givetvis först över gränsen när Kanadas premiärminister förhandlat nytt fredsavtal
20	Ericsson var givetvis först över gränsen när Kanadas premiärminister förhandlat nytt fredsavtal
15	Ericsson var givetvis först över gränsen med Kanadas premiärminister förhandlat nytt fredsavtal
10	Erik som har givit först över gränsen med Kanadas premiärminister förhandlas nytt fredsavtal
5	Ericsson i Olsfors över gränsen med Kanadas premiärminister
2	Eriksson över gränsen med Kanadas premiärminister
1	Ericsson julpyssel över gränsen med Kanadas premiärminister

Table 5.6: The resulting transcriptions when Gaussian noise was added for different SNRs.

5.3 The domain-specific STT model

The wav2vec 2.0 based models whose performance is presented here come from a few different sources, as stated in Section 4.3, and some have been fine-tuned on other datasets as well. This allows for a thorough evaluation of the impact the different data have on the resulting performance. All of the models are based on the large configuration of the wav2vec 2.0 model presented in the paper by Baevski *et al.* [9], which is the configuration with 24 transformer blocks and 317 million parameters. The table below presents a complete list of the different models and explains their origin, so that the performance number presented in Table 5.7 and the details presented in the following sections make sense.

Pre-trained models fine-tuned on thesis data	
LARGE	Pre-trained on LibriSpeech
JRCC	The large model with JRCC transfer learning
XLSR-53	Pre-trained on multilingual data
LARGE_VOX	Pre-trained on the European Parliament dataset
LARGE_SV	Pre-trained on the Swedish subset of the European Parliament data
Complete models	
XLSR-SV	XLSR-53 extended with Swedish pre-training and fine-tuning
LARGE_10m	Fine-tuned on 10 minutes of LibriSpeech
LARGE_960h	Fine-tuned on 960 hours of LibriSpeech

5.3.1 The WER and CER results

Table 5.7 contains the results of all the wav2vec 2.0 models that were evaluated on the three validation sets. For each model, the resulting WER and CER are presented in the rightmost columns, with the used LM to the left of the percentages. The unlabelled data column highlights the datasets used to pre-train each model, the table below contains a detailed description of all the sets used.

MLS[a]	Multilingual LibriSpeech (8 languages, 50.7k hours): Dutch, English, French, German, Italian, Polish, Portuguese, Spanish
CommonVoice[b]	(36 languages, 3.6k hours): Arabic, Basque, Breton, Chinese (CN), Chinese (HK), Chinese (TW), Chuvash, Dhivehi, Dutch, English, Esperanto, Estonian, French, German, Hakh-Chin, Indonesian, Interlingua, Irish, Italian, Japanese, Kabyle, Kinyarwanda, Kyrgyz, Latvian, Mongolian, Persian, Portuguese, Russian, Sakha, Slovenian, Spanish, Swedish, Tamil, Tatar, Turkish, Welsh
BABEL[c]	(17 languages, 1.7k hours): Assamese, Bengali, Cantonese, Cebuano, Georgian, Haitian, Kazakh, Kurmanji, Lao, Pashto, Swahili, Tagalog, Tamil, Tok, Turkish, Vietnamese, Zulu
SR	1000 hours of Swedish from various radio stations
LibriSpeech	960 hours of English, see Section 3.3
LV-60k[d]	60 000 hours+ of English, superset of LibriSpeech
JRCC	The JRCC recordings, see Section 3.1
VoxPopuli[e]	(23 languages, 100k hours) English, German, French, Spanish, Polish, Italian, Romanian, Hungarian, Czech, Dutch, Finnish, Croatian, Slovak, Slovenian, Estonian, Lithuanian, Portuguese, Bulgarian, Greek, Latvian, Maltese, Swedish, Danish
VoxPopuli_sv	(4.5k hours) Swedish, subset of VoxPopuli

[a] https://indico2.conference4me.psnc.pl/event/35/contributions/3585/attachments/1060/1101/Wed-2-6-10.pdf
[b] https://commonvoice.mozilla.org/en/languages
[c] https://catalog.ldc.upenn.edu/byyear
[d] https://github.com/facebookresearch/libri-light
[e] https://github.com/facebookresearch/voxpopuli

The bold text in the top right of each box represent the dataset used to compute the scores and, for the models below the thin line, fine-tune the models. An example is the LARGE model in the first box, which has been pre-trained on LV-60k and JRCC, then fine-tuned on the **Swedish** dataset and achieved a WER of 100% without an LM.

The complete set of transcriptions for the validation datasets can be found in Appendix D.

Model	**Unlabelled data**	**LM**	valid	
			WER	CER
Swedish				
XLSR-SV	MLS, CommonVoice, BABEL, SR	None	78.74	39.58
JRCC	LV-60k, JRCC	None	100	100
		3-gram	100	100
		4-gram	100	100
XLSR-53	MLS, CommonVoice, BABEL	None	77.16	30.39
		3-gram	70.08	46.59
		4-gram	70.08	46.59
LARGE_SV	VoxPopuli_sv	None	100	90.20
		3-gram	100	96.45
		4-gram	100	96.45
LARGE_VOX	VoxPopuli	None	62.99	28.27
		3-gram	59.21	38.21
		4-gram	59.06	38.21
English				
LARGE_10m	LibriSpeech	None	89.11	44.21
		3-gram	67.36	45.76
		4-gram	67.36	45.76
LARGE_960h	LibriSpeech	None	65.80	29.91
		3-gram	59.07	42.76
		4-gram	59.07	42.76
JRCC	LV-60k, JRCC	None	100	100
		3-gram	100	100
		4-gram	100	100
LARGE	LV-60k	None	42.48	18.25
		3-gram	33.68	26.31
		4-gram	34.72	26.91
LARGE_VOX	VoxPopuli	None	43.01	17.14
		3-gram	40.41	27.68
		4-gram	40.41	27.68
Swedish-English				
XLSR-53	MLS, CommonVoice, BABEL	None	99.69	88.51
		3-gram	98.75	95.85
		4-gram	98.75	95.85

Table 5.7: WER and CER for the different wav2vec 2.0 models.

5.3.1.1 Swedish

Model	Unlabelled data	LM	valid	
			WER	CER
Swedish				
XLSR-SV	MLS, CommonVoice, BABEL, SR	None	78.74	39.58
JRCC	LV-60k, JRCC	None	100	100
		3-gram	100	100
		4-gram	100	100
XLSR-53	MLS, CommonVoice, BABEL	None	77.16	30.39
		3-gram	70.08	46.59
		4-gram	70.08	46.59
LARGE_SV	VoxPopuli_sv	None	100	90.20
		3-gram	100	96.45
		4-gram	100	96.45
LARGE_VOX	VoxPopuli	None	62.99	28.27
		3-gram	59.21	38.21
		4-gram	59.06	38.21

The table above contains the Swedish section of Table 5.7, where one can clearly see that the late addition of the VoxPopuli based models was worth the effort. It provided both a very interesting comparison to the poorly performing JRCC model and the best results for both WER and CER. The outcome of the transfer learning is detailed in Section 5.3.2. The LARGE_VOX model managed to outperform the current state-of-the-art model for Swedish, XLSR-SV, and the more general XLSR-53 by a notable amount, whereas the model solely trained on Swedish did not perform at all. This phenomenon is discussed in Section 6.3.3.

One can also see the impact of the Language Models, which decreased the WER but made a noticeable dent in the resulting CER. The example below aims to highlight the reason for this behaviour.

Model	LM	Transcripton
Reference	-	DU KOMMA PÅ SEX SJU SEXTIOSJU SEX SJU
LARGE_VOX	None	TVÅ KAN DO KOLMAR PÅ SEX SJU SEXTISJU FLN SEXSKJU
	4-gram	SEX SJU SEXTIOSJU SEX SJU

In this case all the words in the transcription using an LM is correct, but it has managed to remove the start of the sentence, which results in a higher CER but also highlights the issue with the use of LMs. Since it uses probabilities to build sentences it can deem some words too improbable to show up depending on the context. In the case above there is no harm done, but in the case on the next page the issue really shows itself.

Model	LM	Transcripton
Reference	-	BRO ANNA BRO ANNA STENA GERMANICA
LARGE_VOX	None	KRO ANNA FREGJÄNNAR SENE ER MA NIKE
	4-gram	TRANSPONDER

In the transcription above the entire context is lost, which makes it impossible to detect the intent of the message. This is a recurring pattern throughout the results and therefor it is discussed at length in Section 6.3.2

5.3.1.2 English

Model	**Unlabelled data**	**LM**	valid	
			WER	CER
English				
LARGE_10m	LV-60k	None	89.11	44.21
		3-gram	67.36	45.76
		4-gram	67.36	45.76
LARGE_960h	LV-60k	None	65.80	29.91
		3-gram	59.07	42.76
		4-gram	59.07	42.76
JRCC	LV-60k, JRCC	None	100	100
		3-gram	100	100
		4-gram	100	100
LARGE	LV-60k	None	42.48	18.25
		3-gram	33.68	26.31
		4-gram	34.72	26.91
LARGE_VOX	VoxPopuli	None	43.01	17.14
		3-gram	40.41	27.68
		4-gram	40.41	27.68

The conclusion of the validation on the English dataset were some pretty impressive results, where the best CER is way below the expected value. The other key highlight of these results are the performance of the LARGE_960h model fine-tuned on LibriSpeech. This models achieved a WER of 3.3%, 4.5% without a LM, on the "difficult" other test set but only managed 59.07% on the JRCC data. This highlights the odd nature of both the quality and the type of language used, which differs a lot from the language found in LibriSpeech. It also emphasises the significance of the other results.

The impressive results of the LARGE and LARGE_VOX model are showcased on the next page, so that the differences in their transcriptions can be seen.

Model	Transcripton
Reference	MAYDAY I DELTA KILO TWO EIGHT SIX THREE SKALDET THIS IS LYNGBY RADIO ARE YOU STILL ON CHANNEL ONE SIX
LARGE	MAA KI TELA KILO TWO EAIGH SIX THREESGUARNER THIS IS LYNGBY RADIO YOT SIL ON CHANNEL ONE SIX
LARGE_VOX	MAIDA K DELFHA KILO TWO AIH SIX THREESCULNE THIS IS LYNGBY RADIO Y O STIL ON CHANNEL ONE SIX

Model	Transcripton
Reference	AIDAMAR PLEASE COME CHANNEL SIXTY FIVE SIX FIVE
LARGE	AIMA PLEASE COME TO CHANNEL SIXTY FIVE S FIVE
LARGE_VOX	ADA MAR PLEASE COME CHANNEL SIXTY FIVE SIX FIVE

These two examples aim to highlight the fact that despite the minuscule difference in CER, the LARGE_VOX model tends to produce higher quality transcriptions with noticeably better context preservation.

5.3.1.3 Swedish-English

Model	**Unlabelled data**	**LM**	valid	
			WER	CER
Swedish-English				
XLSR-53	MLS, CommonVoice, BABEL	None	99.69	88.51
		3-gram	98.75	95.85
		4-gram	98.75	95.85

The results of this experiment was disappointing, but perhaps not that surprising. The combination of two languages is a challenge even with good prerequisites. In this case the two languages create to many uncertainties without bringing enough data to allow clarity. The example transcription below shows that the model does not only provide very bad performance numbers, but it also provides bad actual performance.

Model	Transcripton
Reference	STATION CALLING THIS IS SOLSTRAND ZERO SIX ZERO SIX
XLSR-53	SON CHVT

5.3.2 The outcome of the transfer learning

Due to the nature of the two part training of the wav2vec 2.0 implementation, the actual validation of the pre-training could not be done until the second step, the fine-tuning, had been done. During this process and during the evaluation it became clear that something was not right with the models pre-trained on the JRCC data, evident by the horrible WER and especially when looking at the transcriptions themselves. The model only managed to produce "E", "S" or " " no matter the input, indicating some issues with the direct use of low quality data for pre-training. These results became more interesting after the fine-tuning of the LARGE_SV model, which to some extent showed the same issues. It did however manage to produce actual attempts, as shown below.

Model	Transcripton
Reference	TRYGGVE TRYGGVE HÄR ÄR HELIKOPTER LIFEGUARD NOLL NOLL TRE SEXTON
LARGE_SV	SEEE EE

Thoughts and comments on the peformance of the models with domain-specific transfer learning can be found in Section 6.3.1.

5.3.3 The performance lower quality recordings

To see how well the best performing model, LARGE_VOX, handles recordings with lower quality a simple test was performed on the two samples presented in Figure 3.1. This exemplifies how the excess noise affects a model that has been exposed to very limited noise throughout its training.

Model	Transcripton	WER	CER
Reference	SENDING A DISTRESS CALL ON YOUR RADIO WITH RADIO PROBLEMS THIS IS LYNGBY RADIO HOW DO YOU READ ME		
High noise	STA DANI NA N E CAL OUR TI I RA KOAEM HISI LYNBY RADIO HOW D YOU READM	85.0	41.90
Low noise	SENDING A DSSTRES CAL ON YOUR RADIO WITH RADIO PROBLEMS THIS IS LYNGBY RADIO HOW DO YOU READ ME	10.0	2.86

The result on the high noise sample is a lot worse than the low noise counterpart, but it still manages to make out some key information from a recording the authors have trouble hearing themselves. The impact of this is further discussed in Section 6.3.4.

5.4 Keyword detection

The evaluation of the keyword detection was as previously mentioned performed on transcriptions generated by the wav2vec 2.0 models. The dataset used for this was the categorised recordings, presented in Section 3.1.1, with a 75/25 split for the tuning and testing sets respectively. This set contains examples from all the classes specified in Section 3.1. As the objective is to notify the operator if it is an interesting message or not, a binary classification task was needed. The classes for emergencies and JRCC calls were grouped together and all other classes were grouped together to create two classes resulting in a binary classification. The words used by the detector is shown in the table below and are all connected to emergencies or the JRCC.

Keywords
mayday
pan
jrcc
rescue
sjöräddning
sjöräddningen
coastguard
SOS
distress
help
hjälp
sjönöd

This list of phrases contains words in both Swedish and English, as in the context of keyword detection this does not matter, as words will not be detected by mistake in the other language. There are also many words that is used even in Swedish language even if they are English words from the start.

Model	**KD**	valid	
		Acc	F
Swedish			
XLSR-SV	Ham	65.38	0
	Lev	65.38	0
	DL	65.38	0
	LCSS	65.38	0
	MRA	71.34	0.35
XLSR-53	Ham	75.0	0.69
	Lev	88.46	0.83
	DL	86.54	0.81
	LCSS	80.77	0.76
	MRA	73.07	0.68

Continued on next page

Table 5.8 – *Continued from previous page*

Model	**KD**	valid	
		Acc	F
LARGE_SV	Ham	71.15	0.48
	Lev	76.92	0.53
	DL	76.92	0.53
	LCSS	65.38	0
	MRA	65.38	0
LARGE_VOX	Ham	90.38	0.86
	Lev	88.46	0.83
	DL	90.38	0.86
	LCSS	82.69	0.78
	MRA	88.46	0.83
English			
LARGE_960h	Ham	93.61	0.4
	Lev	93.61	0.4
	DL	93.61	0.4
	LCSS	93.61	0.4
	MRA	91.49	0.33
LARGE_VOX	Ham	95.74	0.75
	Lev	93.62	0.57
	DL	93.62	0.66
	LCSS	95.74	0.75
	MRA	100.0	1.0
LARGE	Ham	100.0	1.0
	Lev	100.0	1.0
	DL	95.74	0.83
	LCSS	97.87	0.91
	MRA	100.0	1.0
Swedish-English			
XLSR-53	Ham	74.49	0
	Lev	74.49	0
	DL	74.49	0
	LCSS	74.49	0
	MRA	74.49	0

Table 5.8: The results for the text similarity measures for all models used in this thesis.

5.4.1 Word similarity algorithms

One interesting measure is the ROC curve. It shows the trade-off between True positive rate and False positive rate. The ROC plot shown in Figure 5.3 contains

the ROC curves for all five algorithms. The resulting curves show that Levenshtein and Damerau-Levenstein perform exactly the same and that due to the high true positive rate earlier on the curve flattens out quickly. The curves is also very choppy as there was such a small amount of data.

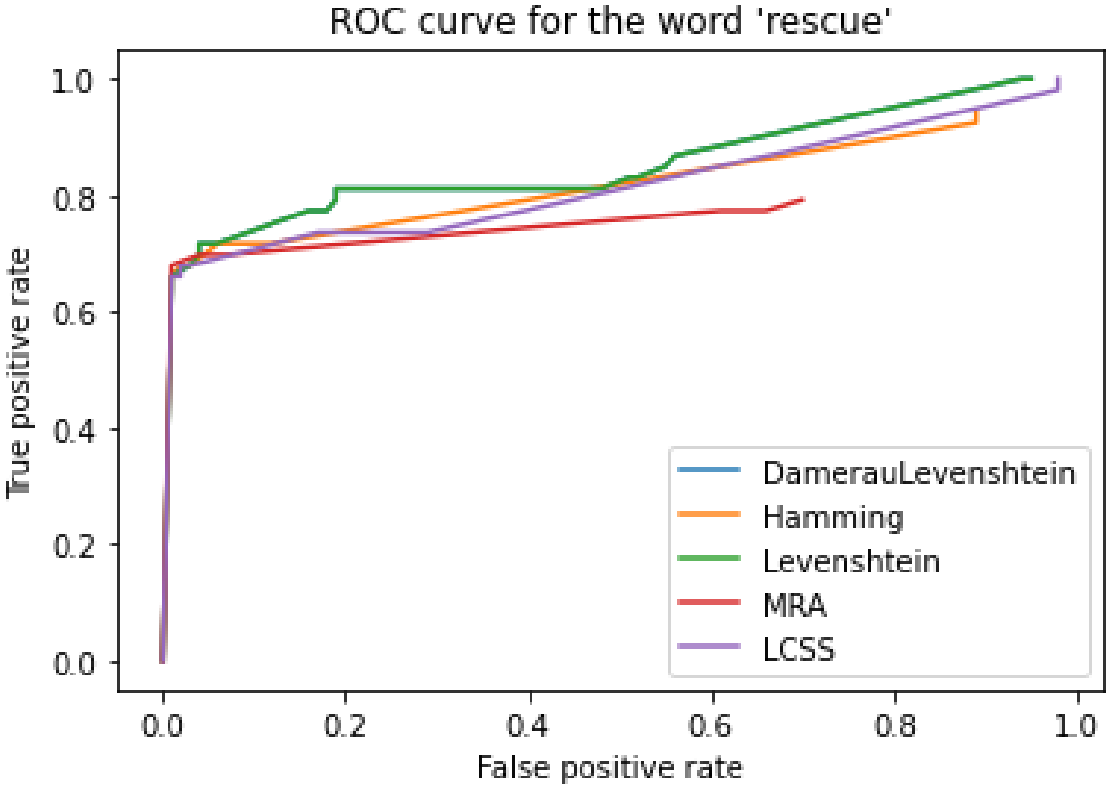


Figure 5.3: ROC curves for the five word similarity algorithms for the word "rescue", performed on the Swedish sample set using the LARGE_VOX model.

To allow for a comparison between the five algorithms, the similarity between the transcribed word "KUSTIEVAKNINGEN" and the correctly spelled "KUSTBEVAKNINGEN" is computed using each method. This shows how the different methods evaluate a a simple spelling mistake that is common in the transcriptions.

5.4.1.1 Hamming distance

The Hamming distance is a classic distance and that shows in the results as it performs in the middle of all the other algorithms, as it is based on one very simple rules it does not account for letter switching for example. But the transcriptions produced by the wav2vec 2.0 models usually do not output any letter switches as can be seen in Section 5.3 so a simple algorithm such as the Hamming distance performs well as a baseline. For the example described above the normalised Hamming distance was 0.93. This resulted in one wrong letter in 15 letters $\frac{15-1}{15} = 0.93$.

5.4.1.2 Levenshtein distance

As the Levenshtein distance is an extension of the Hamming distance it can be seen above that it performs well on most models and is a clear upgrade over the Hamming distance on average but there are cases where it performs worse like for the Swedish VOX model. For the example explained above the Levenstein distance gets the same distance as it is just one replaced letter in the word and no insertions or deletions.

5.4.1.3 Damerau–Levenshtein

The D-L algorithm is an overall well-performing algorithm for this kind of text distance calculation and performs well in the classification task. As it can handle letter switching it is usually a better performing algorithm than Levenshtein but for this use case where letter switching is unusual it performs on pare for most models. As the example only have one replaced letter also Damerau-Levenshtein get a distance of 0.93.

5.4.1.4 Longest Common Substring Similarity

LCSS is the overall worse performer on the classification task at hand. The data shows that the models output generally does not have errors just in the beginning and end but spread over the entire word and such get low similar substrings. When LCSS is used on the two word example above a distance of 0.67 is given. This as the longest common substring is 10 characters long $10/15 = 0.67$

5.4.2 Match Rating Approach

MRA yields interesting results. As it is a very specialised algorithm its performance differs a lot between samples. As MRA is a rule-based algorithm its rules are created for human mishearings. The STT models have the same problems. When the MRA does work it performs well with 100% accuracy at some samples. When using MRA on the example both words get encoded using the rules described in Section 2.8.4 into "KSTNGN" so that the distance is 1 between the words. This is both the strength and the weakness of MRA.

6

Discussion

In this chapter, some key parts of the thesis are highlighted and discussed. It is formatted in such a way that it relates to the method where appropriate. Each section can be seen as an interesting takeaway or an especially important part of the thesis.

6.1 The initial hypothesis and the issues it caused

During the initial phase of the thesis, there were a lot of uncertainties regarding access to data. This meant that none of the relevant data were available and no guarantees for its delivery could be given. Access to the JRCC data was then secured at the halfway point and the project could continue with the intended workflow. The uncertainty, in the beginning, led to some imposed decisions regarding STT effectiveness and overall layout of the thesis.

The biggest hurdle at the beginning of the thesis was the inherited hypothesis from the project introduction, where the use of SE would allow the STT to produce meaningful results. Had steps been taken to validate or at least look at the actual data, this might have been revised at an earlier stage, saving both time and effort. Instead, the hypothesis was used as a source of truth and the thesis immediately started focusing on the peculiarities of advanced SE. This partly depended on the lack of data, which meant that such verification would have taken both time and effort from multiple parties. The issue could have been alleviated with some more care and critical thinking, a notion that should be apparent throughout the rest of the thesis.

6.2 Speech Enhancement and the STT service

Following the poor results presented in Section 5.2 the idea of improving the data sent to the Google API was abandoned and this section aims to highlight some of the reasons hidden behind the results. First and foremost, since it is a closed system the only way to improve the transcription performance is to change the configuration. This means that there is no way to fundamentally impact the outcome of a transcription attempt apart from a few parameters. The output has to be good

enough, as there is no further insight into how to improve it afterward. Through the limited testing, the resulting output simply was not good enough, despite various configuration changes, and thus the approach was not deemed viable.

6.2.1 Abandoning the Google STT rationale

The thinking behind abandoning the plan to improve audio quality so quickly was mainly based on the lack of control, there was simply no way to impact the outcome. The other thing that played a big part in the decision was the manner in which the API transcribed, where the best example of this is probably the transcription presented in Table 5.5. Here the good WER leads one to believe it is good, but it is a bit deceiving.

Configuration	Transcription
Reference	all stations all stations all stations information concerning the port of visby information concerning the port of visby the ferry visby will enter the port within ten minutes
Speech Context English	all stations all stations all stations information concerning the port of little information concerning the port of the state will enter the port within 10 minutes

Comparing the two one can see that the key pieces of information about the port and the subject, the ferry, are not present in the transcription. The issue is thought to be a tightly integrated Language Model, since the probability of both “the port of visby” and “the ferry visby” to appear in a "regular" text corpora is very slim. It is next to impossible to verify this but the amount of locations and lack of typical language structure seems to limit the performance of the model. The LM makes it very hard to trust the results, something that has been seen again with the N-grams used in the domain-specific STT model.

6.2.2 Speech Enhancement

The many researchers in the SE field have presented several interesting algorithms recently, where SEGAN had shown the most promise for the domain. In this thesis, a proof of concept for the SEGAN was produced but it did not achieve the same performance as the one presented in the paper [22]. This kind of SE algorithm is interesting because of how it uses data. As it is based on GANs it is not reliant on parallel data with clean and noisy sample tracks. It can also learn to filter out any kind of noise, which is good for the more complex noise present in the JRCC data used in this thesis, compared to more cyclic humming that can be easily filtered out by a Fourier transform.

The results presented in Section 5.1 indicate that the technique might be suited for the purpose of this thesis, but it was not pursued further due to the severe time limitation caused by the switch to STT development. To make any SE algorithm

work with the STT model, the model needs to be trained on data passed through the SE algorithm. This is something that could have been done if the time had been available, but as is discussed in Section 7.1.1 this is a promising technology and an area of future possibilities

6.3 The domain-specific STT

When the results of the fine-tuning started to come in the initial feeling was good, as the results were way above the expectations set by the previous attempt using the Google API. The numbers are not state-of-the-art in any way, shape, or form but they seem to be very competitive for the domain, and with the use of robust keyword detection the system performance appears to be good enough for the task at hand.

When evaluating the performance of the different models one quickly noticed that the pre-training played a big part, which is both a good and a bad thing. The fine-tuning of models is, in this context, relatively manageable computation-wise, whereas proper pre-training is both time-consuming and expensive due to the computational requirements. This means that most applications will have to rely on openly available models pre-trained on suitable data. In the case of this thesis, it was shown that such models can perform well, but if the know-how and resources would have been available there is no doubt that the results could have been improved drastically with domain-specific pre-training, rather than the simpler transfer learning discussed in Section 6.3.1.

Then there is the letter-based versus word or word-piece argument, commonly found in STT discussions, which is highly relevant for the thesis domain but also for the STT domain at large. It is mentioned in the paper by Baevski *et al.* [9] that the performance would likely be increased if the model was adapted to work on word-pieces rather than letters. This is tied to the fact that their best performance number was achieved using Language Models and that the defining performance metric in the space is WER, but as have been seen in the results here the use of small LMs introduce a hazard of its own. This is especially true for the use case of this domain, where the complete loss of context is outright dangerous. More on this is Section 6.3.2.

6.3.1 The outcome of the JRCC transfer learning

After performing the fine-tuning on the model that had been pre-trained using the JRCC data it was quickly discovered that the transfer learning had ruined its transcription capabilities. This was made evident by the performance of the large model without the transfer learning. It is very hard to pinpoint exactly why the exposure to the new data impacted the performance to such an extent, especially when comparing the number of updates, 7530 to 600 000 [9]. Due to both the time and cost aspect, it was not possible to try any possible remedies, but if the opportunity to do something similar would present itself the first course of action

would be to curate the data. This should ensure that the pre-training would only be performed on recordings of quality audible to the trained operators and the plethora of empty clips would be removed. This curated dataset should then be applied in a slightly different manner than it was in the thesis, using more advanced learning rate settings to better extend the initial learnings as seen in newly released research[1] [79].

The late addition of the Swedish VoxPopuli subset shed some new light on these results, indicating that the quality might not be the only reason. The results on the English dataset indicate that the transfer learning was not successful, but even if performed correctly the new results indicate that the results might not improve significantly for the Swedish dataset. For that case a new base model would probably be needed, where the performance of the larger VoxPopuli shows that the model benefits hugely from the bigger and more diverse dataset when applied to the JRCC recordings.

6.3.2 The Discrepancy of the Language Models

The experiment using LMs clearly showed the issues with the models built on small corpora; they did in many cases improve the WER, but in doing so they often lost the context with increased CER. Language Models are still likely the best way to improve the performance of the models, but to do so they need to incorporate a lot more information than the small corpora allowed. This is especially relevant for the naval domain, as the skewed nature of data means that the occurrence of a small sample of sentences, the broadcast messages, might make up the majority of the corpus and thus be overly probable. This is the case with the N-gram models. Section 7.1.3 goes into more detail on what the next step would be to increase the benefits of the LM.

The behavior observed by the small N-grams used here showed some similarity to the results from Google's STT API, where the believed influence from a Language Model made the transcriptions better in a language sense, but completely lost the context and connection to the original. Since the Google service is lacking implementation details it is impossible to know any specifics, but it is very easy to believe something in this direction, by observing the similar behaviors.

6.3.3 Swedish data for Swedish transcriptions

The results of the smaller VoxPopuli model highlighted the fact that the language in the Swedish JRCC recordings are composed in such a way that the model only familiar with Swedish had immense trouble producing any meaningful transcriptions. This was surprising, as the model was thought to be very specialized and thus should perform well on, at least, the Swedish parts. This was as could be seen in the results, not the case. The model could not manage any decent transcriptions, indicating that the domain requires more robustness and a bigger pallet of speech representations

[1] `https://huggingface.co/KBLab/wav2vec2-large-xlsr-53-swedish`

to produce meaningful results. This was at least the notion received by the larger model's results, which were astonishing considering the results of the purely Swedish model.

6.3.4 The impact of poor audio quality

Due to the outcome of the JRCC transfer learning, none of the performing models have been exposed to a large number of noisy recordings. There are a few in the fine-tuning datasets but the number is limited by the fact that every utterance needs to be audible, as the implementation currently does not handle missing words. This makes the results in Section 5.3.3 far more impressive than the numbers suggest and shows that the selected approach can handle some levels of noise. Further tests are needed to evaluate the impact a successful transfer learning with moderately noisy data could have, but the initial results are more than promising. In Section 7.1.1.1 of the Future work, a suggestion for a wav2vec 2.0 fine-tuning that allows missing words is detailed.

6.4 Keyword detection for intent classification

As shown in Section 5.4 the keyword classification produces good results for individual keywords, but the more interesting thing is the system-wide performance. The keyword detection is evaluated as a system rather than individually, fully dependent on the models trained and presented in Section 5.3. Looking at the results for the full system, results of above 90% can be seen across the board with the outlier of the English large model being validated to an accuracy of 100%. These results in both languages were deemed so good as to be able to be beta tested by the JRCC.

One important difference in this thesis compared to other STT papers is the main metric for evaluating the proficiency of the system. This thesis uses precision-recall and accuracy to evaluate the system. This is to fulfill the overall goal of the thesis to find the messages in need of attention as this is the crucial metric for project success. This differs from the purely research-based approaches where WER is the key metric for these kinds of systems. However, while WER produces good results this metric alone does not guarantee the creation of understandable, context preserving, messages. Which is why this was not used as the defining performance metric.

The threshold tuning explained in Section 4.4 that was performed is a part that was done to maximise accuracy. This part is something that may need to be reworked for real-world testing. In a safety-centric system, like the use-case here it is really important not to miss any incoming messages. Thus the sensitivity of the system should be tuned not to miss anything, rather than having better accuracy. As the system is implemented, a lower sensitivity will also increase the false positives. This could be an issue for an operator as people easily get annoyed by too many false warnings.

This has been proven time and time again in papers such as "The continuing problem of false positives in repeated measures ANOVA in psychophysiology: A multivariate solution" [80]. So for the final deployment, a manual fine-tuning to the threshold is needed to minimize missed messages and at the same time not have too many false positives.

Keyword detection was chosen for this thesis due to the time limitation, but the overall goal of the project is intent analysis. There exist more advanced methods such as BERT [12]. These more advanced methods yield on average higher scores than our accuracy for intent analyses, as high as 97% [81], and this is something that will be discussed further in Section 7.1.2.

6.4.1 Keyword detection algorithms

For this thesis, five algorithms were chosen. These are three character-based ones, one substring-based one, and one phonetic rule-based one. As the models used in this thesis produces transcriptions that are letter based it was found that the three character-based algorithms performed better overall. The interesting finding is the MRA algorithm that was developed for phonetic errors in written text did not perform as well as may be expected. Because it is based on surnames which does have problem with longer words. It also struggled as the phonetic problems experienced by the model is not the same as the problems facing Western airlines when they developed MRA.

6.5 Thoughts, ramblings and remarks

While working with the different parts of the thesis a few things stood out and deserve increased focus to ensure the transfer of knowledge.

6.5.1 Working with raw data

The data did not just have availability issues, it also presented a challenge by lacking any form of annotation. This meant that considerable time had to be spent to generate both the transcriptions and the simpler categorisations. The consequence of this need for manual annotation is a lack of quantity in the experiments, limited possible solutions, and lessened scientific significance of the results. It is also very apparent that many of the results presented would have been improved with more data, especially the wav2vec 2.0 models.

6.5.2 The rise of self-supervised STT implementations

The positive outcome of the thesis relies heavily on the recent developments in self-supervised STT models, where the advanced techniques developed by Baevski *et al.* [9] allowed for the creation of a domain-specific system without spending months on creating transcriptions. These techniques are still developing, and new models are released continuously which should enable even more suited base-models

for the different use-cases. There have even been successful implementations of this using self-training by Xu *et al.* [21], indicating that the research topic is still highly explorative. These advancements show incredible promise for the future of STT systems, especially highly specialised ones in narrow domains such as this one. The possible performance of this approach is highlighted by the fact that the top-performing models for speech recognition are based on wav2vec 2.0, as can be seen on the popular leader-board at Papers with Code[2], where the implementation by Zhang *et al.* [82] were the top performer at the time of writing.

6.5.3 The growing discrepancy inability to build solutions

Something encountered during the thesis was the plethora of new, large machine learning models available both as pre-trained and just as structures. Many of these models had achieved astonishing results in their respective fields, but also highlight a trend that is not as positive. All these models require immense hardware resources to produce the results they have achieved, and figures like 128 powerful GPUs running non-stop for a week are not uncommon or even seen as something special anymore. This points to a gorge in the machine learning community, where these models often invite individuals and corporations with less data processing power but at the same time only enable those with big computation funds to take full advantage. Thankfully some of these models are made available as pre-trained versions, allowing for the less compute-heavy extension of models, but there is a case to be made for the discrepancies this trend is causing.

6.6 Ethical considerations

The topic of ethics should always be given proper thought when developing machine learning systems since the failure to do so could cause severe ramifications with far-reaching consequences. This is especially the case for the thesis domain, as the system is supposed to be used in critical situations, and could be devastating without a proper understanding of the results.

6.6.1 Sensitive data

As every data-based problem with ties to individuals may contain sensitive information, one has to be aware of the impact misuse of such data can have, in order to protect privacy. Since some of the data used in the thesis is not openly available, but instead supplied by the JRCC, this places some extra responsibility on ensuring proper data security.

The circumstance in which these transmissions are sent also play a part in the ethical aspects of managing the data. The recordings contain instances of people in a rather sensitive state, requiring substantial effort to ensure its safekeeping to protect the integrity of said individuals.

[2] `https://paperswithcode.com/task/speech-recognition`

When looking at the broader aspects of the data there exist additional security concerns. This data can be stored by anyone at the time of recording, but due to the nature of the system, it requires immense effort to capture every message. This has created a dataset that contains information about most of the major incidents in and around Swedish waters, and such information is by nature a matter of national security.

6.6.2 Common oversights

In the domain of machine learning there exists a plethora of ethical questions that should always be considered, to ensure that learnings from previous research and deployments are accounted for. Especially since the aspect is easily lost among the technical details and research aspects.

6.6.2.1 Bias

The potential for bias in the created system is highly dependent on the supplied dataset, which in and of itself presents a few challenges. Examples of bias that might be induced from the data are classification differences between genders, naval communication proficiency, and equipment quality. Another potential bias that might be induced from the data is the discrepancy between languages, where some languages represent such a small part of all traffic that they might simply get overlooked by the system. Most of these potential issues are very hard to find a remedy for since the domain is very specific and there exists a lack of research around some of the hassles. For this reason, steps were taken to be aware of as many of the potential issues as possible, so that these are known and documented to enable corrections as the system evolves.

6.6.2.2 Reliability and responsibility

When introducing systems to replace or aid humans there always exists a risk of creating a false sense of security, should the system start to be relied upon. If this would happen, it could result in a move away from audio monitoring to just observing the system output. The switch from augmentation to the main system could lead to faulty handling of emergencies, entirely missed calls for help and as a consequence of this, loss of life. Care is always needed to ensure the correct use of deployed systems, to avoid accidents caused by system reliance. This is an issue for a lot of machine learning systems, as it is very hard to tell exactly how such systems will react to specific scenarios compared to more traditional "mechanical" systems which generally provide a much better understanding of their inner workings. In the JRCC case, the simple mitigation is to ensure the use of the system as augmentation, not as a replacement for traditional monitoring.

6.6.3 Open-source

A trend in software development for many years have been the use of open-source and during the last years the same open-source trend have gained traction in the

field of machine learning. For a long time the artificial intelligence industry have been sharing their papers and findings publicly, but the same could not be said of the code and trained models. This is now an evolving landscape where more and more pre-trained models make their way out to open-source. This can cause concerns for the users of such software and models. Many of the ethical considerations regarding the use of open-source pre-trained models are the same as with all AI systems but with no ability to find and correct them in the trained weights. Another big issue with open-source models in the accountability of the user. For a regular piece of software that can be tested and proven you can prove the correctness of the program. The problem with machine learning models is the lack of explainability and provability. This means that there is no way to prove the model. As the models usually are distributed under a "as is" license, this means that it is up to the user to be accountable for the models use. However, there is no way to prove the software thus creating a gap between the promise of performance and the accountability.

7

Conclusion

This chapter summarises the key take-aways, learnings, and conclusions from the thesis as well as suggesting some future work in Section 7.1.

This thesis has shown that with the use of self-supervised STT architectures one can reach sufficient performance in settings where the domain prohibits the successful application of ready-made systems. The achieved level of performance enables the use of both simpler forms of intent analysis, as has been shown with the keyword detection, but should also enable more advanced intent analysis. All this can be achieved with limited amounts of annotated data and without the need for a costly annotation process. It has also been shown that the self-supervised approach can be implemented with limited hardware thanks to the availability of pre-trained models, where most of the heavy lifting has already been done. The thesis has also proved the enormous power that is accessible through open-source, with projects such as wav2vec 2.0, and that these can compete with proprietary state-of-the-art systems.

The project has also validated the use of STTs as an integral part in more complex domain-specific applications, that enable the continued exploration of highly specialized Language Models and the use of similar systems to extend the functionality. This includes advanced information extraction and augmentation, not just in the naval VHF environment, but also in other domains with similar challenges.

To summarise the outcome and learnings from the development of a domain-specific STT model, one can just express the importance of the strive to keep results open-source. Without this, none of the results presented here would have been possible. The rise of the self-supervised STT models means that qualitative transcription systems can be utilised in more areas and thus the impact will be far greater than what was previously allowed through the use of proprietary and data-heavy solutions.

7.1 Future work

Below are some topics the authors would have liked to pursue should there have been more time. Instead they are left as suggestions for further research.

7.1.1 wav2vec 2.0 improvements

With the proper hardware allocation, it would be beneficial to attempt to pre-train a wav2vec 2.0 model on the VHF recordings. Steps would have to be taken to increase the quality of the data and ensure that the recordings used are audible to seasoned rescue leaders or removed otherwise. The possibilities of such a model would be immensely useful to the JRCC. It would also shed some light on the impact on the use of completely domain-specific wav2vec 2.0 implementations rather than the domain-tuned ones.

7.1.1.1 Handle missing words in the fine-tuning

One of the biggest reasons for the lack of transcriptions was the exclusion of messages that contained un-hearable or missing words since wav2vec 2.0 requires complete transcriptions during the fine-tuning. It would be very interesting to see the impact of changes to the fine-tuning procedure, that would allow for the use of transcriptions with missing pieces, as this would yield more usable transcriptions and should allow the model to be exposed to more noisy recordings.

7.1.1.2 Speech Enhancement

As has been shown in this thesis, SE is an area of research with impressive performance and the huge potential that should be investigated further and in combination with the more specialized STT models. The results shown by advanced machine learning-based SE such as SEGAN shows what these technologies can accomplish. The implementation of a wav2vec 2.0 model trained on SEGAN processed recordings could produce very interesting results and push the boundaries of combined SE and STT forward.

7.1.2 Advanced intent analysis

The sole reason for the use of keyword detection for intent analysis was the lack of time, because of the apparent limitations of the approach. The use of proper intent analysis could greatly improve the performance and robustness of such a system, and the first approach to experiment with given more time would have been the application of the BERT model [12]. The use of BERT's classification ability could allow the system to detect poorly formulated emergency requests and might even handle the less successful transcriptions. It might however require even more annotated data and higher quality transcriptions to work properly.

7.1.3 The future of the Language Model

The key to the excellent performance of the ASR systems in many of the research papers on the topic is the Language Models [75, 9, 11]. The LMs in this thesis are built the same way as the ones used by Baevski *et al.* [9], but were way too small to have a positive effect. The biggest possible performance gain for the evaluated models is thus probably the creation of an effective Language Model. The key to this is the corpus used for building it, which is why the creation of a proper naval corpus should serve as the basis for any further development. This should allow for a lot more information to be incorporated into whatever LM is created, and significant effort in creating such a corpus is likely to be rewarded with excellent performance. The other two approaches are the use of a larger Transformer LM, but this is highly connected to the availability of a larger corpus, and the use of an external LM, that is a Language Model trained of data not tied to the domain. This should enable insights into how the transcription is affected by regular language knowledge.

7.1.4 Avoiding the transcriptions all together

Another approach that was considered during the project was the implementation of emergency classification directly on the recording, skipping the transcription step. This could be done directly on the recordings using an Recurrent Neural Network or other network with similar properties. A completely different approach could use something like the audio version of Word2Vec to create embeddings and then do the classification on these fixed-length vectors [83]. The result of such an attempt would be highly interesting to see as the development does not require any transcription work. The main difficulty with such an approach is the skewed data and general lack of interesting messages, which provide a whole set of challenges.

Bibliography

[1] Sjöfartsverket, "Joint rescue and co-ordination centre." https://www.sjofartsverket.se/en/Maritime-services/Search-and-Rescue/.

[2] Sjöfartsverket, "Statistik sjö- och flygräddning 2020." https://www.sjofartsverket.se/contentassets/b6149761283749728ffa0b2f8f149c62/arsstatistik-sjo--och-flygraddning-2020.pdf.

[3] Transportstyrelsen, "Maritime assistance service." https://www.transportstyrelsen.se/en/shipping/Accidents--near-misses/Maritime-Assistance-Service-MAS/.

[4] Sjöfartsverket, "Svensk kustradio." https://www.sjofartsverket.se/sv/Batliv/Svensk-kustradio/.

[5] K. Röshammar, "pers.comm."

[6] E. Brask and E. Tingström, "Lättare att uppfatta nödrop till sjöss," *Sjörapporten*, no. 1, p. 10, 2021.

[7] M. Alam, M. D. Samad, L. Vidyaratne, A. Glandon, and K. M. Iftekharuddin, "Survey on deep neural networks in speech and vision systems," *Neurocomputing*, vol. 417, pp. 302–321, 2020.

[8] A. Nassif, I. Shahin, I. Attili, M. Azzeh, and K. Shaalan, "Speech recognition using deep neural networks: A systematic review," *IEEE Access*, vol. PP, pp. 1–1, 02 2019.

[9] A. Baevski, H. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," 2020.

[10] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," 2018.

[11] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding by generative pre-training," *OpenAI*, 2018.

[12] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of

deep bidirectional transformers for language understanding," 2019.

[13] O. J. Hénaff, A. Srinivas, J. D. Fauw, A. Razavi, C. Doersch, S. M. A. Eslami, and A. van den Oord, "Data-efficient image recognition with contrastive predictive coding," 2020.

[14] P. Bachman, R. D. Hjelm, and W. Buchwalter, "Learning representations by maximizing mutual information across views," 2019.

[15] I. Misra and L. van der Maaten, "Self-supervised learning of pretext-invariant representations," 2019.

[16] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," 2020.

[17] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," 2020.

[18] S. Schneider, A. Baevski, R. Collobert, and M. Auli, "wav2vec: Unsupervised pre-training for speech recognition," 2019.

[19] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an asr corpus based on public domain audio books," in *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pp. 5206–5210, IEEE, 2015.

[20] A. Conneau, A. Baevski, R. Collobert, A. Mohamed, and M. Auli, "Unsupervised cross-lingual representation learning for speech recognition," 2020.

[21] Q. Xu, A. Baevski, T. Likhomanenko, P. Tomasello, A. Conneau, R. Collobert, G. Synnaeve, and M. Auli, "Self-training and pre-training are complementary for speech recognition," 2020.

[22] S. Pascual, A. Bonafonte, and J. Serra, "Segan: Speech enhancement generative adversarial network," *arXiv preprint arXiv:1703.09452*, 2017.

[23] Y. Luo and N. Mesgarani, "Conv-tasnet: Surpassing ideal time–frequency magnitude masking for speech separation," *IEEE/ACM transactions on audio, speech, and language processing*, vol. 27, no. 8, pp. 1256–1266, 2019.

[24] S. Douglas, *Audio Engineering Explained*, pp. 200, 446. Taylor & Francis US, 2012.

[25] IBM Corporation and Microsoft Corporation, "Multimedia programming interface and data specifications 1.0," 1991. `https://www.aelius.com/njh/wavemetatools/doc/riffmci.pdf`.

[26] K. J. Piczak, "Environmental sound classification with convolutional neural

networks," in *2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)*, pp. 1–6, IEEE, 2015.

[27] R. J. R. J. Baken, *Clinical measurement of speech and voice*. San Diego: Singular Thomson Learning, 2nd ed.. ed., 2000.

[28] S. S. Stevens, J. Volkmann, and E. B. Newman, "A scale for the measurement of the psychological magnitude pitch," *The Journal of the Acoustical Society of America*, vol. 8, no. 3, pp. 185–190, 1937.

[29] D. of homeland security coast guard, "Radio information for boaters." `https://www.navcen.uscg.gov/?pageName=mtBoater`.

[30] Sjöfartsverket, "Vhf-radio." `https://www.sjofartsverket.se/Batliv/VHF-radio/`.

[31] J. Ralston, J. Heagy, and R. Sullivan, "Environmental/noise effects on vhf/uhf uwb sar," *INSTITUTE FOR DEFENSE ANALYSES*, p. 24, 09 1998.

[32] U. C. Guard, "Mv summit venture mayday call." Wikimedia, 5 1980.

[33] K. Zechner and A. Waibel, "Minimizing word error rate in textual summaries of spoken language," in *Proceedings of the 1st North American Chapter of the Association for Computational Linguistics Conference*, NAACL 2000, (USA), p. 186–193, Association for Computational Linguistics, 2000.

[34] R. Li, X. Sun, Y. Liu, D. Yang, and L. Dong, "Multi-resolution auditory cepstral coefficient and adaptive mask for speech enhancement with deep neural network," *EURASIP Journal on Advances in Signal Processing*, vol. 2019, 04 2019.

[35] P. Mermelstein, "Evaluation of a segmental snr measure as an indicator of the quality of adpcm coded speech," *The Journal of the Acoustical Society of America*, vol. 66, no. 6, pp. 1664–1667, 1979.

[36] I. T. Union, "Methods for subjective determination of transmission quality," recommendation, International Telecommunications Union, Aug. 1996.

[37] G. Forman *et al.*, "An extensive empirical study of feature selection metrics for text classification.," *J. Mach. Learn. Res.*, vol. 3, no. Mar, pp. 1289–1305, 2003.

[38] F. Rosenblatt, "Principles of neurodynamics. perceptrons and the theory of brain mechanisms," *American Journal of Psychology*, vol. 76, p. 705, 1963.

[39] K. O'Shea and R. Nash, "An introduction to convolutional neural networks," *arXiv preprint arXiv:1511.08458*, 2015.

[40] A. Sherstinsky, "Fundamentals of recurrent neural network (rnn) and long

short-term memory (lstm) network," *Physica D: Nonlinear Phenomena*, vol. 404, p. 132306, 2020.

[41] Z. Khabazi, "Generative algorithms," *Accessed on*, vol. 3, 2011.

[42] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *arXiv preprint arXiv:1406.2661*, 2014.

[43] M. Bińkowski, J. Donahue, S. Dieleman, A. Clark, E. Elsen, N. Casagrande, L. C. Cobo, and K. Simonyan, "High fidelity speech synthesis with adversarial networks," *arXiv preprint arXiv:1909.11646*, 2019.

[44] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, and A. A. Bharath, "Generative adversarial networks: An overview," *IEEE Signal Processing Magazine*, vol. 35, no. 1, pp. 53–65, 2018.

[45] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[46] L. Theis, W. Shi, A. Cunningham, and F. Huszár, "Lossy image compression with compressive autoencoders," *arXiv preprint arXiv:1703.00395*, 2017.

[47] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2017.

[48] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," 2014.

[49] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015.

[50] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," 2016.

[51] O. Press and L. Wolf, "Using the output embedding to improve language models," 2017.

[52] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[53] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826, 2016.

[54] J. Seppänen, "Audio signal processing basics." `https://www.cs.tut.fi/sgn/arg/intro/basics.html`.

[55] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, pp. 1026–1034, 2015.

[56] D. Hendrycks and K. Gimpel, "Gaussian error linear units (gelus)," 2020.

[57] A. Mohamed, D. Okhonko, and L. Zettlemoyer, "Transformers with convolutional context for asr," 2020.

[58] S. Dieleman, A. van den Oord, and K. Simonyan, "The challenge of realistic music generation: modelling raw audio at scale," 2018.

[59] E. Jang, S. Gu, and B. Poole, "Categorical reparameterization with gumbel-softmax," 2017.

[60] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, (New York, NY, USA), p. 369–376, Association for Computing Machinery, 2006.

[61] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "Specaugment: A simple data augmentation method for automatic speech recognition," *Interspeech 2019*, Sep 2019.

[62] D. Jurafsky and J. H. Martin, "Speech and language processing." 3rd ed. draft, 2021.

[63] R. Kneser and H. Ney, "Improved backing-off for m-gram language modeling," in *1995 International Conference on Acoustics, Speech, and Signal Processing*, vol. 1, pp. 181–184 vol.1, 1995.

[64] A. Bookstein, V. A. Kulyukin, and T. Raita, "Generalized hamming distance," *Information Retrieval*, vol. 5, no. 4, pp. 353–375, 2002.

[65] G. Navarro, "A guided tour to approximate string matching," *ACM computing surveys (CSUR)*, vol. 33, no. 1, pp. 31–88, 2001.

[66] E. Brill and R. C. Moore, "An improved error model for noisy channel spelling correction," in *Proceedings of the 38th annual meeting of the association for computational linguistics*, pp. 286–293, 2000.

[67] D. Gusfield, "Algorithms on stings, trees, and sequences: Computer science and computational biology," *Acm Sigact News*, vol. 28, no. 4, pp. 41–60, 1997.

[68] G. B. Moore, *Accessing individual records from personal data files using non-unique identifiers*, vol. 13. US Department of Commerce, National Bureau of Standards, 1977.

[69] L. Gwilliams, T. Linzen, D. Poeppel, and A. Marantz, "In spoken word recognition, the future predicts the past," *Journal of Neuroscience*, vol. 38, no. 35, pp. 7585–7599, 2018.

[70] C. Valentini-Botinhao *et al.*, "Noisy speech database for training speech enhancement algorithms and tts models," *Centre for Speech Technology*, 2017.

[71] H. Phan, H. L. Nguyen, O. Y. Chén, P. Koch, N. Q. K. Duong, I. McLoughlin, and A. Mertins, "Self-attention generative adversarial network for speech enhancement," 2021.

[72] C. Donahue, B. Li, and R. Prabhavalkar, "Exploring speech enhancement with generative adversarial networks for robust speech recognition," in *Proc. ICASSP*, 2018.

[73] H. Phan, I. V. McLoughlin, L. Pham, O. Y. Chen, P. Koch, M. De Vos, and A. Mertins, "Improving gans for speech enhancement," *IEEE Signal Processing Letters*, vol. 27, p. 1700–1704, 2020.

[74] M. Ott, S. Edunov, A. Baevski, A. Fan, S. Gross, N. Ng, D. Grangier, and M. Auli, "fairseq: A fast, extensible toolkit for sequence modeling," in *Proceedings of NAACL-HLT 2019: Demonstrations*, 2019.

[75] A. Håkansson and K. Hoogendijk, "Transfer learning for domain specific automatic speech recognition in swedish: An end-to-end approach using mozilla's deepspeech," Master's thesis, Chalmers University of Technology, 2020.

[76] C. Wang, M. Rivière, A. Lee, A. Wu, C. Talnikar, D. Haziza, M. Williamson, J. Pino, and E. Dupoux, "Voxpopuli: A large-scale multilingual speech corpus for representation learning, semi-supervised learning and interpretation," 2021.

[77] V. Pratap, A. Hannun, Q. Xu, J. Cai, J. Kahn, G. Synnaeve, V. Liptchinsky, and R. Collobert, "Wav2letter++: A fast open-source speech recognition system," *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2019.

[78] G. Synnaeve, Q. Xu, J. Kahn, T. Likhomanenko, E. Grave, V. Pratap, A. Sriram, V. Liptchinsky, and R. Collobert, "End-to-end asr: from supervised to semi-supervised learning with modern architectures," 2020.

[79] M. Malmsten, "pers.comm."

[80] M. W. Vasey and J. F. Thayer, "The continuing problem of false positives in repeated measures anova in psychophysiology: A multivariate solution," *Psychophysiology*, vol. 24, no. 4, pp. 479–486, 1987.

[81] H. Jiang, P. He, W. Chen, X. Liu, J. Gao, and T. Zhao, "Smart: Robust and

efficient fine-tuning for pre-trained natural language models through principled regularized optimization," *arXiv preprint arXiv:1911.03437*, 2019.

[82] Y. Zhang, J. Qin, D. S. Park, W. Han, C.-C. Chiu, R. Pang, Q. V. Le, and Y. Wu, "Pushing the limits of semi-supervised learning for automatic speech recognition," 2020.

[83] Y.-A. Chung, C.-C. Wu, C.-H. Shen, H.-Y. Lee, and L.-S. Lee, "Audio word2vec: Unsupervised learning of audio segment representations using sequence-to-sequence autoencoder," 2016.

A

Compute resources

This appendix lists all the different hardware configurations used during the thesis.

A.1 Google

A.1.1 Compute Engine Config

Series: N1

Machine type: n1-standard-32

CPU: 32 Intel Skylake vCPU

Ram: 120GB

GPU: 8x Nvidia Tesla V100 (16GB)

OS: Container-Optimized OS (COS) 85-13310.1209.17

Nvidia Driver: 450.51.06

CUDA: 11.0

Price/h: 14.355 USD

Estimated running time: 19100 USD

Estimated Total Cost: 19100 USD

A.1.2 Compute Engine Config

Series: A2

Machine type: a2-highgpu-8g

CPU: 96 intel Cascade Lake vCPU

Ram: 680GB

GPU: 8x Nvidia Tesla A100 (40GB)

OS: COS 85-13310.1209.17

Nvidia Driver: 460.32.03

CUDA: 11.2

Price/h: 30.022 USD

Estimated running time: 360h

Estimated Total Cost: 10800 USD

A.1.3 Compute Engine Config

Series: A2

Machine type: a2-megagpu-16g

CPU: 96 intel Cascade Lake vCPU

Ram: 1330GB

GPU: 8x Nvidia Tesla A100 (40GB)

OS: COS 85-13310.1209.17

Nvidia Driver: 460.32.03

CUDA: 11.2

Price/h: 56.665 USD

Estimated running time: 180h

Estimated Total Cost: 10200 USD

A.1.4 Instance Setup

Check disk portion assignment.

```
sudo fdisk -l
```

Create directory to bind externals disks to.

```
sudo mkdir /home/disk
sudo mkdir /home/data
```

Mount external drives to folder in container.

```
sudo mount /dev/diskPartion /home/disk/
sudo mount -o ro /dev/dataPartion /home/data/
```

For V100 Install nvidia drivers into COS

```
cos-extensions install gpu
sudo mount --bind /var/lib/nvidia /var/lib/nvidia
sudo mount -o remount,exec /var/lib/nvidia
/var/lib/nvidia/bin/nvidia-smi
```

Start the docker container in background mode and assign all devices to it.

```
docker run --shm-size 200G
    --volume /home/disk/:/home/disk/
    --volume /home/data/:/home/data/
    --volume /var/lib/nvidia/lib64:/usr/local/nvidia/lib64
    --volume /var/lib/nvidia/bin:/usr/local/nvidia/bin
    --device /dev/nvidia-uvm:/dev/nvidia-uvm
    --device /dev/nvidiactl:/dev/nvidiactl
    --device dev/nvidia0:/dev/nvidia0
    --device dev/nvidia1:/dev/nvidia0
    --device dev/nvidia2:/dev/nvidia0
    --device dev/nvidia3:/dev/nvidia0
    --device dev/nvidia4:/dev/nvidia0
    --device dev/nvidia5:/dev/nvidia0
    --device dev/nvidia6:/dev/nvidia0
    --device dev/nvidia7:/dev/nvidia0
    -d -it "link to docker container"
```

For A100 Install the latest Nvidia drivers. Note that a restart will be performed and when rebooted the command needs to be executed a second time.

```
/usr/bin/docker run --rm --privileged --net=host --pid=host
    --volume /dev:/dev --volume /:/root
    --volume /var/lib/toolbox/nvidia:/usr/local/nvidia
    --env NVIDIA_DRIVER_VERSION=460.32.03
    gcr.io/cos-cloud/cos-gpu-installer:latest
```

Start the docker container in background mode and assign all devices to it.

```
docker run --shm-size 200G
    --volume /home/disk/:/home/disk/
    --volume /home/data/:/home/data/
   --volume /var/lib/toolbox/nvidia/lib64:/usr/local/nvidia/lib64
    --volume /var/lib/toolbox/nvidia/bin:/usr/local/nvidia/bin
    --device /dev/nvidia-uvm:/dev/nvidia-uvm
```

```
--device /dev/nvidiactl:/dev/nvidiactl
--device dev/nvidia0:/dev/nvidia0
--device dev/nvidia1:/dev/nvidia0
--device dev/nvidia2:/dev/nvidia0
--device dev/nvidia3:/dev/nvidia0
--device dev/nvidia4:/dev/nvidia0
--device dev/nvidia5:/dev/nvidia0
--device dev/nvidia6:/dev/nvidia0
--device dev/nvidia7:/dev/nvidia0
-d -it "link to docker container"
```

A.2 AI-Sweden

Series: Data Factory

Ram: 1TB

GPU: 8x Nvidia Tesla A100 (40GB)

OS: Ubuntu (Nvidia BaseOS)

From this machine one GPU was made available for fine-tuning, more information about the AI-Sweden machine can be found on their website[1]

[1] https://www.ai.se/en/data-factory

B

wav2vec 2.0

This appendix details a few key elements of the wav2vec 2.0 development, where the setup and the detailed account of the approach with complete configs can be found in the GitHub repository[1].

B.1 Docker config

The docker container was created using NVIDIA hpc-container-maker based on the PyTorch 21.02 Release[2].

OS: Ubuntu 20.04

CUDA: 11.2.0

APEX: 0.1

KenLM: 0c4dd4e8a29a9bcaf22d971a83f4974f1a16d6d9

Flashlight: 1.0.0

Fairseq: 1.0.0a0+dd74992

The ready to use container configuration based on the AI-Sweden startupKit can be found in the repository.

[1] `https://github.com/JonathanGildevall/Automatic-Emergency-Detection-in-Naval-VHF-Transmissions`

[2] `https://docs.nvidia.com/deeplearning/frameworks/pytorch-release-notes/rel_21-02.html`

B.2 Pre-training config

For the pre-training a modified config from fairseq wav2vec 2.0 large but with a few modifications

Sample size: The sample size is how long the audio files are. This value is given in samples. As the data is sampled in 16kHz this means a audio file length between 4 and 20 seconds

Num workers: This is how many processes for data processes is available. The value of 64 was chosen given each GPU 8 workers

Max tokens: How many tokens is allowed to be in GPU memory for each GPU. Running on 8 A 100 GPUs gives an total memory usage of 300 GB

Distributed world size: How many GPUs that are available. Here 8 was used for the pre-training.

Update freq: To simulate a higher GPU count and a update frequency of 16 with a distributed world size simulates 128 GPUs. This was done to keep the same batch size as in the wav2vec 2.0 large training done by Facebook.

B.3 Fine-tuning config

For the fine-tuning two modified configs from fairseq wav2vec 2.0 large was used, selected based on the size of the dataset. The 10m config was used for the Swedish tuning and the 1h config for the English. They both run for 13 000 updates with the first 10 000 updates only affected the added final layer. They differ in the learning rate, where the smaller dataset uses a rate of 0.0001 and the larger uses 0.0003.

distributed world size: How many GPUs that are available. Here 1 was used for the fine-tuning.

Update freq: To simulate a higher GPU count and a update frequency of 24 with a distributed world size simulates 24 GPUs. This was done to keep the same batch size as in the wav2vec 2.0 large fine-tuning.

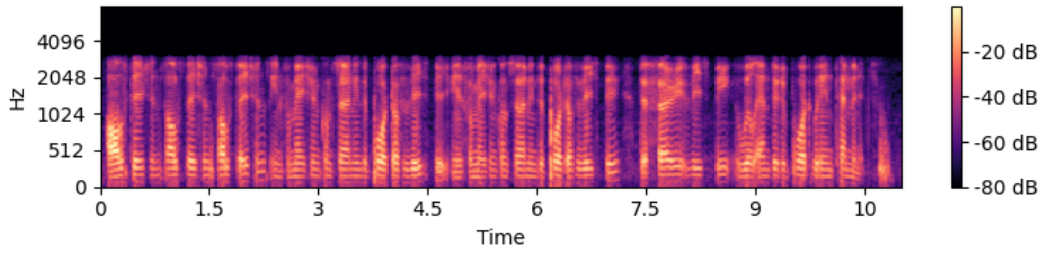
C

Google Speech-To-Text results

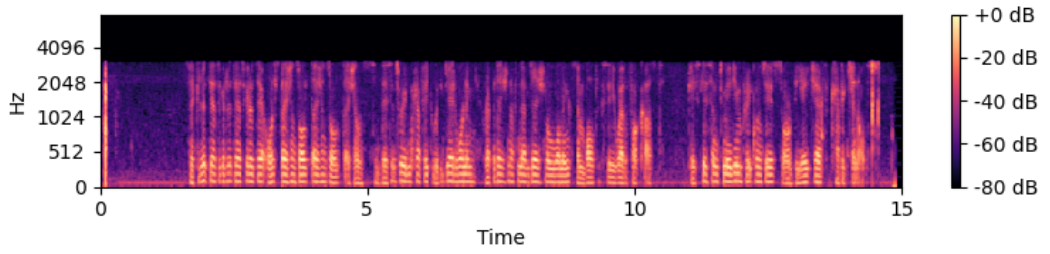
The two tables below details the words used in the *speech_context* and the different API configurations.

	speech_context
	mayday
	pan
	allmänt anrop
	sweden rescue
	sjöräddningsenhet
	sunningesundet
	all stations
	alla båtar

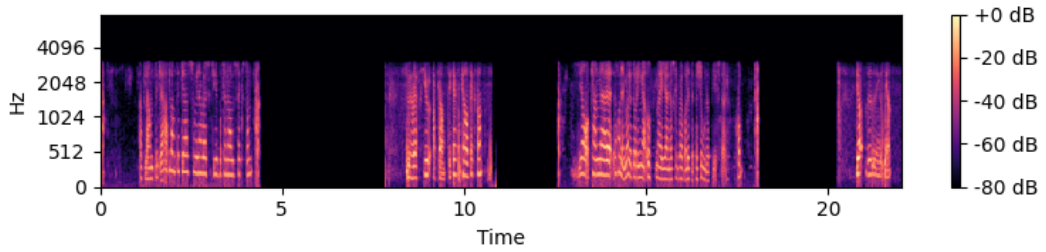
Configuration	Parameters
Default SV	`{language_code="sv-SE"}`
Default EN	`{language_code="en-US"}`
Command SV	`{language_code="sv-SE", model="command_and_search"}`
Command EN	`{language_code="en-US", model="command_and_search"}`
Speech Context SV	`{language_code="sv-SE", speech_contexts=[`*speech_context*`]}`
Speech Context EN	`{language_code="en-US", speech_contexts=[`*speech_context*`]}`



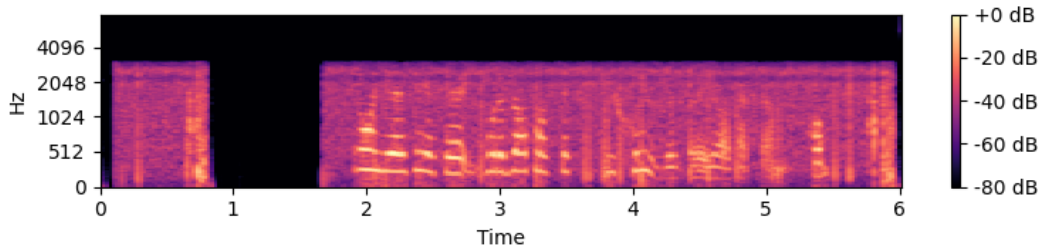
Configuration	Transcription
Reference	all stations all stations all stations information concerning the port of visby information concerning the port of visby the ferry visby will enter the port within ten minutes
Default Swedish	Alstigen 12 station sopstation information concerning the Port Of Visby information concerning the Port Of Visby the Ferry Visby Willy and the poor people in ten minutes
Default English	all Station South Station South Station information concerning the port of little information concerning the port of the state will enter the port within 10 minutes
Command Swedish	Alstigen 12 station sopstation information concerning the Port Of Visby information concerning the Port Of Visby the Ferry Visby Willy and the poor people in ten minutes
Command English	all Station South Station South Station information concerning the port of little information concerning the port of the state will enter the port within 10 minutes
Speech_Context Swedish	all stations all stations all stations information concerning the Port Of Visby information concerning the Port Of Visby the Ferry Visby Willy and the poor people in ten minutes
Speech_Context English	all stations all stations all stations information concerning the port of little information concerning the port of the state will enter the port within 10 minutes



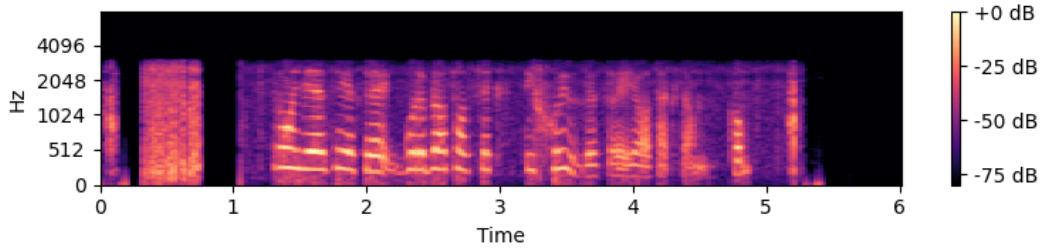
Configuration	Transcription
Reference	securite securite securite all stations all stations all stations this is sweden traffic with gale warning repetition of navigational warnings and baltic sea weather forecast please listen to medium frequencies or vhf traffic channels
Default Swedish	säker att jag skulle jag skulle säga åt dig som sålt i centralstation the Swedish traffic jam warning reputation of the Baltic Sea weather forecast please listen to me game princess of professionals
Default English	take a look at to go to Tesco to stay on station Soul II Soul station reputation of navigational warnings and Baltic Sea weather forecast please listen to medium frequency check traffic channels
Command Swedish	Tycker du att jag ska till skulle till Årsta consultations all types of the Swedish Classic reputation of navigation ordningsam Baltic Sea weather forecast please listen to me frequency srhf traffic channels
Command English	take a look at to go to Tesco to stay on station Soul II Soul station reputation of navigational warnings and Baltic Sea weather forecast please listen to medium frequency check traffic channels
Speech_Context Swedish	säker att jag skulle jag skulle säga all stations Voltaire centralstation the Swedish traffic jam warning reputation of the Baltic Sea weather forecast please listen to me game princess of professionals
Speech_Context English	take a look at to go to Tesco they're all stations all stations all stations reputation of navigational warnings and Baltic Sea weather forecast please listen to medium frequency check traffic channels



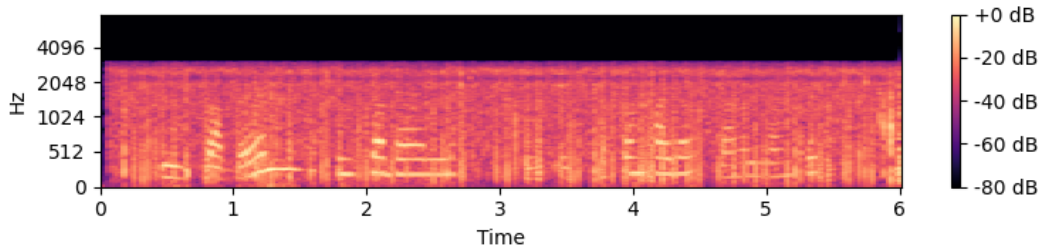
Configuration	Transcription
Reference	atlantica atlantica sweden rescue på kanal sexton sweden rescue atlantica på kanal sexton ja kan har ni möjlighet att ringa upp mig igen jag skulle behöva lite personuppgifter kom ja vi ringer upp igen
Default Swedish	Atlantica Atlantica Sweden Rescue på Kanal 6 Atlantica 16 jag kan han är möjligt att ringa upp mig igen Jag skulle behöva lite personuppgifter om vi ringer sen
Default English	Atlantic Atlanta jazz Suite arrest you but can all Sexton
Command Swedish	Atlantica Atlantica Sweden Rescue kanal 16
Command English	Atlantic Atlanta jazz Suite arrest you but can all Sexton
Speech_Context Swedish	Atlantica Atlantica of Sweden Rescue på Kanal 6 Atlantica 16 jag kan han är möjligt att ringa upp mig Jag skulle behöva lite personuppgifter om vi ringer sen
Speech_Context English	Atlantic Atlanta Jazz Sweden rescue Beacon all Sexton



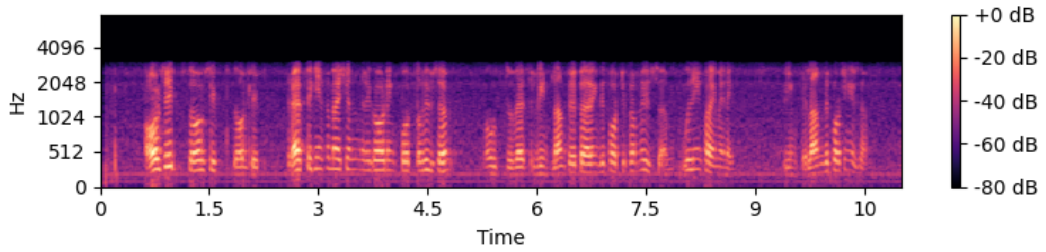
Configuration	Transcription
Reference	från jrcc går det bra att ta sextiosju så är jag tacksam kom
Default Swedish	Går det bra på en 67 därför jag kanske kompis
Default English	what is half of 64
Command Swedish	Ronja GPS Går det bra på en 67 därför jag kom
Command English	story ideas that could have dropped off 60 Freeway
Speech_Context Swedish	Går det bra på en 67 därför jag kanske kompis
Speech_Context English	what is half of 64



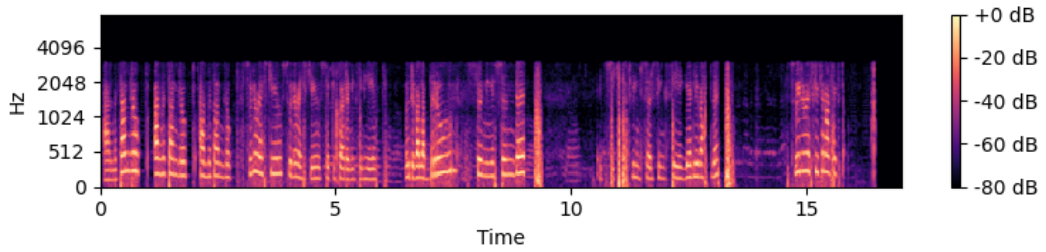
Configuration	Transcription
Reference	från jrcc går det bra att ta sextiosju så är jag tacksam kom
Default Swedish	från tncc Går det bra 67 så är jag tacksam
Default English	Philadelphia said what about the sixties Quizlet
Command Swedish	Går det bra att ha 67 så är jag tacksam
Command English	going to see if they would have dropped off 60 Quizlet
Speech_Context Swedish	från tncc Går det bra 67 så är jag tacksam
Speech_Context English	Philadelphia said what about the sixties Quizlet



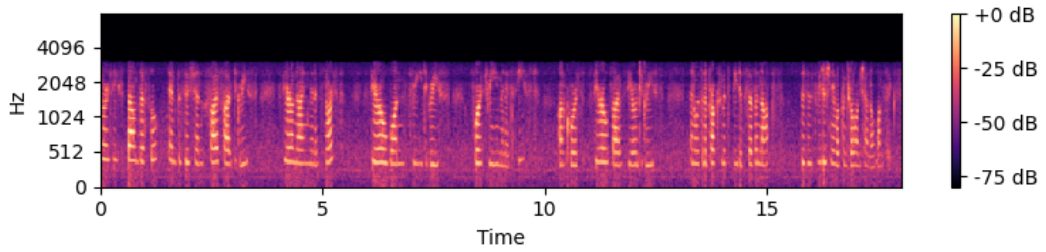
Configuration	Transcription
Reference	isabelle isabelle this is stockholm pilot channel one six
Default Swedish	Isabelle Isabelle
Default English	
Command Swedish	Isabelle
Command English	
Speech_Context Swedish	Isabelle Isabelle
Speech_Context English	



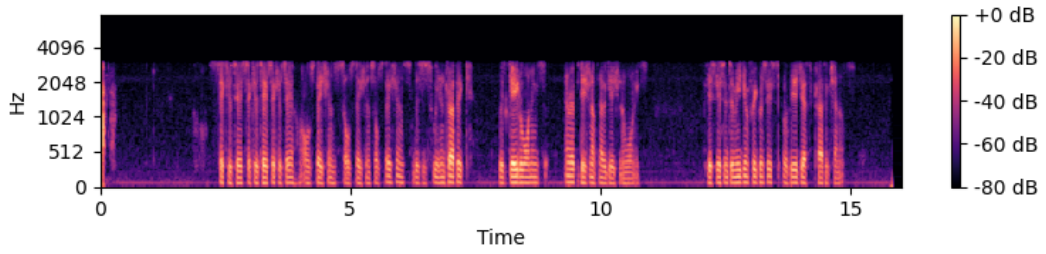
Configuration	Transcription
Reference	all ships all ships all ships traffic information regarding port of husö motor vessel vinterland preparing departure from husö within in five minutes vinterland departing husö
Default Swedish	all ships Star Citizen ship traffic information regarding på Husum vinterlandet
Default English	traffic information regarding photo loser
Command Swedish	all ships Star Citizen ship traffic information regarding Vinterland
Command English	traffic information regarding photo loser
Speech_Context Swedish	all ships Star Citizen ship traffic information regarding på Husum vinterlandet
Speech_Context English	traffic information regarding photo loser



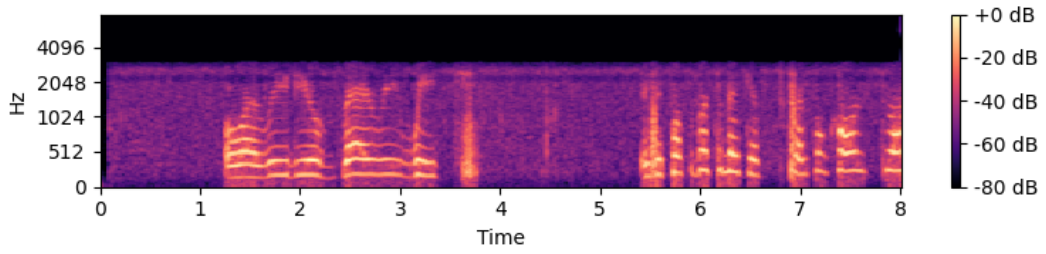
Configuration	Transcription
Reference	allmänt anrop allmänt anrop allmänt anrop sweden rescue sweden rescue city sjöräddningsenhet uddevallabron sunningesundet ett siktat windsurfingsegel i vattnet sweden rescue kanal sexton
Default Swedish	Alexandra Alexandra Alexandra Rescue Swedish music Academy Uddevallabron Sunes under segel i vattnet
Default English	Alexandra Alexandra Alexandra cigarettes juice with the rescues to cancel at any CVS
Command Swedish	Alexandra Alexandra Alexandra Rescue Swedish music Academy Uddevallabron sunninge Sundet
Command English	Alexandra Alexandra Alexandra cigarettes juice with the rescues to cancel at any CVS
Speech_Context Swedish	allmänt anrop Alessandro juicy juicy sjöräddningsenhet sunninge sundet segel i vattnet
Speech_Context English	Alexandra Alexandra Alexandra cigarettes juice with the rescues to cancel at any CVS



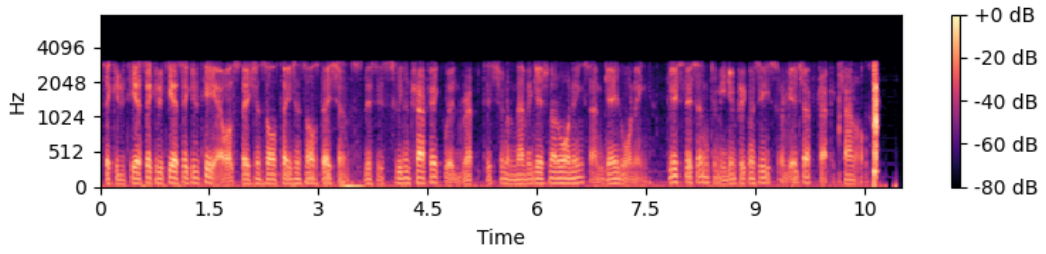
Configuration	Transcription
Reference	north seas bound vessel in position five five degrees zero nine minutes north zero one three degrees four three minutes east on course zero five zero degrees and with an approximate speed of seven knots this is swedish naval control on channel one six
Default Swedish	Precision 50 Greece 49 minutes North 30 degrees for 3 minutes is on course Hero of nuts precis Vill du skriva på grantjärn om sex
Default English	5 degrees
Command Swedish	Precision 50 Greece 49 minutes North 30 degrees for 3 minutes is on course 45 degrees to swedish Channel One sex
Command English	Northeast Bountiful and precision 5530 9 Minutes North 013 degrees 43 minutes east on Coursey 0 5 0 degrees and with an approximate speed of 7 knots is a Swedish table control on Channel one-six
Speech_Context Swedish	Precision 50 Greece 49 minutes North 30 degrees for 3 minutes is on course Hero of nuts precis Vill du skriva på grantjärn om sex
Speech_Context English	5 degrees



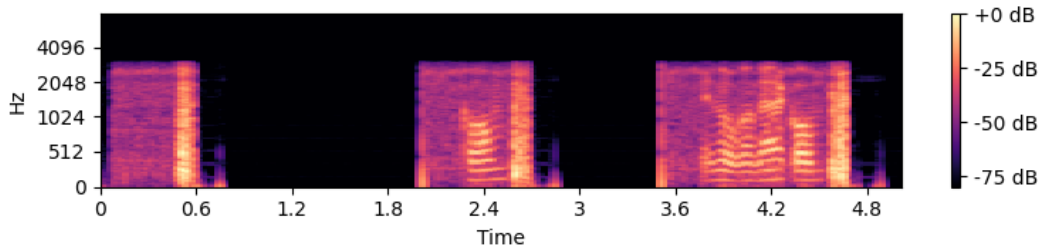
Configuration	Transcription
Reference	securite securite securite all stations all stations all stations this is sweden traffic with gale warnings and repetition of navigational warnings please listen to medium frequencies or the vhf traffic channels
Default Swedish	sekretess sekretess sekretess och station centralstation station is Sweden traffic with the Mornings navigation Mornings security
Default English	security security hold station substation substations this is sweet and traffic with Gale warnings and repetition of navigational morning
Command Swedish	sekretess sekretess sekretess station sopstation som station BBC Sweden traffic with the Wings navigation morning
Command English	602 security security hold station Seoul Station Sub Station this is sweet and traffic with Gator warnings and repetition of navigation warning
Speech_Context Swedish	sekretess sekretess sekretess all stations tågstation tågstation i Sweden traffic with the Mornings navigation Mornings
Speech_Context English	securite securite securite all stations sell station substations this is sweet and traffic with Gale warnings and repetition of navigational warnings



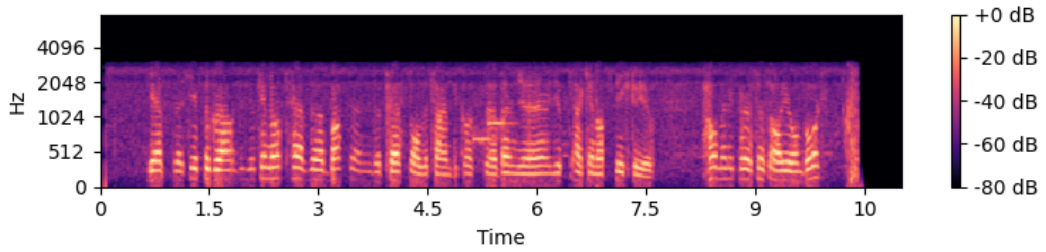
Configuration	Transcription
Reference	oh I read you very weak Is it possible to make a telephone call to us
Default Swedish	Audio Video very week
Default English	oh I read you very weak
Command Swedish	Audio Video very week
Command English	oh I read you very weak
Speech_Context Swedish	all with you very weak
Speech_Context English	oh I read you very weak



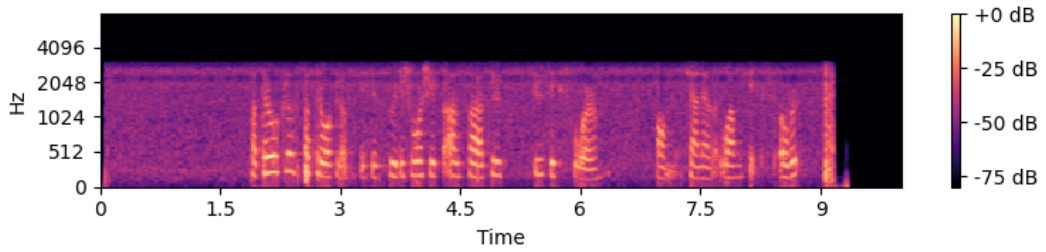
Configuration	Transcription
Reference	securite securite securite all stations all stations all stations this is sweden traffic with repetition of navigational warnings and gale warning please listen to medium frequencies or vhf traffic channels
Default Swedish	sekreterare sekreterare sekreterare all stations avstängd centralstation Miss Sweden traffic with rapture navigation morning Sunday Morning please listen to me please sorry I cannot
Default English	checking to see if they could see if they can see all the stations all stations all stations this is sweet and traffic with repetition of navigational warnings on gale warning please listen to medium frequency so I get a chance
Command Swedish	sekreterare sekreterare sekreterare all stations avstängd centralstation Miss Sweden traffic with rapture navigation morning Sunday Morning please listen to me please sorry I cannot
Command English	checking to see if they could see if they can see all the stations all stations all stations this is sweet and traffic with repetition of navigational warnings on gale warning please listen to medium frequency so I get a chance
Speech_Context Swedish	sekreterare sekreterare sekreterare all stations Australian stations Miss Sweden traffic with rapture navigation morning Sunday Morning please listen to me please sorry I cannot
Speech_Context English	taking the TSI Cupra TSI give it to you all stations all stations all stations this is sweet and traffic with repetition of navigational warnings on gale warning please listen to medium frequency so I get a chance



Configuration	Transcription
Reference	kom är någon där kom
Default Swedish	Kom här kom
Default English	Comfort Inn
Command Swedish	Kom här kom
Command English	Comfort Inn
Speech_Context Swedish	Kom här kom
Speech_Context English	Comfort Inn



Configuration	Transcription
Reference	yes ships aground you can not have the button depressed all the time because then you i can not speak to you how many persons are you onboard
Default Swedish	Yes you can not have the button the press all the time Because I cannot speak to you How many percent are you bored
Default English	skip that if the ground you cannot have the bunker and depressed all the time because the Avengers I cannot speak to you how many persons are you on board
Command Swedish	Yes you can not have the button the press all the time Because I cannot speak to you How many percent are you bored
Command English	skip that if the ground you cannot have the bunker and depressed all the time because the Avengers I cannot speak to you how many persons are you on board
Speech_Context Swedish	Yes you can not have the button the press all the time Because I cannot speak to you How many percent are you bored
Speech_Context English	skip that if the ground you cannot have the bunker and depressed all the time because the Avengers I cannot speak to you how many persons are you on board



Configuration	Transcription
Reference	patroklos patroklos this is swedish warship alpha two two six four on channel one six over
Default Swedish	patron Rusta trådlös sticker Swedish italfarad c64 on Channel One Six
Default English	hotel close Petrol kiosk it is Swedish worship Alpha 2264 Channel
Command Swedish	trådlös trådlöst icke Swedish Warship Alltså jag typ 10 64 on Channel One Six over
Command English	Swedish worship Alpha to 26416
Speech_Context Swedish	patron Rusta trådlös sticker Swedish italfarad c64 on Channel One Six
Speech_Context English	hotel close Petrol kiosk it is Swedish worship Alpha 2264 Channel

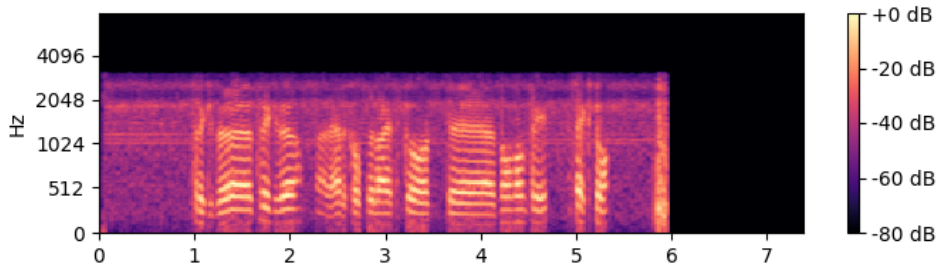
D

wav2vec 2.0 results

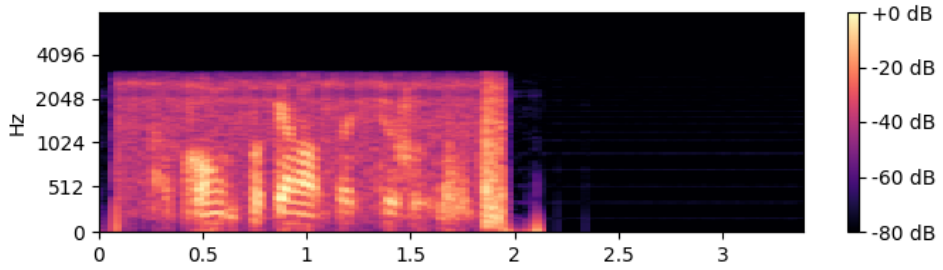
The table below details the different wav2vec 2.0 models used in the thesis.

Pre-trained models fine-tuned on thesis data	
LARGE	Pre-trained on LibriSpeech
JRCC	The large model with JRCC transfer learning
XLSR-53	Pre-trained on multilingual data
LARGE_VOX	Pre-trained on the European Parliament data set
LARGE_SV	Pre-trained on the Swedish subset of the European Parliament data
Complete models	
XLSR-SV	XLSR-53 extended with Swedish pre-training and fine-tuning
LARGE_10m	Fine-tuned on 10 minutes of LibriSpeech
LARGE_960h	Fine-tuned on 960 hours of LibriSpeech

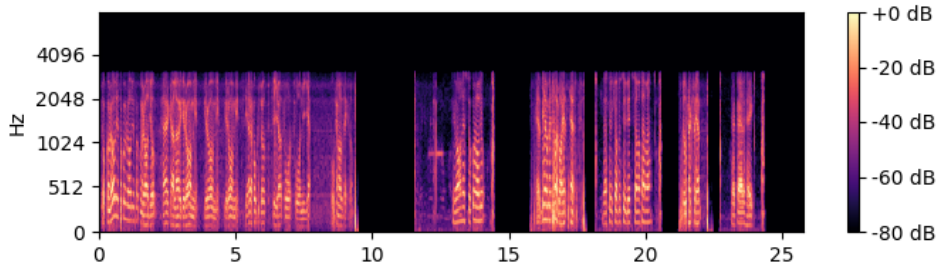
Swedish



Model	Transcription	WER	CER
Reference	TRYGGVE TRYGGVE HÄR ÄR HELIKOPTER LIFEGUARD NOLL NOLL TRE SEXTON		
Fine-tuned on other data			
XLSR-SV	TRYKDERER TREGRÖ MER ELIKOPTERLITGAT NONOM PRIS SEKTORN	100	48
Fine-tuned on domain data			
JRCC	E	100	98
XLSR-53	TREGEVÖ TRGEVÖ HÄR HÄLE KOM SELASKGTE NOL NON TRI SEXTON	80	45
LARGE_SV	SEEE	100	95
LARGE_VOX	TRGBÖ TRUGVÖ HÄR ÄL KOWELAS GASTE SNO VAN TRE SEXTON	80	48
Swedish-English			
XLSR-53		100	100

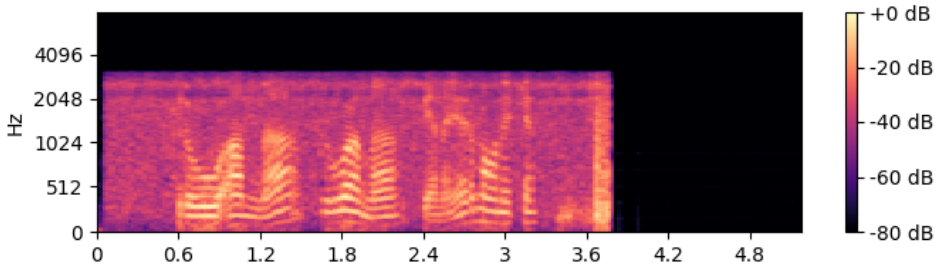


Model	Transcription	WER	CER
Reference	KOM PÅ KANAL SEXTIOSJU		
Fine-tuned on other data			
XLSR-SV	ATTÅNG FÅT GANALSEK TUSHUTOT	100	82
Fine-tuned on domain data			
JRCC	E	100	95
XLSR-53	AR KOMSPÅ KANAL SEXTUESJU	75	26
LARGE_SV	JE	100	95
LARGE_VOX	AKOM PÅ KANAL SEXTIOSJU TVÅT	50	30
Swedish-English			
XLSR-53		100	100

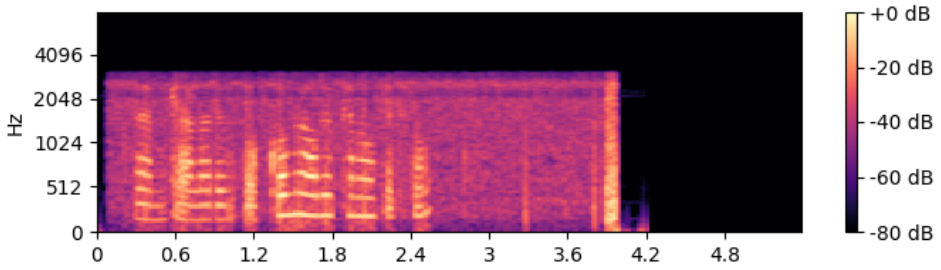


Model	Transcription	WER	CER
Reference	STOCKHOLM RADIO STOCKHOLM RADIO STOCKHOLM RADIO HÄR KALLAR GRAMI GRAMI GRAMI SIERRA FOXTROT FYRA TVÅ TVÅ NIO PÅ KANAL SEXTON GRAMI STOCKHOLM RADIO BER OM ETT HÖRBARHETSTEST LÄSER ER KLART OCH TYDLIGT FEMMA FEMMA STORT TACK FÖR DET TACK SJÄLV HEJ		
Fine-tuned on other data			
XLSR-SV	STOCKOMRADIO SLOCKOM RADIO SLOCKOM RADIO HÄR KALLAR GRAMI GYRAMI GRAMI FÄRA FOXTROTT FYRA TVÅ TVÅ NYO TVÅ KANALSEXTORNRA NÄ STOCKOMRADIOEMER OM ETT FÖRBARHETS TÄSTLÄSTER VI KRAV NA TUDLIGT SÖMMAS HEMMA STOR TACK FÖR DET TACK KÄR HEJ	60	22

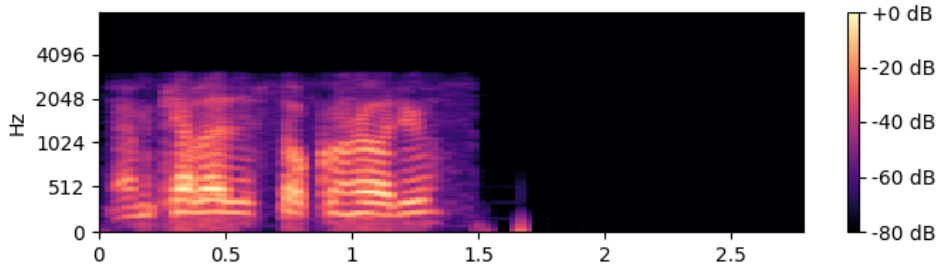
Fine-tuned on domain data			
JRCC	E	100	99
XLSR-53	STOCKHOLM RADIO STOCKHOLM RADIO STOCKHOLM RADIO HÄR KALLAR GRAMI GRAMI RA MI SIHÄRA FOCSTROÅT FIRA FVÅ FÅ NIA FÅ KANAL SEXTON GEVANE STOCKHOLM RADIOE VER OMET FÖRBA HET EST VÄR SERVI KLASE TULIT SEN VA FEMA DORTACK FRE TÄA SHÄDE HE	70	24
LARGE_SV	SL AUNT SER AR A S V A R	100	91
LARGE_VOX	STOCKHOLM RADIO STOCKHOLM RADIO STOCKHOLM RADIO HÄR KALLAHÄR GRAMI GRAI GRAMI SRA FOCSTROT FIRA TVÅ TVÅ NIO FO KANAL SEXTON GRANE STOCKHOL RADIO BER OM MET SVÖRBARHET T DÄR SERIG KLART FVO TIYLIT SENNAR FEMMASTOR TACK FÖRDE TRÄC SKÄDEOHÄ	56	21
Swedish-English			
XLSR-53	OA P	100	96



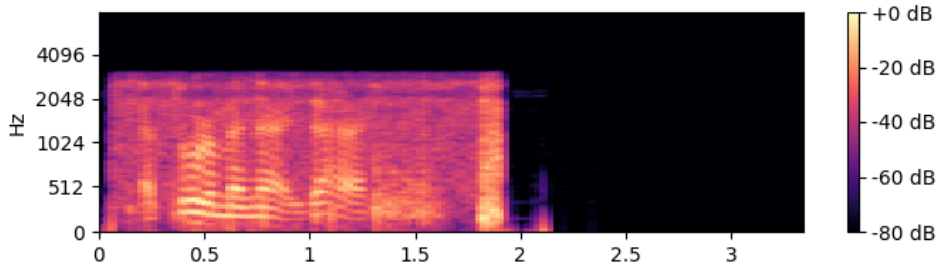
Model	Transcription	WER	CER
Reference	BRO ANNA BRO ANNA STENA GERMANICA		
Fine-tuned on other data			
XLSR-SV	LÅ ANNA FO ANNA FENA I ARMANIKER	83	39
Fine-tuned on domain data			
JRCC	E	100	96
XLSR-53	KRO ANNA RENNA SENN ER MAÅNIKE	83	39
LARGE_SV	RELE	100	90
LARGE_VOX	KRO ANNA FREGJÄNNAR SENE ER MA NIKE	100	45
Swedish-English			
XLSR-53	A	100	96



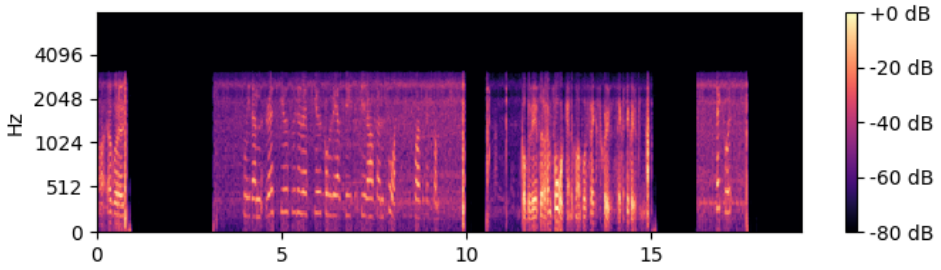
Model	Transcription	WER	CER
Reference	VEM KALLADE STOCKHOLM RADIO KANAL SEXTON		
Fine-tuned on other data			
XLSR-SV	FEM KALLADESÅKOM RADIO KANALSEXOS	83	25
Fine-tuned on domain data			
JRCC	E	100	97
XLSR-53	EM KALLADE STOCKHOLM RADIO KANAL SEXTON	16	2
LARGE_SV	JA	100	97
LARGE_VOX	EM KALLADE STOCKHOLM RADIO KANAL SEXTON	16	2
Swedish-English			
XLSR-53	AAA	100	92



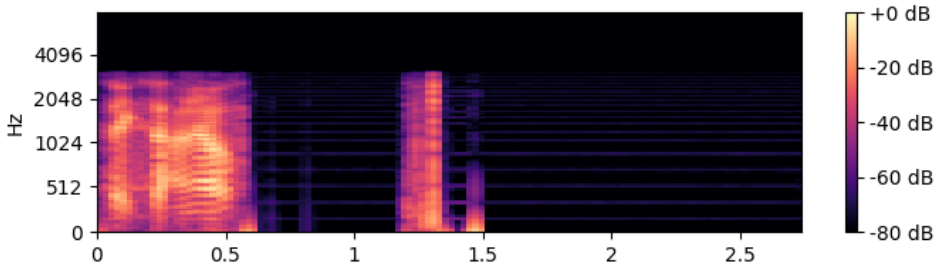
Model	Transcription	WER	CER
Reference	VEM KALLADE STOCKHOLM RADIO		
Fine-tuned on other data			
XLSR-SV	VEM KALLADE STOCKHOLMRADIO	25	3
Fine-tuned on domain data			
JRCC	E	100	96
XLSR-53	EM KALLADE STOCKHOLM RADIO	25	3
LARGE_SV	J	100	96
LARGE_VOX	VEM KALLADE STOCKHOLM RADIO	0	0
Swedish-English			
XLSR-53	A	100	96



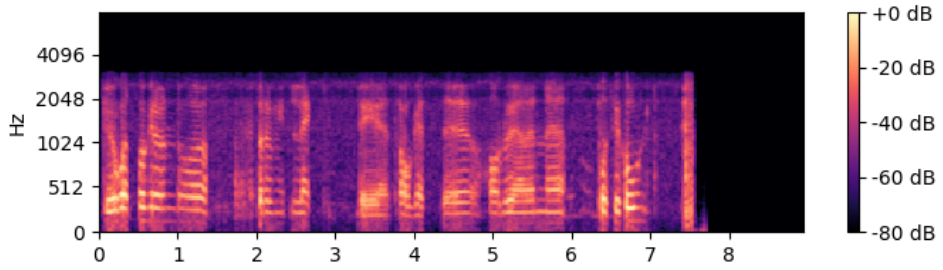
Model	Transcription	WER	CER
Reference	JAG TROR DOM ÄR NÖJDA HÄR		
Fine-tuned on other data			
XLSR-SV	JAG TROR DE ÄR NÖJDA HÄR HILES	33	28
Fine-tuned on domain data			
JRCC	E	100	100
XLSR-53	E TOLMENNE DA HÄR	83	57
LARGE_SV	VE	100	100
LARGE_VOX	ASTOR ME NÄDA HÄR	83	46
Swedish-English			
XLSR-53		100	100



Model	Transcription	WER	CER
Reference	AH JAG GÅR UR SWEDEN RESCUE SWEDEN RESCUE K B V FY FYRA FEM TVÅ SVAR PÅ SEXTON K B V FYRA FEM TVÅ KAN DU KOMMA PÅ SEX SJU SEXTIOSJU SEX SJU		
Fine-tuned on other data			
XLSR-SV	JAG GÅR UTSÅIDEL RESLUSUDVESTGU KOMMER VI AT I FYRA AV FEM TVÅ VAR FÖR SEXTOKOBBER VI FÖRA HEMTÅG KAN DU KOMMA PÅ SEX SJU SEXTIO SJU INMMARSEX OE	72	42
Fine-tuned on domain data			
JRCC	E	100	99
XLSR-53	AH JÖRGÅRSWEDEN RESCUE SWEDEN RESCUE KABIVEI FRA EN TÅA AR FÅ SEXTON GABEVSRAR HEM TÅ KANDE KON A PÅ SEXS SJY SEXSTIOSJY SEXK SFJU	81	27
LARGE_SV	ALR J SU JA A	100	89
LARGE_VOX	JA JA GÅR UTSWEDEN RESCUESWE DE RESCUE KOMER VIASCU FYRA FEM TVÅSVAR FÖER SEXTON KOBE V FÖERAR FEM TVÅ KAN DO KOLMAR PÅ SEX SJU SEXTISJU FLN SEXSKJU	63	26
Swedish-English			
XLSR-53	AU O	100	95

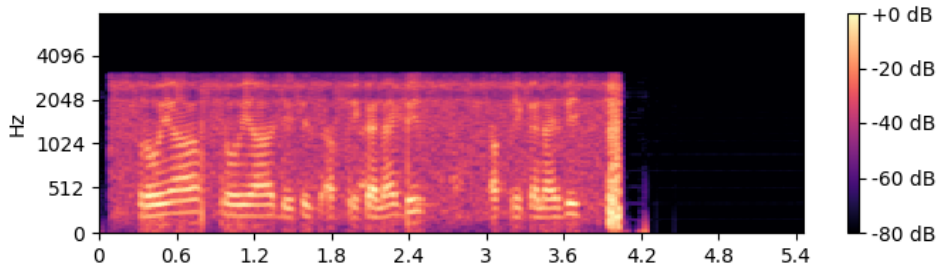


Model	Transcription	WER	CER
Reference	SEXAN JA		
Fine-tuned on other data			
XLSR-SV	TECXA HAN JAG	100	75
Fine-tuned on domain data			
JRCC	E	100	87
XLSR-53	TEXCKFAN AO	100	75
LARGE_SV	R	100	100
LARGE_VOX	TEKSFAAN AO	100	87
Swedish-English			
XLSR-53	A	100	87

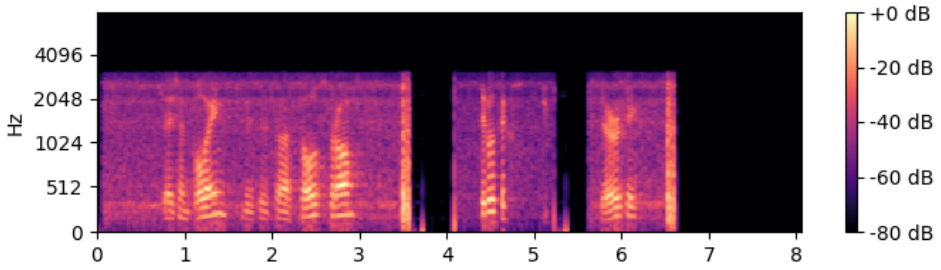


Model	Transcription	WER	CER
Reference	DU HAR GÅTT PÅ GRUND OCH SPRUNGIT LÄCK DET ÄR TAGE HAR DU EN POSITION		
Fine-tuned on other data			
XLSR-SV	DU ÅGÅT PÅ GRUND OCHPRUNGIT LÄCK DET TAGET HAR DU EN EN POSITION	46	20
Fine-tuned on domain data			
JRCC	E	100	98
XLSR-53	GJULBVAT PÅGRUND ORR STERUO MIT LC DE TAGESTE HARBDÄENE PÅSITION	100	49
LARGE_SV	SER T	100	93
LARGE_VOX	DJ JOÅGÅT PÅ GRUND OA SFERU MNIT LEK DE TAGETTTST HAR BVÖÄREN POSITION	80	43
Swedish-English			
XLSR-53		100	100

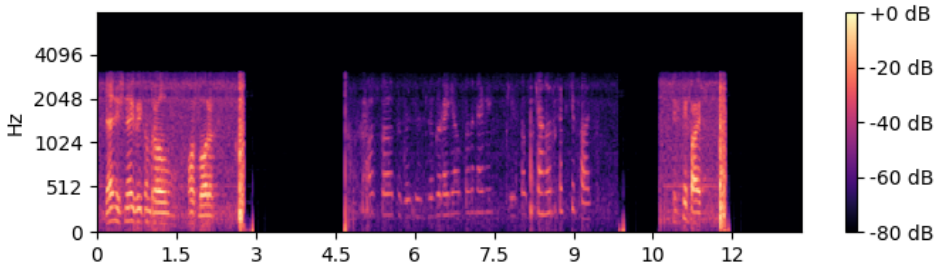
English



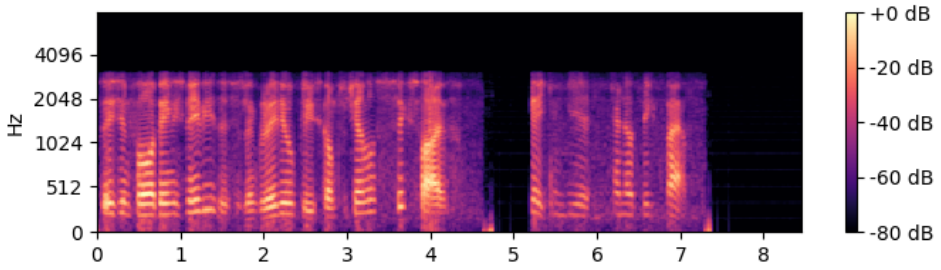
Model	Transcription	WER	CER
Reference	TROJA TROJA THIS IS AFRICAN IBIS AFRICAN IBIS		
Fine-tuned on other data			
LARGE_10m	UTWYAR STR WYARD IS AIS UTIG ANAVIS SHILE TAK AN IVBISS	100	72
LARGE_960h	FROYA FROYA DESIS AFICANABIS AFRICANABIS	100	28
Fine-tuned on domain data			
JRCC		100	100
LARGE	PROIA PROA THIS IS ASSTHRICA NABI ASSTRICA NAIVY	75	44
LARGE_VOX	SFROEJA SROEA THIS IS AFRICA NAGBY AFHRICANIDI	75	31
Swedish-English			
XLSR-53	S	100	97



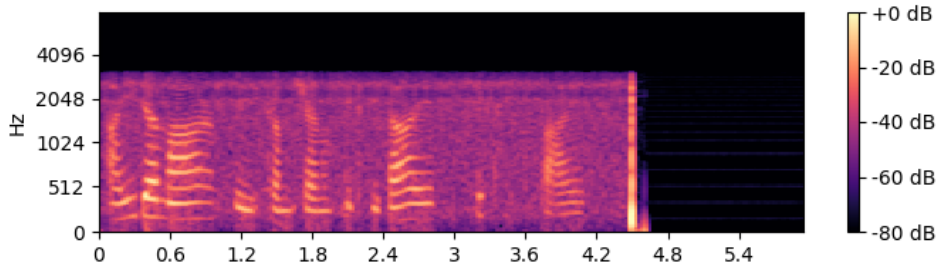
Model	Transcription	WER	CER
Reference	STATION CALLING THIS IS SOLSTRAND ZERO SIX ZERO SIX		
Fine-tuned on other data			
LARGE_10m	SAIS AND POLANE SIS IS AR SALS THRONG SUFITDLE SCXSIR THEY PT	100	77
LARGE_960h	SESAN POLENS MISESA SOLSTRAMP SITOSIX SERTIX	100	48
Fine-tuned on domain data			
JRCC		100	100
LARGE	STATION CALLING THIS ISA SALLTRAND ZERO SIX ZERO SIX	22	5
LARGE_VOX	STATION CALLING THIS IS SALSROND ZERO SIX ZERO SIX	11	5
Swedish-English			
XLSR-53	SON	100	92



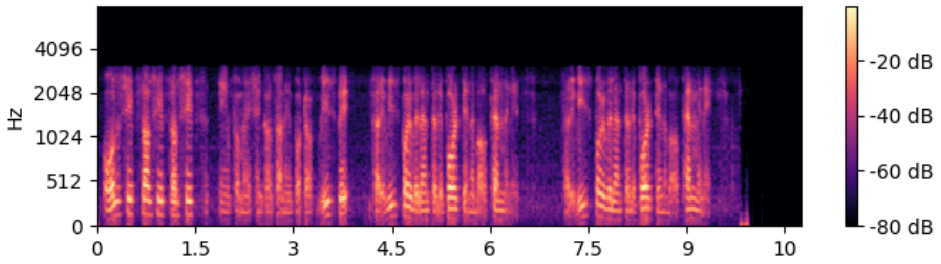
Model	Transcription	WER	CER
Reference	DANISH NAVY CONTROL OSBORG OSBORG THIS IS LYNGBY RADIO FOR THE NAVY PLEASE GO TO CHANNEL SIXTY FIVE SIXTY FIVE		
Fine-tuned on other data			
LARGE_10m	THEN HIS NAVY COMNTROLE WHAS BARTER HATRT BO HO T T THA TE RADIO TO THE NAVY T GRANM E OIPIED SIXTY FIVE	100	55
LARGE_960h	TANIS NAVY CONTROL OSBORUS CROS BOD TE TISIS I BEREGIOO THE NAVY O E CHANEL SIXTY FIVE SIXTY FIVE	65	32
Fine-tuned on domain data			
JRCC		100	100
LARGE	DANISH NAVAY CONTROL OSBOARO HASBOUT THIS IS LYNGBY RADIO FOR THE NAV PLEASE GO TO CHANNEL SIXTY FIVE SIXTY FIVE	20	7
LARGE_VOX	DANISH NAVY CONTROL OSBOR HAWSBORG THIS IS LYNGBY RADIO FWOR THE NAVI PLEASE CO TO CHANNEL SIXTY FIVE SIXTY FIVE	25	6
Swedish-English			
XLSR-53	A H	100	96



Model	Transcription	WER	CER
Reference	STATION FOR DANISH NAVY THIS IS LYNGBY RADIO PLEASE COME TO CHANNEL SIX FIVE OKAY CHANGING CHANNEL SIX FIVE		
Fine-tuned on other data			
LARGE_10m	STACION FOR BANYS NAVY THIS IS LONGOIRADIOU PLESE COME TO CHENALS SIX FIVE ACH NDERE CANO BIG FMOWTHH	58	32
LARGE_960h	ATION FOR DANISH NAVY THE SITLING GERADIO PLEASE COME TO CHELUS SIX FIVE ATIN DEA ANNO BIG MOUTH	58	35
Fine-tuned on domain data			
JRCC		100	100
LARGE	ASTATION FOR DANISH NAV THIS IS LYNGBY RADIO PLEASE COME TO CHANNEL SIX FIVE A TN BY CHANNEL SIX FIVE	31	12
LARGE_VOX	STATION FOR DANISH NAVY THIS IS LYNGBY RADIO PLEASE COME TO CHANNEL SIX FIVE KEY TN BY CHANNEL SIX FIRE	21	9
Swedish-English			
XLSR-53	STNN Y E HEN	100	88

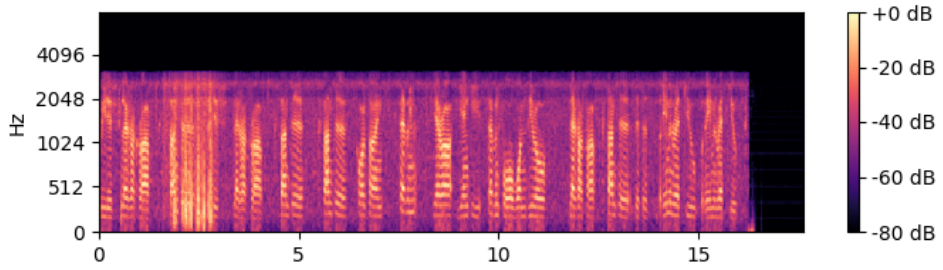


Model	Transcription	WER	CER
Reference	AIDAMAR PLEASE COME CHANNEL SIXTY FIVE SIX FIVE		
Fine-tuned on other data			
LARGE_10m	A EDAMAR SAINT COMN CANO SIXTY FIVE SIXTY FIVE SS	100	37
LARGE_960h	AIDAMAR SICAMCAN SIXTY FIVE SIX FIVE	37	27
Fine-tuned on domain data			
JRCC		100	100
LARGE	AIMA PLEASE COME TO CHANNEL SIXTY FIVE S FIVE	37	17
LARGE_VOX	ADA MAR PLEASE COME CHANNEL SIXTY FIVE SIX FIVE	25	4
Swedish-English			
XLSR-53		100	100



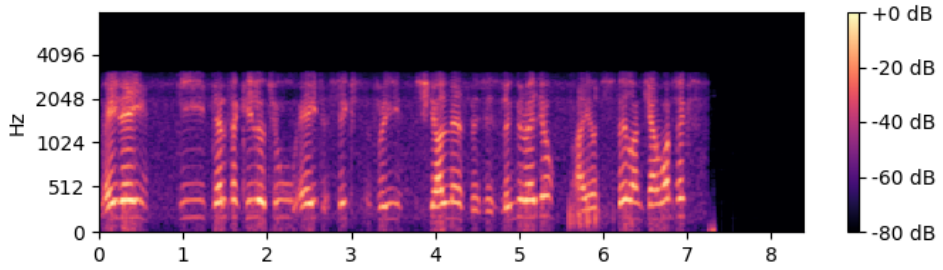
Model	Transcription	WER	CER
Reference	ALL SHIPS ALL SHIPS ALL SHIPS INFORMATION CONCERNING PORT OF VISBY FERRY VISBORG WILL ENTER THE PORT IN ABOUT TEN MINUTES FERRY VISBORG WILL ENTER THE PORT IN ABOUT TEN MINUTES		
Fine-tuned on other data			
LARGE_10m	AL SHIP SALD SHIP SOLD SHIPS INFORMATION CONSERNING PUTROF WVILS PKFOR IVIS PORYVEL ENTERED HE PORT IN ABOUT TEN MINITH FOR IVIS PORIVELENTERED E PORT IN ABOUT TEN MINAITH	68	30
LARGE_960h	ALL SHIPS OL SHIPS ON SHIPS INFORMATION CONCERNING PORTOP VISPE FA RY VISOY WILL ENTO REPORT IN ABOUT TEN MINUTES FARYVISPO WILLENTO REPORT IN ABOUT TEN MINUTES	52	17

Fine-tuned on domain data			
JRCC		100	100
LARGE	ALL SHIPS ALL SHIPS ALL SHIPS INFORMATION CONCERNING PORT OF VISBY FAR VISBOR WIL ENTER TEPORT IN ABOUT TEN MINUTES FR VISBOR WIL ENTR THE PORT IN ABOUT TEN MINUTES	32	7
LARGE_VOX	ALL SHIPS ALL SHIPS ALL SHIPS INFORMATION CONCERNING PORT OF VYSBY FIR O VYSBOR VILENTER DEPORT IN ABOT TEN MINUTES FIR O VYSBORG VWILLENTE DHEPORT IN ABOUT TEN MINUCTE	48	13
Swedish-English			
XLSR-53	ASSS LS SA ALS PSPSIIINNOOSTEN	100	85

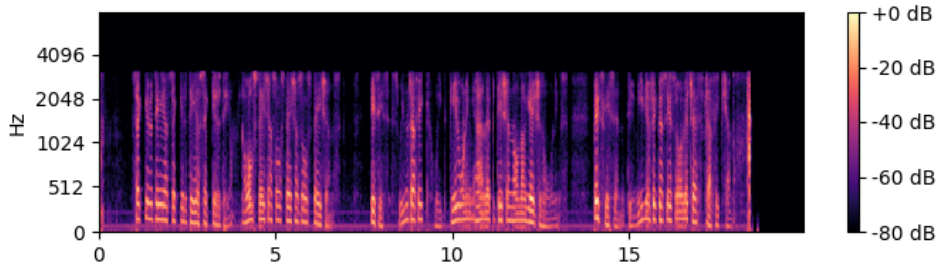


Model	Transcription	WER	CER
Reference	SWEDISH PLEASURE CRAFT XANTE SWEDISH PLEASURE CRAFT XANTE XANTE CALLSIGN SEVEN SIERRA YANKEE SEVEN FOUR ONE ZERO PLEASURE CRAFT XANTE THIS IS BREMEN RESCUE BREMEN RESCUE		
Fine-tuned on other data			
LARGE_10m	BREDERS LESERAR CRAFT SANTR TDIS CLEVERAR CRAFFT SANTERS SANTERS TCOURLTINE SEVEN GERART YANCKE SEVEN FORE WON SIRO LEIVEAR CROFT SANTERS SISERS BUT AM AN RESCKUE O TAM AN RESCIU	100	45
LARGE_960h	EDIS CLERARKRAFT SANTER WEDISH CLERARKRAFT SANTERS TANTERS SOLTINE SEVEN GERAR YANKEE SEVEN FOUR ONE ZERO CLERARKRAFT SANTER TISUS UTAM AND RESCUE OF AM AND RESCUE	77	36

Fine-tuned on domain data			
JRCC		100	100
LARGE	SWEDISH PLERARPRAFT SANT SWEDSH PLEARARCRAFT SANTE SANTET CALSINE SEVEN GERRA YANKCE SEVEN FOUR ONE ZERO SLERARCRAFT SANTE THIS IS RIMEN RESCU AMUN RESCU	69	21
LARGE_VOX	SWEDISH SLERACRAST SANTE SWEDISH SLETRACRAS SANTXSANT CALLSIN SEVEN KERA GENK SEVEN FOUR ONE ZERO SLERRACRASTSANT THIS IS RIMN RESCUO F EMAN RESCUEO	69	28
Swedish-English			
XLSR-53	AA N	100	97

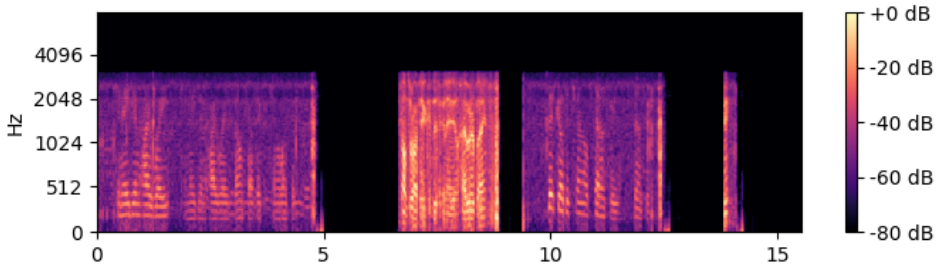


Model	Transcription	WER	CER
Reference	MAYDAY I DELTA KILO TWO EIGHT SIX THREE SKALDET THIS IS LYNGBY RADIO ARE YOU STILL ON CHANNEL ONE SIX		
Fine-tuned on other data			
LARGE_10m	MAY DAY ENE TELFUCCELOW TOUW AT SIX TO THEREE SIALNERS IS HIS LIGBLUR RADIOUW I OUD TESPELL ON CHAT WAN SIX	85	49
LARGE_960h	MAY DAY KI TELTA KILLO TWO EIGHT SIX THREE SCALNES WIS IS LINGE RADIO IOD SPILL ON CAL ONE SIXTH	65	24
Fine-tuned on domain data			
JRCC		100	100
LARGE	MAA KI TELA KILO TWO EAIGH SIX THREESGUARNER THIS IS LYNGBY RADIO YOT SIL ON CHANNEL ONE SIX	45	20
LARGE_VOX	MAIDA K DELFHA KILO TWO AIH SIX THREESCULNE THIS IS LYNGBY RADIO Y O STIL ON CHANNEL ONE SIX	45	18
Swedish-English			
XLSR-53	STEN	100	96



Model	Transcription	WER	CER
Reference	SECURITE SECURITE SECURITE ALL STATIONS ALL STATIONS ALL STATIONS THIS IS SWEDEN TRAFFIC WITH GALE WARNING AND REPETITION OF NAVIGATIONAL WARNINGS PLEASE LISTEN TO MEDIUM FREQUENCIES OR VHF TRAFFIC CHANNEL		
Fine-tuned on other data			
LARGE_10m	SECULATEYA SECULITEYOF SECULATEIOME ALL STACION SOLL STACHIONSALE STACHIONES THISS IS SWEDAN TRAFIK WETE GAL WORNING AD REPEITITION ON MARGATION MNORNINGS PLECE LISSEN TO MEADIOME FREQUENYES OR RECHREST TRAFFICK CHANELS	87	30
LARGE_960h	SECURITY A SECURITY OF SECURITY ALL STATIONS OLL STATIONS OL STATIONS THIS IS SWEDEN TRAFFIC WITH GILWONING A REPETITION ON MALIGATION WARNINGS PLEASE LISTEN TO MEDIUM FREQUENCIES OR THECHESS TRAFFICTIONS	50	18

Fine-tuned on domain data			
JRCC		100	100
LARGE	SECURITE SECURITTE SECURITE ALL STATIONS ALL STATIONS ALL STATIONS THIS ISS SWEDEN TRAFFIC WITH GALE WARNING AND REPETITION OF NAVIGATIONAL WARNINGS PLEASE LISTEN TO MEDIUM FREQUENCIE	26	13
LARGE_VOX	SECURITE SECURITE SECURITE ALL STATIONS ALL STATIONS ALL STATIONS THIS IS SWEDEN TRAFFIC WITH GALE WARNING AND REPETITION OF NAVIGATIONAL WARNINGS PLEASE LISTEN TO MEDIUM FREQUENCIES OR VHF TRAFFIC CHANNELS	3	0
Swedish-English			
XLSR-53	ECRIIEEEEEL TONS AS TIONSASLSSSTTTIITON NAON HSSTHSHIS IS IS ISSESESEAINANANANANP ESEL LISSTEN TOS	96	69



Model	Transcription	WER	CER
Reference	CANADIAN HIGHWAY CANADIAN HIGHWAY SOUND TRAFFIC SOUND TRAFFIC THIS IS CANADIAN HIGHWAY REPLYING PLEASE GO TO CHANNEL SEVEN THREE SEVEN THREE THREE		
Fine-tuned on other data			
LARGE_10m	CANADIAN HIY WAY ANADIOAS HIYWAY SOMNE SOUBEFECT SANALONSIFONTRI TIK OF THI GL O SOER PLILIT GOUT HIS CHANALE SATLD S SAENIBIC	95	59
LARGE_960h	CANADIAN HIGHWAYS CANADIAN HIGHWAYS DONT SOTIXSIMILONTIS TOTDEROTHICK IN THIS CANADIAN HIGWAR BA SLIT GOTI CANAL SEDENTIS STENSE B	86	48
Fine-tuned on domain data			
JRCC		100	100
LARGE	CANADIO HI WEA CANAION HI WA LOUNCASEX CHANNL ONE SIXSOMNTORASHIXK PLIS CANNADIOE HYE REPLAYING PLEASE GO TO CHANNEL SEVEN THREE SEVEN SIX	72	43
LARGE_VOX	CU NAIN HIWEY CAUNAIO HI WE CAUN TASEX CHANNEL ONE SIX OMNTOROASHIK THIAS CANADION HIWER REPLYN PLEASE GO TO CHANNEL SEVEN THREE SEVEN SRIX	81	43
Swedish-English			
XLSR-53	O EN	100	95