Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Dept. of Information Security and Communication Technology

Marit Kristine Moe

Post-processing Automatic Speech Recognition Transcriptions: A Study for Investigative Interviews

Master's thesis in Communication Technology and Digital Security
Supervisor: Kyle Porter
Co-supervisor: Thomas Beka
June 2023

NTNU
Norwegian University of
Science and Technology

Marit Kristine Moe

Post-processing Automatic Speech Recognition Transcriptions: A Study for Investigative Interviews

Master's thesis in Communication Technology and Digital Security
Supervisor: Kyle Porter
Co-supervisor: Thomas Beka
June 2023

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Dept. of Information Security and Communication Technology

NTNU
Norwegian University of
Science and Technology

Title: Post-processing Automatic Speech Recognition Transcriptions: A Study for Investigative Interviews

Student: Moe, Marit Kristine

Problem description:

The master's thesis will be concerning post-processing of Automatic Speech Recognition (ASR) transcriptions for Norwegian. The objective is to be able to improve the readability and quality of ASR transcriptions by applying different approaches, such as Inverse Text Normalization (ITN) and speaker diarization, to convert text output from *oral text form* to *written text form*. Different algorithms for post-processing of ASR transcriptions already exist for other languages than Norwegian. Therefore, the plan is to adapt these algorithms for Norwegian and test them on Norwegian data extracted from Språkbanken. The motivation for the project is the Norwegian Police's interest in improving and optimizing the transcription process for investigative interviews. AI4INTERVIEWS is a project currently working on this task.

Approved on: 2023-02-23

Main supervisor: Porter, Kyle, NTNU

Co-supervisors: Beka, Thomas, Politiets IT-enhet and
Giri, Charul, UiA

Abstract

The Norwegian Police is interested in creating an efficient transcription process for investigative interviews. Today, the transcription is done manually, or just a summary is written. The hope is to be able to improve the readability and quality of Automatic Speech Recognition (ASR) transcriptions by converting them from oral to written form, so they can be adapted by the Police to make the transcription process less manual.

In this master's thesis capitalization of proper nouns, adding of commas, and conversion of numbers and abbreviations to the written domain are investigated. For the capitalization and adding of commas, a new approach utilizing Named Entity Recognition is tested. For the conversion of numbers and abbreviations, an Inverse Text Normalization approach is considered utilizing the Norwegian Parliamentary Speech Corpus's standardization scripts.

The evaluation shows a slight improvement of 1.38% for the average Word Error Rate (WER) and 1,69% for the average Character Error Rate (CER) of the Exhibition Corpus. For the text in the dataset with the most proper nouns, the WER was improved by almost 10%. While the text with the most numbers in it got an improvement of over 13% CER. The readability of the transcriptions is improved, but if the ASR model and the post-processing sequence should be applied by the Police, a manual correction must be expected.

Sammendrag

Politiet er interessert i å effektivisere transkripsjonen av avhør. I dag transkriberes avhør manuelt eller kun et sammendrag blir skrevet. Håpet er å kunne forbedre leseligheten og kvaliteten av automatisk talegjenkjennings transkripsjoner ved å konvertere dem fra muntlig til skriftlig format slik at de kan bli anvendt av Politiet for å gjøre prosessen mindre manuell.

I masteroppgaven vil det bli undersøkt hvordan man kan legge til store forbokstaver i egennavn, legge til komma til lister av egennavn og konvertere tall og forkortelser til det skriftlige domenet. For å kunne legge til stor forbokstav og komma til lister, blir en ny fremgangsmåte ved å benytte seg av en modell for navnegjenkjenning utprøvd. For å konvertere tall og forkortelser blir en invers tekstnormaliseringstilnærming vurdert ved å ta i bruk Stortingskorpusets standardiseringsskript.

Resultatene viser en svak forbedring på 1,38 % gjennomsnittlig ordfeilrate og 1,69 % gjennomsnittlig tegnfeilrate på Utstillingskorpuset. Den teksten med flest egennavn fikk en forbedring på nesten 10 % ordfeilrate, mens den teksten som inneholdt flest tall ble forbedret med 13 % tegnfeilrate. Leseligheten til transkripsjonene ble forbedret, men dersom Politiet vil anvende den automatiske talegjenkjennings modellen og etterbehandlingsstegene fra masteroppgaven må det påregnes en manuell korrigering.

Preface

This master's thesis completes my degree in the 5-year master's degree program Communication Technology and Digital Security at the Norwegian University of Science and Technology. I am grateful for the opportunity I was given to explore a new field of study with investigations, ASR, and post-processing methods in this master's thesis. I have learned a lot, even if the results were not revolutionary. I want to thank my two supervisors Kyle Porter and Thomas Beka for giving me important feedback and guidance this last year.

Contents

Appendix

List of Figures

List of Tables

List of Algorithms

List of Acronyms

AI Artificial Intelligence.

ASR Automatic Speech Recognition.

CER Character Error Rate.

DNN Deep Neural Network.

FASR Forensic Automatic Speaker Recognition.

GMM Gaussian Mixture Model.

HMM Hidden Markov Model.

ITN Inverse Text Normalization.

MER Match Error Rate.

MFCC Mel Frequency Cepstral Coefficient.

NER Named Entity Recognition.

NLP Natural Language Processing.

NN Neural Network.

NPSC Norwegian Parliamentary Speech Corpus.

PER Phoneme Error Rate.

RQ Research Question.

SER Sentence Error Rate.

WER Word Error Rate.

WFST Weighted Finite State Transducer.

Chapter 1 Introduction

1.1 Overview

In this introductory Chapter, the motivation and objectives for the master's thesis will be presented. The necessary theory to understand the experiments carried out in connection with this master's thesis will be presented in Chapter 2. Related work will be presented in Chapter 3. In Chapter 4 the methodology used for the thesis is described. The results of the experiments will be presented in Chapter 5 and discussed in Chapter 6. Finally, a conclusion will be given in Chapter 7.

1.2 Motivation

Material for the motivation was identified in the preliminary project preceding this thesis [Moe22], some of the material is therefore reused in this Section of the thesis. The material has been restructured and additional paragraphs about transcription have been added.

The Norwegian Police is interested in creating an efficient transcription process for investigative interviews. Currently, the investigative interviews are transcribed manually. While the accuracy of Automatic Speech Recognition (ASR) has improved in recent years, some ASR models output text in only lowercased letters. By improving the readability and quality of such ASR transcriptions the hope is to make the transcription of investigative interviews less manual. Etterforskningsløftet (the Investigation Reform) which was initiated by Nærpolitireformen (the Police reform) in 2015 describes several objectives for the criminal proceedings, for instance, that the proportion of criminal cases that are solved should be increased and that the processing time should be reduced [Pol16]. Improving the readability and quality of ASR transcriptions may contribute to both.

Creating a more efficient transcription process can save the Police valuable time that can be used on other responsibilities instead. For instance, Police officers on

patrol can use the time to increase their presence in society. This is positive since the government believes that visibility and local presence are essential for maintaining trust in the Police [Jus20]. An increase in the efficiency may also contribute to the "Golden Hour", and may make the transcriptions available earlier. The "Golden Hour" is the earliest hours in an investigation when it is crucial to collect data and evidence to secure relevant and significant material [MAHW13]. Further, this may contribute to an earlier clarification of the investigation and perhaps lead to solving crimes faster. Today, cases can be more extensive and complicated than before, because a perpetrator may be responsible for many hundreds of victims of online assault, fraud, and other forms of digital crime [Jus20]. Therefore, there is a potential that solving a case can prevent a lot of new crimes. The trust in the Norwegian Police may also be strengthened. To maintain trust, the population needs a Police force that can handle crime professionally and efficiently [Jus20]. High trust in the Police, contributes to increased social security, both centrally and in the local communities, because trust is necessary for the population to share relevant information and cooperate with the Police [Jus20].

With the help of Laboratoire d'Informatique de l'Universite du Maine's ASR system based on CMU Sphinx in 2008, Bazillon *et al.* [BEL08] carried out some experiments to compare manual transcription with assisted transcription. The audio they experimented with was both prepared speech and spontaneous speech, where the advantage of assisted transcription turned out to be the greatest for prepared speech [BEL08]. For a 10-minute prepared speech audio file, the transcriber needed approximately 83 minutes to transcribe it, assign speakers, and correct spelling, while the transcriber only needed 40 minutes with the assistance of the ASR system [BEL08, p. 2]. It was thus a saving of half the time to get assistance with the transcription by the ASR system. It will be possible to save time by introducing an ASR system to the Police. In addition, there has been major development on the ASR front since 2008.

In 2019 Kim *et al.* [KLC+19] compared transcriptions from two human transcribers and five different ASR systems. In the paper, they concluded that ASR systems are significantly faster than human transcription [KLC+19, p. 9]. While the ASR systems used approximately 15 minutes to process a 15-minute video, the two human transcribers used approximately 1 hour to process the same video (this included starting and ending timestamps of sentences) [KLC+19, p. 9]. This indicates a fourfold increase in processing speed from the ASR systems to the human transcribers. Another consideration when choosing an ASR system is the accuracy of the system. The authors concluded that the human transcribers were the most accurate with an average WER of 17.4%, while the most accurate ASR system tested was YouTube Captions services with an average WER of approximately 28%. While both these values are relatively high to be WERs, the authors emphasize that the

medical terminology in the data used may have contributed to this.

In addition to making the transcription process more efficient, there is also a desire to standardize transcriptions. Meaning to ensure consistency between different investigative interviews, but also reducing the possible variability and randomness that an ASR algorithm may output. Accuracy is important for the Police to be able to adopt an ASR model for the transcription of investigative interviews. The purpose of the interrogation report from the interview is to provide the legal apparatus with a version of the witness's statement and shall serve as basis in a criminal case context [Eri13]. The interrogation reports are often central to assessments and decisions, and they can be presented to the court several times [Eri13]. In addition, the interrogation reports are read and used beyond police activities [Eri13].

Lastly, a successful post-processing sequence will possibly contribute to the Norwegian speech-to-text community. Norwegian is a low-resource language, and therefore not many tools for the post-processing of Norwegian ASR transcriptions exist yet. Technology is evolving, and it is important that Norwegian society also follows the digital development. An improvement in the quality of Norwegian ASR transcriptions may be useful in multiple parts of society, some usecases of ASR can, for instance, be the transcription of meetings for various businesses, doctors dictating medical records, and automatic transcriptions of television shows for people with impaired hearing.

1.3 Objectives

The objective of the master's thesis will be to *investigate to which extent it is possible to convert Norwegian text output from an ASR algorithm from oral text form to written text form* [Moe22]. Some examples of phrases that can be post-processed are displayed in Table 1.1. The phrases are sorted into different categories. In the table, the phrase before and after post-processing is called oral and written form respectively.

Two of the Research Questions (RQs) from the preliminary project in TTM4502 are maintained [Moe22]:

RQ1 How can Inverse Text Normalization (ITN) be adapted to convert numbers, monitory values, dates, and abbreviations to the proper written format for the Norwegian language?

RQ2 To what extent can we improve sentence structure in ASR transcriptions by improving punctuation, the spelling of compound words, abbreviations, and capital letters in proper nouns?

The focus will be on converting numbers, abbreviations, and proper nouns from an ASR transcription to a suitable format. Furthermore, investigate if these changes improve the accuracy of the transcription. As well as investigating if the ASR process will be of benefit to the Police in an investigative interview setting.

Originally, as mentioned in the problem description, speaker diarization was one approach to be explored in the thesis. It was chosen not to investigate speaker diarization for Norwegian anyway because no appropriate open-source dataset was found for this purpose. A solution could have been to make a new dataset by merging files in an existing dataset so that the audio files include different speakers and some overlap in the speech. Because the speech from the different speakers would not have been connected and had a context like normal conversations, the audio would not have been representative of a dialog in for instance an investigative interview setting.

Category	**Oral form**	**Written form**
Numbers	nitten år	19 år
	en til åtte	1-8
	femten komma to prosent	15,2 %
	åtte og tjuende april	28. april
Currencies	tre dollar	$3
	hundre og femti kroner	150 kr
Abbreviations	og liknende	o.l.
	blant annet	bl.a.
	kilo gram	kg
	centimeter	cm
	fra og med	f.o.m.
	minutter	min
	cirka	ca.
	for eksempel	f.eks.
	et cetera	etc.
	og så videre	osv.
	i forhold til	ift.
Placenames	trondheim	Trondheim
	norge	Norge
Names	marit	Marit

Table 1.1: Examples of phrases before and after post-processing.

1.4 Contributions

The main contribution of this thesis is the new approach of utilizing a Named Entity Recognition (NER) model to capitalize proper nouns and add commas to an ASR transcription. Additionally, qualitative results are obtained for the utilization of normalization scripts from the Norwegian Parliamentary Speech Corpus (NPSC) as a tool for the ITN task. The evaluation of the final sequence of post-processing steps shows a slight improvement of 1.38% WER and 1.69% Character Error Rate (CER). In addition, the two texts with the most proper nouns and numbers in them achieved an improvement of nearly 10% WER and 13% CER respectively.

Chapter 2
Background

This Chapter will contain the most relevant background theory on investigations, ASR, and post-processing of ASR transcriptions. In Section 2.1 a description of a criminal investigation will be given alongside information about the KREATIV method used by the Police and some previous applications of Artificial Intelligence (AI) in investigations. Section 2.2 is a short Section about the written standards of Norwegian. In Section 2.3 traditional and modern end-to-end ASR architecture are described and some evaluation metrics typical for ASR are presented. Lastly, Section 2.4 provides information about post-processing methods to be explored in this thesis and some evaluation metrics.

2.1 Investigations

Andersen [And19, p. 16] defines criminal investigation to be: a "systematic inquiry and examination to determine whether an incident constitutes a crime, and to sufficiently discover, describe and document the particulars of the incident in order to prepare for a legal resolution". Fahsing [Fah16, p. 4] states that the main purpose of the criminal investigation is to determine how, where, when, why, and by whom a crime was, or will be, committed. In order to do this an investigator needs to collect, process, and present information to explain the circumstances of the possible crime [And19, p. 4]. One way of gathering information is through investigative interviews.

An interrogation report is a written rendering of the investigative interview [Eri13, p. 1]. The report should be a summary of the information that emerges in the interview, which can contribute to the investigation and decision of whether to prosecute [Eri13, p. 2]. Today, dialog reports are made by transcribing investigative interviews manually either fully or partially, or only a summary is written as a report [AI4]. The reports are used by the Police to communicate with various people connected to the handling of the criminal case [Eri13, p. 2], for instance, investigators, defense attorneys, assistance lawyers, other experts, and higher prosecution authorities [Eri13,

p. 16]. The reports are often central to assessments and decisions, and they can, among other things, be presented to the court several times [Eri13, p. 2].

2.1.1 KREATIV

The Norwegian Police use the training program and interrogation methodology KREATIV whose values and principles are reflected in the acronym: communication (Kommunikasjon), legal certainty (Rettssikkerhet), ethics (Etikk) and empathy (Empati), active awareness raising (Aktiv bevisstgjøring), trust through transparency (Tillit gjennom åpenhet), and scientific grounding (Vitenskapelig forankring) [Rik13, p. 8]. The methodology was a settlement with outdated and confession-focused interrogation methods [JSL18]. Overall, KREATIV is about the interviewer gathering information and being open to what may have happened because the purpose of an investigative interview is to obtain as much reliable and relevant information as possible. To do this the interviewer asks open-ended questions. The interviewee speaks freely and without interruptions before the interviewer begins to ask more in-depth questions or confront other information [Eri13, p. 12].

2.1.2 Application of AI in Digital Forensics and Investigations

AI is not a new topic for investigations and digital forensics. It exists a variety of different applications of AI to different areas of digital forensics. In the paper of Du *et al.* [DHS+20] existing work in the application of AI to specific areas of digital forensics is shown. Some areas that already make use of AI mentioned in the paper are data discovery, network traffic analysis, and multimedia forensics [DHS+20]. Applications of AI techniques in network traffic analysis for instance help to automate the detection of crimes and filter out redundant information [DHS+20, p. 3]. In multimedia forensics, AI techniques have started to be used to, among other things, uncover forgeries in images [DHS+20, p. 6]. Another application is mentioned in Vajpai and Bora [VB16], automatic forensic voice comparison systems called Forensic Automatic Speaker Recognition (FASR) [VB16, p. 91]. FASR is used to identify speech samples of a suspected speaker for juridical and law enforcement purposes [VB16, p. 91].

2.2 Norwegian

The Norwegian language has two official written standards Bokmål and Nynorsk. Bokmål is the most widespread one with approximately 85% of the population using this standard. In addition, the Norwegian language has many spoken dialects, but no standard dialect.

2.3 Automatic Speech Recognition

ASR is an interdisciplinary activity that converts audio signals into readable text. Some subject areas that are particularly important for ASR are linguistics, computer science, machine learning, and signal processing. An example of an ASR system is Wav2vec 2.0 [BZMA20] which will be presented in Section 2.3.2. Other solutions will be presented in Chapter 3.

According to Yu *et al.* [YD16, p. 4], an ASR system usually consists of four main components: signal processing and feature extraction, acoustic model, language model, and hypothesis search. This is the traditional architecture of ASR systems, an illustration of the components is shown in Figure 2.1. In recent years, ASR systems have changed to become end-to-end models using deep learning [WWL19].

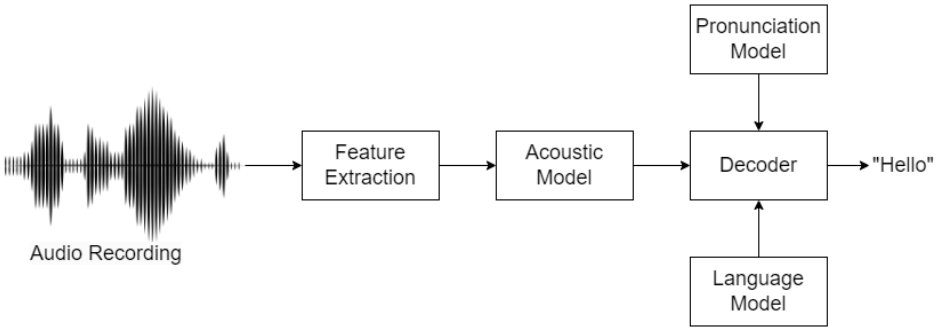


Figure 2.1: Illustration of a traditional ASR pipeline, where the audio signal for "hello" is converted to text. The figure is inspired by a figure shown in this blog [MCC19], but the Pronunciation Model is added.

2.3.1 Traditional ASR Architecture

Historically, speech recognition has been cast as a statistical optimization problem. The ASR system seeks the most likely word sequence W of a given sequence of acoustic observations O [WDMH17, p. 299]. To maximize the posterior probability $P(W|O)$ we can apply Bayes' theorem, ignore the marginal probability of $P(O)$, and end up with Equation 2.1, the "fundamental equation" of speech recognition [WDMH17, p. 299]. The first component of Equation 2.1, $P(O|W)$, is computed using the acoustic model, while the second component, $P(W)$, corresponds to the probability from the language model [WDMH17, p. 300]. The output of an ASR system will be $\hat{W}$.

$$\hat{W} = argmax_W P(O|W)P(W) \tag{2.1}$$

Signal Processing and Feature Extraction

In the first component of a traditional ASR system noise and channel distortions are removed, the signal is transformed into the frequency domain, and salient feature vectors suitable for the acoustic model are extracted [YD16, p. 4]. The most commonly used features for ASR are the Mel Frequency Cepstral Coefficients (MFCCs) [KLW19, p. 372]. They are able to perform similar types of filtering as the human auditory system and have low dimensionality [KLW19, p. 372].

Speech is a non-stationary signal, which means that its statistical properties change over time. The signal is therefore examined in chunks, to ensure that the speech can be assumed stationary within the chunks [KLW19, p. 373]. Typically, a window length of 20ms is used combined with a 10ms overlap [KLW19, p. 373].

The Acoustic Model

The task of the acoustic model is to take the feature vectors from the signal processing and feature extraction component as input and convert it to a probability of phonemes or characters based on knowledge of phonetics [YD16, p. 4]. In linguistics, a phoneme is one of the smallest units of speech distinguishing one word from another [Cam]. For instance, the vowel in the word “pin” is the phoneme /I/ and it is the only thing distinguishing “pin” from “pan” which has the phoneme /æ/ in the middle of the word [Cam]. Different languages have different sets of phonemes [RW01, p. 10]. The English language has for instance 44 phonemes [Cho20, p. 658].

Traditionally, the conventional acoustic model has been the Hidden Markov Model (HMM)-Gaussian Mixture Model (GMM) [YD16, p. 6]. But as stated in the preliminary project [Moe22], recently, hybrid models combining HMMs and Deep Neural Networks (DNNs), and deep-learning end-to-end models have become more popular [WWL19].

In the hybrid approach, the GMMs is replaced with deep learning to predict phonetic states based on the acoustic observations [KLW19, p. 537]. A DNN cannot provide the conditional probability $P(O|S)$ directly [KLW19, p. 389]. Instead, the probability $P(O|S)$ is turned into a classification problem $P(S|O)$ using the framewise posterior distribution, and the pseudo-likelihood is applied as an approximation of the joint probability [KLW19, p. 389].

In addition to the acoustic model that maps the acoustic features to phonetic states, a lexicon or pronunciation model is used to compute the probability of a sequence of phonetic states given a word sequence [KLW19, p. 384]. The lexicon can be expressed as $P(S|W))$ [KLW19, p. 384].

The Language Model

The language model $P(W)$ in a traditional ASR system assigns probabilities to hypothesized word sequences [YD16, p. 4]. To be able to estimate the probabilities the model has to learn the correlation between words from training data [YD16, p. 4]. The model wants to compute the relative frequencies of different word sequences, but with a large vocabulary, the total number of unique sequences can grow exponentially with the length of the sequence [Cho20, p. 659]. It is therefore impossible to parameterize the language model by listing the probability of every possible word sequence. Instead, the chain rule is used to factor the word sequence probability into a product of conditional word probabilities, and the Markov assumption is applied to limit the number of states [WHK+17]. N-gram models are typically used where each word is conditioned only on the previous N-1 words [KLW19, p. 107].

Hypothesis Search

The hypothesis search combines the probabilities from the acoustic model, lexicon, and language model and outputs the sequence of words with the highest likelihood [YD16, p. 4].

2.3.2 End-to-end ASR Architecture

Both the traditional statistical approach and the newer hybrid models rely on training each component of the model separately [KLW19, p. 537]. This can lead to sub-optimal results because of the lack of error propagation between the models [KLW19, p. 537]. Furthermore, models may become sensitive to noise and speaker variation [KLW19, p. 537]. By transitioning to end-to-end models, the ASR model can learn from the data directly instead of depending on heavily engineered features [KLW19, p. 537]. The training data pairs may be the speech utterance and the linguistic representation of the transcription [KLW19, p. 537].

The end-to-end models try to optimize the posterior probability $P(W|O)$ directly, instead of separating it [KLW19, p. 537]. The most important benefits from the joint optimization of all the components of the ASR system together are lower complexity, faster processing, and higher quality [KLW19, p. 538].

Wav2vec 2.0

Wav2vec 2.0 is a framework for self-supervised learning of speech representations and is presented in the paper of Baevski *et al.* [BZMA20]. Compared to other ASR systems, Wav2vec 2.0 uses a small amount of labeled data [BZMA20]. This is an advantage for low-resource languages or different dialects because it is no need to collect a big dataset with labeled data, which can be difficult [BZMA20, p. 9].

The Wav2vec 2.0 framework consists of three main components: a feature encoder, a context network with a Transformer architecture, and a quantization module [BZMA20, p. 2]. First, raw speech audio is converted to latent speech representations in the multi-layer convolutional feature encoder [BZMA20, p. 2]. The latent speech representations are fed into a transformer network to build representations that capture information from the whole sequence [BZMA20, p. 2]. The latent speech representations are also discretized with the quantization module to represent the targets in the self-supervised objective [BZMA20, p. 2].

Language models used by the framework are a 4-gram model and transformers [BZMA20, p. 5].

2.3.3 Evaluation Metrics for ASR

The WER is the most commonly used evaluation metric for ASR [KLW19, p. 387]. As also stated in the preliminary project [Moe22], the WER is the proportion of word errors to words processed [MMG04, p. 1]. The metric measures the edit distance between a manually written transcript (the reference) and the prediction the ASR system makes by considering the number of insertions, deletions, and substitutions [KLW19, p. 387]. The definition of WER is given in Equation 2.2 where S is the number of substitutions, D is the number of deletions, I is the number of insertions, H is the number of hits (correctly predicted words), and N is the total number of words in the manual transcription (sum of hits, substitutions, and deletions) [MMG04, p. 1].

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

An alternative for WER is the Match Error Rate (MER), the probability that a given match is incorrect [MMG04, p. 3]. Equation 2.3 gives the definition of MER, where M is the sum of the number of hits, substitutions, deletions, and insertions [MMG04, p. 3]. The MER can be used when the information in the transcriptions is more important than the edit distance [KLW19, p. 388]. Unlike WER, MER provides probabilistic interpretations because the metric has values between 0% and 100% while the WER can be larger than 100% [KLW19, p. 388].

$$MER = \frac{S + D + I}{M} \tag{2.3}$$

Other possible metrics are CER, Phoneme Error Rate (PER), and Sentence Error Rate (SER). These can be used for evaluating character-based models, phoneme-based models, or to evaluate the system by looking at each sentence as a whole [Cho20, p. 388][YD16]. The CER can be an interesting alternative to the WER because it is less punishing for single-letter changes. Meaning if only one character

in one word is different between two sentences the WER will say that a whole word is wrong, while the CER only see one wrong character.

2.4 Post-processing of ASR Transcriptions

Without post-processing, the ASR output from the Wav2vec 2.0 framework is only given in lowercase letters. Different types of post-processing can be applied to increase the structure and the readability of the transcription so that there will be less manual work to correct the transcription afterward. Possible post-processing steps that can improve the readability of ASR transcriptions or standardize the output are for instance: differentiating between speakers in the audio file, text styling by adding punctuation and capitalizing names, or making sure that numerical values and abbreviations are presented in an appropriate format.

The post-processing methods to be explored in this master's thesis will be presented in the following Subsections. The methods include ITN and NER.

2.4.1 Inverse Text Normalization

As also stated in the preliminary project [Moe22], one post-processing step that can be applied to an ASR system is called ITN, which with the intention of converting *oral text form* output from the ASR system to *written text form* transcriptions, improves the readability of the transcription. Normalizing text means converting it to a more convenient, standard form [JM18]. ITN is often applied to transcriptions to convert numbers, dates, monitory values, and abbreviations to the suitable written format.

Weighted Finite State Transducers (WFSTs) are often used in ITN tools for production [ZBGG21, p. 1]. The main reason for rule-based WFST still being used is the low tolerance against unrecoverable errors that can occur with Neural Network (NN) based ITN systems [GCD+22, p. 1][ZBGG21, p. 1]. Unrecoverable errors are errors that can for instance change the meaning of the text, in other words, errors that change the semantics of the input [ABG22]. An example given by Zhang *et al.* [ZBGG21] is the conversion of "one hundred and twenty three dollars" to "$23". A WFST is a graph with an input label, output label, and weight on each edge. A path through the transducer encodes a mapping from an input label sequence to an output label sequence [MPR02, p. 69]. Now, hybrid models that combine NN with lightweight grammars are also available [ZBGG21, p. 1]. They can provide better scalability than the WFST based models, but at the same time be more adaptable and robust than a complete NN based ITN model [GKX+23, p. 238].

2.4.2 Named Entity Recognition

NER is not originally designed for ASR post-processing purposes, but aims to locate and categorize important nouns and proper nouns in a text [Moh14, p. 221]. Examples of classes that are typical for a NER system are: person (PER), organization (ORG), and location (LOC) [Moh14, p. 222]. NER can serve as a useful source of information for Natural Language Processing (NLP) applications, for instance, machine translation and information retrieval [Moh14, p. 222]. If for instance, a NER model gets the sentence "Marit studies at NTNU's campus in Trondheim", it will be expected to output three tokens. Typically, the model will be able to identify that "Marit" is the name of a person, "NTNU" is an organization, and "Trondheim" is a location.

2.4.3 Evaluation Metrics of Post-processing

WER is often used to measure the performance of ITN systems [SSBK21, p. 7575]. Sunkara *et al.* presents two additional metrics: ITN-WER (I-WER) and Non-ITN WER (N-WER) [SSBK21]. I-WER differs from normal WER in that the denominator K in the calculation consists of only the number of words from the reference transcription that is normalized [SSBK21, p. 7575]. Meaning the number of words that have the potential to be converted to a standard form in the hypothesis transcription. The numerator is the same as for normal WER in Equation 2.2, the sum of substitutions, deletions, and insertions between the hypothesis and reference transcription. I-WER is shown in Equation 2.4. Similarly, the N-WER is computed over the words from the reference transcription that should not be normalized, instead of the number of words in the reference.

$$I - WER = \frac{S + D + I}{K} \tag{2.4}$$

NER systems are evaluated by comparing the results from the system to a human-labeled reference. The most common metrics for evaluating NER are *precision* and *recall* [Moh14, p. 228]. To measure this only three numbers are needed C: number of correctly labeled named entities, L: number of labeled named entities, G: number of human-labeled (gold-standard) named entities [Moh14, p. 228]. Precision and recall are given in Equation 2.5 and Equation 2.6 respectively.

$$Precision = \frac{C}{L} \tag{2.5}$$

$$Recall = \frac{C}{G} \tag{2.6}$$

Precision measures how precise or accurate the labeled named entities are, while recall measures the coverage of the NER system [Moh14, p. 228]. The two metrics can also be combined to the F_1 measure. F_1 is shown in Equation 2.7.

$$F_1 = \frac{2 * Precision * Recall}{Precision + Recall} \tag{2.7}$$

Chapter 3
Related Work

This Chapter will show previous studies with the same topics as this thesis. This includes some previous applications of ASR in investigations, results from Norwegian ASR, and both Norwegian and foreign papers on post-processing of ASR transcriptions.

3.1 Application of ASR for Investigations

At least two speech recognition systems are available for use in law enforcement. The first one: Dragon Law Enforcement, is a speech recognition software developed by Nuance [Nua]. While the second one: Philips SpeechLive, is a speech recognition system available in the cloud. Port Huron Police Department, Michigan, USA is one Police Department that uses the Philips SpeechLive solution [Phi].

Negrão and Domingues [ND21] present the SpeechToText plugin, a digital forensic tool that utilizes open-source machine learning software for automatic detection and transcription of voice recordings [ND21, p. 9]. They apply ASR to digital forensics to be able to automatically transcribe audio files found in digital devices so that the content can be indexed for searches and included in digital forensic reports [ND21, p. 2]. In the paper, they showed that the plugin performed faster than real-time given adequate hardware [ND21, p. 9]. Faster than real-time means that the elapsed time of the transcription process was faster than the duration of the audio file. Additionally, they showed that the quality of the plugin was sufficient for the purpose of quickly finding relevant information although the WER of 27.2% is high [ND21, p. 8]. The authors explain that the cause of the high WER was probably that none of the speakers were native English speakers and that the audio files were not of high quality [ND21, p. 8].

Vásquez-Correa and Muniain [VÁ23] propose to utilize ASR and keyword spotting to detect the presence of offensive online audiovisual data related to child abuse [VÁ23, p. 14]. They tested two open-source ASR models for this task, Wav2vec 2.0

and Whisper. Depending on the language the models achieved a WER between 11% and 25% [VÁ23, p. 1].

3.2 Related ASR for Norwegian

For Norwegian, there are mainly two released open-source models for the ASR task. These are the Wav2vec 2.0 framework from Facebook AI [BZMA20] and Whisper from OpenAI [RKX+22]. On Hugging Face there are a total of 37 fine-tuned models if you choose the task "automatic speech recognition" and the language "Norwegian" [Hug], all models are either based on Whisper or Wav2vec 2.0.

In addition to the two open source models mentioned above there exists different commercial models for instance Google Speech-to-Text and virtual assistants like Apple's Siri.

Solberg and Ortiz [SO22] present a new dataset the NPSC that aims to improve ASR for Norwegian spontaneous speech and dialects [SO22, p. 5]. Their baseline ASR system is based on DeepSpeech 2 where the acoustic model is combined with a n-gram language model [SO22, p. 4]. In the paper, they tested how the NPSC data could help when transcribing spontaneous, dialectal speech from another dataset: Module 3 of NB Tale, and showed that the model trained on the NPSC dataset was better to transcribe dialects and spontaneous speech than an ASR model only trained on manuscript read text. Transcription of Module 3 of the NB Tale dataset obtained a WER of 48.4% by the baseline model and 37.3% when the acoustic model was fine-tuned with NPSC data [SO22, p. 4].

3.2.1 Wav2vec 2.0

In a recent paper by De la Rosa *et al.* [DBK+23] a new set of baseline ASR models for Norwegian based on the Wav2vec 2.0 framework is developed and evaluated [DBK+23, p. 555]. The best result obtained, with a WER of 4.30% (4.93% without 5-gram language model) was obtained by evaluating the 1 billion parameters model trained on NST-NPSC-Bokmål on the test set of NST [DBK+23, p. 559].

Solberg *et al.* show that ASR models trained on combinations of two datasets (NPSC and Rundcast) perform better on new data than the ASR models trained only on one of the datasets, even when the length of the training data is the same [SOP+23]. Wav2vec 2.0 models were fine-tuned with both datasets and a combination of the two. WER was reported for evaluations of the fine-tuned Wav2vec 2.0 models with the test sets of the two datasets as well as for the NB Tale and the NST corpora. For instance, the Wav2vec 2.0 model just trained on Rundcast obtained a WER of 10.6% (without language model) on the test set of NST, while the long model trained on

both Rundcast and NPSC obtained a WER of 8.6% (without language model) on the same test set [SOP+23]. For the NB Tale Module 3 dataset containing spontaneous speech, the evaluation of the same Wav2vec 2.0 model trained on Rundcast achieved a WER of 28.9% (without language model) while the long model achieved a WER of 23.0% (without language model) [SOP+23].

3.2.2 Whisper

A new ASR system called Whisper [Ope22] published in the autumn of 2022 performs well for Norwegian and provides some normalization. Whisper is a neural net and focuses on weakly supervised pre-training instead of self-supervision and self-training techniques used in recent studies for instance in Baevski *et al.* [BZMA20] [RKX+22]. The paper of Radford *et al.* [RKX+22] demonstrates that training on a large and varied supervised dataset and focusing on zero-shot transfer can significantly improve the robustness of an ASR system [RKX+22]. The zero-shot transfer involves testing the model on datasets the model has not been previously trained on to measure the generalization of the model, meaning how well the model generalizes across domains, tasks, and languages [RKX+22, p. 5]. If the model generalizes well, the model can work reliably without the need for dataset-specific fine-tuning to achieve high-quality results [RKX+22, p. 5].

Whisper is trained on 680,000 hours of multilingual and multitask supervised data collected from the web [Ope22]. The system is able to transcribe as well as translate different languages. 6 different models of Whisper are available, the differences are the amount of data the model is trained on and therefore also how accurate the prediction will be and how fast the model will be. WER results for Norwegian with the different Whisper models are obtained from Radford *et al.* [RKX+22] and shown in Table 3.1.

Model	**WER(%)**
Whisper tiny	62.0
Whisper base	44.0
Whisper small	24.2
Whisper medium	12.9
Whisper large	11.4
Whisper large-v2	9.5

Table 3.1: WER results for the different Whisper models on the Norwegian Fleurs dataset shown in Radford *et al.* [RKX+22].

3.3 Post-processing Norwegian ASR Transcriptions

3.3.1 NER

Jørgensen *et al.* [JAH+19] presents the first publicly available dataset for the NER task for Norwegian. The NorNE dataset, as it is called, distinguishes between eight entity types: person (PER), organization (ORG), location (LOC), geo-political location (GPE_LOC), geo-political organization (GEO_ORG), product (PROD), event (EVT), and derived (DRV) [JAH+19, p. 1].

The AI-lab at The National Library of Norway has fine-tuned the NB-Bert base model for the NER task on the NorNE dataset [NB b]. The NB-Bert base model achieves, according to Kummervold *et al.* [KDWB21], an F_1 score of 91.2 when trained on the NorNE Bokmål dataset and 88.9 with the model trained on the NorNE Nynorsk dataset [KDWB21, p. 7].

Other previous NER results include the work of Johansen [Joh19], Haaland [Haa08], and Johannessen *et al.* [JHH+05]. Johansen uses only four entity types: person, organization, location, and miscellaneous [Joh19, p. 1]. For the combined written form of Bokmål and Nynorsk, he achieves an F_1 score of 86.73, consisting of a precision of 87.22 and a recall of 86.25 [Joh19, p. 6]. The best result achieved by Haaland was an F_1 score of 81.36 [Haa08, p. 95]. Johannessen *et al.* [JHH+05] presents results from the Nomen Nescio project which developed NER systems for the Scandinavian languages: Norwegian, Swedish, and Danish [JHH+05, pp. 91–92]. Three of the developed systems were for Norwegian, one rule-based model based on constraint grammar, one based on maximum entropy, and one model for memory-based learning [JHH+05, p. 92]. The models achieved an average recall of 72%, 60%, and 68%, and an average precision of 52%, 60%, and 68% respectively [JHH+05, p. 98].

3.3.2 The Norwegian Parliamentary Speech Corpus Standardization

When the Language Bank at the National Library of Norway developed the NPSC they ran a correction script to correct common errors in the transcriptions, applied normalization grammars to write numbers, dates, and years in standardized formats, and made a machine translation pipeline that translated between Bokmål and Nynorsk [SO22]. Although these post-processing scripts were developed to help make the different versions of the NPSC only, they are included in the corpus as a reference.

3.4 Foreign Post-processing Work

3.4.1 NER

Mayhew *et al.* [MTR19] have experimented with different amounts of cased and lowercased training and test data to overcome the problem of decrease in performance of NER and Part of Speech tagging models in common lowercased scenarios, for instance, noisy web text or ASR output [MTR19, p. 6256]. They conclude that the most effective method is to use a concatenation of cased and lowercase training data [MTR19, p. 6260]. Kuster *et al.* [KFM21] have tried to reproduce the work of Mayhew *et al.* and ended up with the same conclusion that lowercasing half of the training data provides the best performance [KFM21, p. 5].

3.4.2 ITN

WFST Based

The Nvidia NeMo ITN tool is rule-based and uses WFST to deploy grammars for ITN [ZBGG21, p. 1]. Their text-processing module currently supports eight languages: English, Spanish, Portuguese, Russian, German, French, Vietnamese, and Arabic [NVI]. An example of the type of ITN applied by the Nvidia NeMo tool is available on their website: "on may third we paid one hundred and twenty three dollars" is converted to "on may 3 we paid $123" [NVI]. Here the numbers and the currency are converted to a more readable format. Zhang *et al.* [ZBGG21] report a WER of 12.7% on full sentences of the Google Text Normalization dataset for the tool.

NN Based

The Thutmose tagger of Antonova *et al.* [ABG22] is a single-pass token classifier, a tagging-based approach to the ITN task. It is NN based but is simpler than a seq2seq model [ABG22]. In the paper, they evaluate three metrics: sentence accuracy, WER, and the number of unrecoverable errors [ABG22]. On the English GTN test set the Thutmose tagger with BERT achieves a WER of 3.7%.

Hybrid

Sunkara *et al.* presents a hybrid framework for integrating neural ITN with an FST to overcome common recoverable errors in production environments [SSBK21]. They measure the performance of the framework with WER and I-WER [SSBK21]. One of the results presented in the paper was the performance on ASR output. For the conversational data with the numbers usecase, the FST based ITN model achieved a WER of 5%, the neural ITN model achieved a WER of 4.5%, while the hybrid model achieved a WER of 4.2% [SSBK21].

Gaur *et al.* [GKX+23] introduce a new modeling solution that enables on-device ITN utilizing a transformer-tagger and a WFST. To evaluate the ITN performance they use precision, recall, F_1, and Token Error Rate [GKX+23]. For the speech-to-text evaluation in the paper, their model obtained a F_1 score of 73% for ASR output on oral form, while an end-to-end ASR model trained to display written style text directly obtained a F_1 score of 63% [GKX+23].

Tan *et al.* [TBK+23] propose a four-in-one approach to ITN, capitalization, punctuation, and disfluency removal. The disfluency removal includes removing interruptions such as false starts, corrections, repetitions, and filled pauses from the transcription [TBK+23]. The four-in-one approach consists of a single transformer tagging model and WFST grammars [TBK+23]. The metrics used in the paper are similar to the ones in Gaur *et al.* [GKX+23]: word-level precision, recall, and F_1 [TBK+23]. For instance, their model achieved a F_1 score of 88% on the test set of CNN stories, and a F_1 score of 67% on a test set of ASR output for NPR Podcasts [TBK+23].

Chapter 4
Methodology

In this Chapter, the methodologies used in the thesis to answer the RQs will be presented. Figure 4.1 is made to show the overall process to experiment with different post-processing models. The process starts by finding a dataset containing both text and speech. Then the speech data can be transcribed by the chosen ASR model, and the initial metrics can be measured. The intent is to iterate and implement different post-processing steps to improve the structure and readability of the transcriptions and answer the RQs. Post-processing steps are evaluated individually and in sequence. To evaluate the various post-processing steps the WER will be the most important metric, but CER will also be measured.

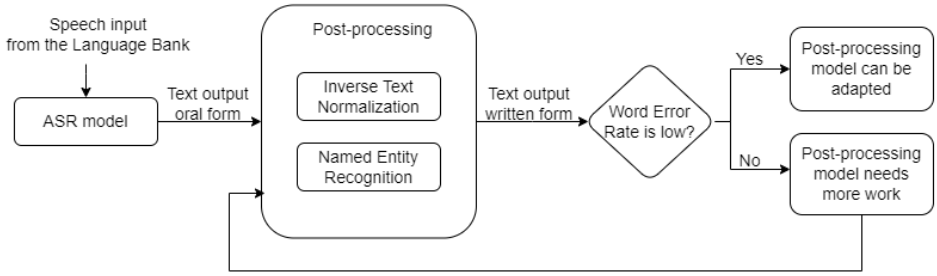


Figure 4.1: Simplified diagram showing the process from speech data to transcriptions and verification of the post-processing algorithms [Moe22].

4.1 Data Description

The main focus of the thesis is to improve ASR text outputs to be in optimal written format for the Police by applying different post-processing approaches. Consequently, the training and test data had to contain different kinds of features which had the potential to be optimized. This included numbers, currencies, abbreviations,

punctuation, place names, and names. Specific examples of phrases that were of interest can be found in Table 1.1.

The Norwegian Language Bank at the National Library of Norway provides open-source datasets with Norwegian speech and text [Spr]. They provide resources so that companies, researchers, and students can develop new language technology for Norwegian [Spr]. The dataset used in the thesis was found in their resource catalog.[1]

4.1.1 Exhibition Corpus

The Exhibition Corpus consists of 23 texts from the exhibition "Leve Språket" from the National Library of Norway [Spr13]. The texts are available in either Norwegian Bokmål or Nynorsk with a corresponding audio file [Spr13]. The recordings were made to serve as an audio guide in the exhibition room of "Leve Språket" [Spr13]. This is different from a recording of spontaneous speech in the context of for example an investigative interview. The narrators are two men, one with a dialect close to Bokmål and one with a Nynorsk-like dialect. It may be a weakness that the dataset has only two speakers and both are men. The evaluation of the state-of-the-art models will not be able to say anything about how the systems perform when women or children speak. On average the audio files has a length of 73 seconds with a standard deviation of 12 seconds. 9 texts from the dataset are written in Nynorsk while the rest is written in Bokmål. Since the ASR model that is utilized for transcribing the dataset is fine-tuned on a dataset in Norwegian Bokmål it can be assumed that the texts in Bokmål will have lower WER than the texts where the reference is in Nynorsk.

The texts include number words written in numerals, place names, personal names, and names of organizations in appropriate capitalization, punctuation, and some abbreviations.

Pre-processing

Before using the Exhibition Corpus texts as reference transcriptions in the evaluation of the models a minimum amount of pre-processing was done. Capitalization, arabic numerals, and punctuation were retained to be able to detect mistakes the models make with regard to these features. In the baseline evaluation of the ASR model chosen, retaining these features will likely increase the WER compared to for instance the benchmark results reported by the model developers.

The text files were modified to remove unnecessary empty lines and spaces at the beginning, middle, or end of the file. This was to ensure that every text used as a

[1]The resource catalog can be found on this page: https://www.nb.no/sprakbanken/en/resource-catalogue/.

reference had the same structure so that this would not affect the evaluation of the post-processing methods on the various texts.

In addition, the texts were modified to correspond to exactly what was heard in the recordings. In some places words were added, in other places, words were removed or the order of words in the sentence was changed. No nasal noises, coughing, or hesitations were heard, so non-linguistic sounds were not included in the reference transcription.

4.2 Experiments ASR

4.2.1 ASR

The first step towards answering the problem statement was to find an ASR model and utilize this model to transcribe the chosen dataset. This was important to be able to use real Norwegian text output from an ASR model for the further experimentation of converting ASR transcriptions from oral text form to written text form. In order to evaluate the extent to which it was possible to convert the text output into written form, it was also important to be able to evaluate how the state-of-the-art Norwegian open-source ASR models perform without any post-processing.

The framework chosen to be used for the transcription of the dataset is Wav2vec 2.0. This framework does verbatim transcription, which can be of interest to the Police for full transcriptions of investigative interviews. A model for Norwegian Bokmål has been fine-tuned by the AI-lab at The National Library of Norway [NB a].[2] The name of the model is `nb-wav2vec2-1b-bokmaal` and it has been fine-tuned with the Bokmål subset of the NPSC dataset [NB a]. The AI-lab has reported a WER of 6.33% and a CER of 2.48% on the NPSC test set for Norwegian Bokmål [NbA22]. A weakness of choosing this fine-tuned model is that it is fine-tuned on only one of Norway's two written standards, and it is desirable that an ASR system for use in the Police is able to transcribe both Bokmål and Nynorsk.

For the transcription set-up, Google Colab with access to GPUs was used. From Colab the fine-tuned `nb-wav2vec2-1b-bokmaal` model could be downloaded through the `Transformers` library in Python. After loading the model, the transcription was done utilizing the `pipeline` function in `Transformers`. For every file in a directory that was uploaded to Colab containing the audio recordings from the Exhibition Corpus, the pipeline transcribed the audio and saved the output text to a file with the same name as the audio file. The files were downloaded to be used for evaluation and further experiments locally. A code cell from Colab is shown in Algorithm 4.1.

[2]The model nb-wav2vec2-1b-bokmaal can be found at https://huggingface.co/NbAiLab/nb-wav2vec2-1b-bokmaal.

Algorithm 4.1 Code for transcribing audio files with the fine-tuned Wav2vec 2.0 model.

```
import os
from transformers import pipeline

model_name = 'NbAiLab/nb-wav2vec2-1b-bokmaal'

# For GPU: device=0
pipe = pipeline('automatic-speech-recognition', model=
    model_name, device=0)

directory = "audio"
output_directory = "text_wav2vec"

for filename in os.listdir(directory):
  f = os.path.join(directory, filename)
  if os.path.isfile(f):
    transcription = pipe(f)
    with open(os.path.join(output_directory, filename.
        replace(".mp3",".txt")), "w") as txtfile:
      txtfile.write(transcription['text'])
```

4.2.2 Testing Whisper

In addition to `nb-wav2vec2-1b-bokmaal`, the Exhibition Corpus was also transcribed by the Whisper models: Base, Small, and Medium. This was done to have some results to compare the post-processing steps in the thesis with. Similarly to the transcription with `nb-wav2vec2-1b-bokmaal`, the Whipser models were tested in Colab with access to GPUs. The code for transcribing with the Whisper Medium model is shown in Algorithm D.10, the transcription with the other Whisper models is done by replacing "medium" with another name. The Whisper models output already post-processed text, but unlike Wav2vec 2.0 the models do not do verbatim transcription. Time and accuracy are two parameters that can be interesting to compare between the different ASR models available for Norwegian.

4.3 Experiments Post-processing

The post-processing experiments and evaluation were run locally (except when using the NER model) on a laptop with Windows 10, processor AMD Ryzen 5 3500U with Radeon Vega Mobile Gfx, and 8 GB RAM. The Python programs were written in Visual Studio Code and Python version 3.8.16 was used. For the experiments with capitalization and punctuation, Google Colab was utilized to download the

`nb-bert-base-ner` model with the `transformers` library in Python. In Colab Python version 3.10.12 was used. In contrast to the utilization of GPUs in Colab for the transcription, the GPU was not utilized for the post-processing steps with NER.

4.3.1 Utilizing NER for Capitalizing Proper Nouns

The AI-lab at the National Library of Norway has fine-tuned the NB-Bert base model for the NER task [NB a], with the name `nb-bert-base-ner`.[3] The model is able to identify names of persons, locations, and organizations. The code for utilizing the `nb-bert-base-ner` model is shown in Algorithm D.1. After applying the NER model on a text, the result is a list of tokens. Every token has an entity group, probability score, word, start, and end. By utilizing the model, code can be written to capitalize the first character (the character at the index "start") in the words identified by the model. And if the "word" in the token consists of multiple words, the start of every word was capitalized.

For instance, in file 02.txt the phrase "helene uri forfatter og språkviter" triggered the `nb-bert-base-ner` model to return this token:

```
{'entity_group': 'PER', 'score': 0.87166595, 'word': 'helene
    uri', 'start': 751, 'end': 761}
```

After running the capitalization code the phrase looks like this "Helene Uri forfatter og språkviter". Code for the capitalization of all texts in the dataset is shown in Algorithm D.2.

4.3.2 Utilizing NER for Punctuation

One punctuation rule for Norwegian is to use commas to separate three or more words written in a series where there is not any conjunction present. For instance, the list of names "Marit Kyle og Thomas" should be written with a comma between the two names that are following each other "Marit**,** Kyle og Thomas". The same fine-tuned NB-Bert base NER model can possibly be used for this purpose. First, the NER model identifies the proper nouns in the text. Then the code checks if the "end" of a token is one less than the "start" of the next token. If this is true a comma can be added to the output text. The code for adding commas to a list of proper nouns in an example sentence utilizing a NER model is shown in Algorithm D.3. The code for utilizing the `nb-bert-base-ner` model is the same as for capitalization with NER and is shown in Algorithm D.1.

[3]The nb-bert-base-ner model can be found at https://huggingface.co/NbAiLab/nb-bert-base-ner.

4.3.3 Utilizing Post-processing Scripts from the NPSC for ITN

In the preliminary project, the plan was to adapt the English NeMo Inverse Text Normalization tool to convert numbers, dates, and abbreviations in the ASR transcriptions to the suitable written format [Moe22]. The NeMo ITN tool is based on WFST grammars.

While the Language Bank at the National Library of Norway worked to develop the NPSC they also developed post-processing scripts that were used to convert years, dates, other numbers, and abbreviations to a standardized format.[4] These post-processing scripts are made available with the speech corpus and already include grammar for converting the kind of features that is interesting in this thesis. The post-processing scripts will be used to test their effectiveness as a tool for ITN of ASR transcriptions.

The files: `abbrev_grammar.py`, `date_grammar.py`, `year_grammar.py`, and `number_grammar.py` were copied from the `project_files/postprocessing_scripts/grammars` folder in the NPSC. A new file `normalizer.py` was written to combine the grammar from the different files and apply them to the ASR output to convert numbers and abbreviations. The `normalizer.py` file is shown in Algorithm 4.2.

[4]The post-processing scripts are downloaded with the speech corpus available at https://www.nb.no/sprakbanken/en/resource-catalogue/oai-nb-no-sbr-58/.

Algorithm 4.2 Code for combining the NPSC grammars and applying them to ASR transcriptions.

```
1  from number_grammar import numbergrammar_abovethirteen
2  from year_grammar import yeargrammar
3  from date_grammar import dategrammar
4  from abbrev_grammar import abbrevgrammar
5  import os
6
7  def itn(text, grammar):
8      result = grammar.searchString(text)
9      for i in reversed(range(len(result))):
10         text = text[:int(result[i][1])-len(result[i][0][0])]
               + str(result[i][0][1]) + text[int(result[i][1])
               :]
11 return text
12
13 # Combining the NPSC grammars:
14 grammar = abbrevgrammar ^ dategrammar ^ yeargrammar ^
       numbergrammar_abovethirteen
15
16 input_directory = "name_of_input_directory"
17 output_directory = "name_of_output_directory"
18
19 for file in os.listdir(input_directory):
20     filepath = os.path.join(input_directory, file)
21     t = open(filepath, "r", encoding="utf8").read()
22     output = itn(t, grammar)
23     with open(os.path.join(output_directory, file), "w",
           encoding="utf8") as outputfile:
24         outputfile.write(str(output))
```

The original output from the `searchString` function call on line 8 in Algorithm 4.2, used to locate the words that should be normalized, is a list containing the different instances of phrases to be normalized, their corresponding normalized value, and the location of the last character of the word in the text. An example of the result of: `grammar.searchString(05.txt)` is shown below:

```
[[('to hundre', 200), 152], [('cirka', 'ca.'), 188], [('åtte
    hundre', 800), 203], [('fjorten hundre', 1400), 402],
   [('fjorten hundre', 1400), 575], [('hundre', 100), 1182]]
```

In this example, six phrases to be normalized were found, and the list of results contains a new list for every phrase found. Each list contains a tuple at index 0.

The tuple contains two values, first the located phrase to be normalized, then the corresponding normalized value. At index one in the list, the index of the last character in the word is found. Line number 10 in Algorithm 4.2 is written to convert the located phrases to the correct written format in the output text from the ITN post-processing step. Slicing is used to delete the located phrase and replace it with the value at index 1 in the tuple.

Function for Dividing Words

A common mistake made by the fine-tuned Wav2vec 2.0 model was to merge number words with the neighboring words. Because some words like "tallet" and "årene" often can be found after years, a function `split_number_words` was written to add a space between the number and "tallet/årene". This could make it easier for the ITN grammars to understand the year and convert it correctly. The function is shown in Algorithm 4.3. The texts were run through this division code before trying the ITN step in Algorithm 4.2 again.

Algorithm 4.3 Code for splitting number words from neighboring words, and example of a split_words list.

```
def split_number_words(text, split_words):
    new_text = ""
    for word in text.split():
        divided = False
        for split_word in split_words:
            if (split_word in word) & (len(word) > len(
                split_word)):
                new_text += " ".join(word.split(split_word))
                    + split_word + " "
                divided = True
                break
        if not divided:
            new_text += word + " "
    return new_text

split_words = ["tallet", "årene"]
```

4.4 Evaluation Methodologies

The ASR models and post-processing steps are evaluated by how accurate they are in addition to the time taken to transcribe the audio files or convert the transcription text. In Chapter 5 the results will be presented for the evaluation of individual post-

processing steps and different sequences to find out which sequence of post-processing steps will give the best performance of the system.

4.4.1 Accuracy

The results of the accuracy of transcription and post-processing steps were evaluated using the metrics WER and CER. The chosen metrics could both be found in the JiWER package in Python.[5] After importing JiWER, functions of all the metrics could be used by simply providing a reference string and a hypothesis string. Both the reference and hypothesis texts were written in their own text files, so they had to be read into a string first. A Python file was made to ease the evaluation of the different experiments. In the file functions for evaluating either two files or two whole directories were made. This code's output was a table of the individual texts WER and CER like the tables shown in Appendix A. The evaluation code used can be found in Appendix D, in Algorithm D.6 the function for evaluating one text file against a reference text can be found, while Algorithm D.7 shows the evaluation of a whole directory.

A new Python file was made to calculate the median, minimum, maximum, average, and standard deviation of the WER and CER in the tables. These values were used to produce the tables, box plots, and bar charts presented in Chapter 5. Code from the Python file to calculate average and standard deviation can be found in Algorithm D.8. Algorithm D.9 shows a complete code to generate a box plot of the baseline evaluation table.

In addition to the automatic calculation of the WER and CER, a small-scale manual method was used to evaluate the accuracy of the NER model in the capitalization experiment. This was done because the NER based capitalization is dependent on the underlying NER model. The number of tokens in the reference and the ones identifies by the NER model was counted manually to calculate the precision, recall, and F1 score.

4.4.2 Speed

To be able to evaluate the speed of the different ASR models and post-processing steps, the transcription of every audio file and the application of every post-processing step to each ASR output was timed. Then the average and standard deviation could be calculated. The timing was done with the Python library `time`, and the `perf_counter` function was utilized to measure the time before the process that was timed started and directly after the process was finished. The elapsed time is found by subtracting the start time from the end time. An example of timing a function

[5]The documentation of JiWER can be found at: https://github.com/jitsi/jiwer.

called `function()` is shown in Algorithm D.4. Algorithm D.5 shows how the average and standard deviation of the elapsed time is calculated.

Chapter 5
Results

This Chapter will present the results obtained through the experiments described in Chapter 4. First, Section 5.1 and Section 5.2, show the results of transcription with different ASR models without any post-processing steps. Section 5.3 and Section 5.4 show the results of utilizing the NER model for capitalization of proper nouns and adding commas respectively. Section 5.5 shows the evaluation of the NPSC grammars and the post-processing steps of capitalizing, and converting numbers and abbreviations in sequence. Section 5.6 shows results similar to Section 5.5 after the code for division of commonly merged number words is applied. Lastly, Section 5.7 shows the final results of the best sequence of post-processing steps.

5.1 Baseline Results ASR

The results achieved by evaluating the fine-tuned Wav2vec 2.0 ASR output before any post-processing are shown in Figure 5.1, Figure 5.2, and Figure 5.3. In Appendix A a table with individual WER and CER scores for the files is presented. The reference transcriptions given in the Exhibition Corpus from the Language Bank contain capitalization, numbers, and punctuation, unlike the Wav2vec 2.0 model which only outputs words in lowercase letters. This may explain the large gap between these results and those reported by the AI-lab at The National Library of Norway for the model (WER: 6.33%, CER: 2.48%). In addition, the Exhibition Corpus was not used in fine-tuning the Wav2vec 2.0 model, which can also contribute to an increase in the metrics from what was reported. The transcription time of the Wav2vec 2.0 model is presented alongside the transcription time of the Whisper models in Table 5.1.

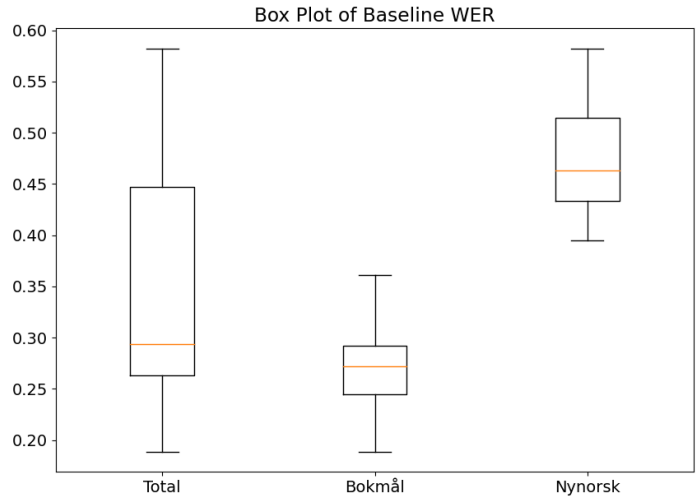


Figure 5.1: Box Plot of the nb-wav2vec2-1b-bokmaal model's WER for all 23 texts, and texts written in Bokmål and Nynorsk separately.

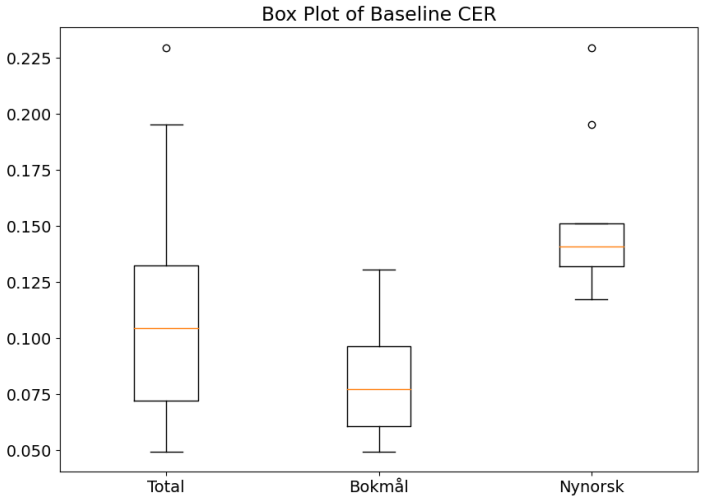


Figure 5.2: Box Plot of the nb-wav2vec2-1b-bokmaal model's CER for all 23 texts, and texts written in Bokmål and Nynorsk separately.

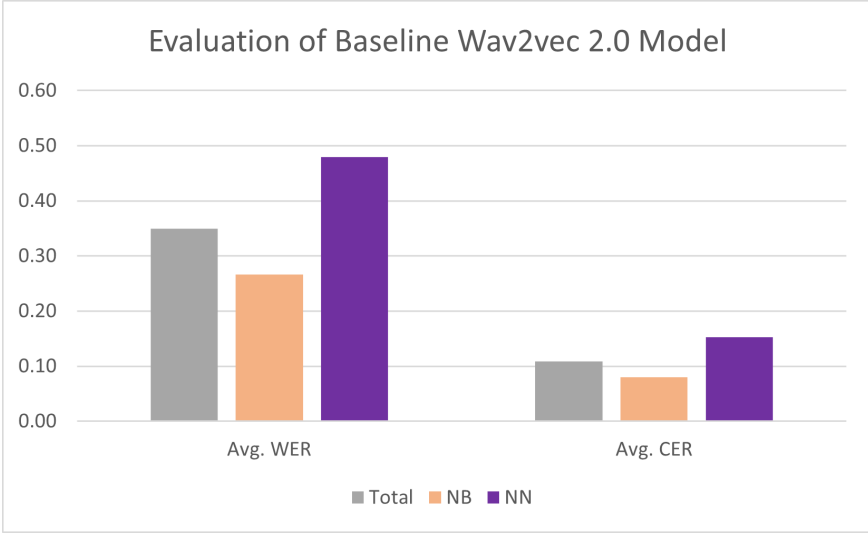


Figure 5.3: Average WER and CER of the nb-wav2vec2-1b-bokmaal model on the Exhibition Corpus divided into Norwegian Bokmål (NB) and Norwegian Nynorsk (NN) according to which written language is used in the reference transcription.

5.2 Time and Accuracy of ASR Models

The average and standard deviation of the transcription time for the different models are shown in Table 5.1. In addition to the transcription time, the models needed time before starting the transcription of the files in the Exhibition Corpus to load the model. The Wav2vec 2.0 model used approximately 3 minutes and 32 seconds to load the model, Whisper Base needed 18 seconds, Whisper Small 9 seconds, and Whisper Medium 51 seconds. The time to transcribe all files in the dataset spans from 1 minute and 33 seconds (Whisper Base) to 5 minutes and 29 seconds (Whisper Medium). For all ASR models, Google Colab was used with the hardware accelerator chosen to be GPU.

Model	**Average Time [s]**	**Standard Deviation [s]**
Nb-wav2vec2-1b-bokmaal	5.25	1.15
Whisper Base	4.05	0.72
Whisper Small	7.38	1.40
Whisper Medium	14.31	2.64

Table 5.1: Average and standard deviation of the transcription time for different ASR models.

The average metrics from evaluating the different Whisper models and the baseline Wav2vec 2.0 model are shown in Table 5.2. The Wav2vec 2.0 model does not include any post-processing, in contrast to the Whisper models which all applies punctuation, capitalization, and writing numbers while transcribing, this may explain why the Wav2vec 2.0 model has lower accuracy than both the Whisper Small and Whisper Medium model. The Whisper Base model is trained on a minimal amount of data and fails to transcribe words such as "om" and "fikk" which all the other models recognize. It also capitalizes words that should not be capitalized. For instance, one phrase from "02.mp3" was transcribed "vi fikk syttisju bidrag til konkurransen" by the Wav2vec 2.0 model, "Vi fikk 77 bidrag til konkurransen." by the Whisper Small and Medium models, and "Vi fæk 70 bidrag til Konkurrensen" by the Whisper Base model. In this example, both Whisper Small's and Whisper Medium's transcription corresponds to the reference transcription in the dataset. When transcribing with the different Whisper models, Whisper identified the language itself. If the language ("Norwegian") had been given as an argument, the results might have been better for the Whisper Base model.

Model	**Avg. WER**			**Avg. CER**		
	Tot.	**Nb**	**Nn**	**Tot.**	**Nb**	**Nn**
Nb-wav2vec2-1b-bokmaal	34.94	26.60	47.90	10.86	8.03	15.26
Whisper Base	47.92	41.03	58.64	14.30	11.94	17.97
Whisper Small	31.64	22.72	45.50	7.92	5.34	11.92
Whisper Medium	24.44	16.64	36.58	5.60	3.58	8.74

Table 5.2: Average WER and CER of different ASR models. WER and CER are given in %.

5.3 Capitalization

Applying NER and using the tokens to capitalize proper nouns took approximately 0.66 seconds per file (with a standard deviation of 0.22 seconds), improved the WER by a little over 1% on average, and the CER by 0.26%. The average WER and CER are shown in Table 5.3, both for all the texts and for the Bokmål texts and Nynorsk texts separately.

Post-processing Step	Avg. WER			Avg. CER		
	Tot.	Nb	Nn	Tot.	Nb	Nn
None	34.94	26.60	47.90	10.86	8.03	15.26
NER for capitalization	33.77	25.44	46.73	10.60	7.77	15.02

Table 5.3: Evaluation of NER for capitalization experiment compared to the nb-wav2vec2-1b-bokmaal model. Metrics are given in %.

The `nb-bert-base-ner` model was not able to locate and categorize all the proper nouns in the text. For instance, the model did correctly categorize "norge" in the entity group GPE_LOC or GPE_ORG sometimes, and other times did not categorize it at all. This did lead to the word "norge" not always being properly capitalized. For the Nynorsk equivalent of "Norge", namely "Noreg", the NER model does not label the word at all. An example of the NER output from text 05 where the model was able to locate the proper nouns: "norge", "danmark", and "ivar aasen" is shown below. From the complete baseline evaluation (given in Table A.1), text 05 had a WER of approximately 29.08% and a CER of 13.04%, while after applying the NER model and capitalizing the located nouns, the WER was 26.53% and the CER 12.53%.

```
Tokens from the NER model for text 05:
[{'entity_group': 'GPE_LOC', 'score': 0.7027802, 'word': '
    norge', 'start': 351, 'end': 356},
{'entity_group': 'GPE_LOC', 'score': 0.7834933, 'word': '
    norge', 'start': 471, 'end': 476},
{'entity_group': 'GPE_LOC', 'score': 0.5355901, 'word': 'dan
    ', 'start': 517, 'end': 520},
{'entity_group': 'GPE_ORG', 'score': 0.64517725, 'word': '##
    mark', 'start': 520, 'end': 524},
{'entity_group': 'GPE_ORG', 'score': 0.55862474, 'word': '
    norge', 'start': 528, 'end': 533},
{'entity_group': 'PER', 'score': 0.9090406, 'word': 'iv', '
    start': 902, 'end': 904},
{'entity_group': 'PER', 'score': 0.9274455, 'word': '##ar
    aasen', 'start': 904, 'end': 912}]
```

The recall of the NER model for the individual Wav2vec 2.0 transcribed files is given in Table C.1. The average recall of all the files that contained proper nouns was approximately 43%, this means that of the identified proper nouns in the reference transcriptions only 43% of them were located by the NER model in the Wav2vec

2.0 transcriptions, and became capitalized. Of all proper nouns located by the NER model, all of them were correct, meaning the precision of the system was 100%. Combining these two metrics gives a F_1 score of approximately 60%.

While the `nb-bert-base-ner` model was trained on the NorNE dataset that includes already capitalized proper nouns, the transcriptions that the NER model is applied to in this post-processing step consist of text in only lowercase letters. This may have caused the big gap between the F_1 scores reported by Kummervold *et al.* [KDWB21], 91.2% when trained on NorNE Bokmål and 88.9% with the model trained on NorNE Nynorsk, and the F_1 scores obtained in this experiment.

5.4 Adding Commas

The results achieved by applying the comma rule to the ASR transcriptions were that none of the lists was detected by the NER model. No comma was added to the whole dataset.

While writing the comma code an example sentence was used. The sentence was "Jeg liker mange byer for eksempel trondheim bergen oslo og tromsø.". Here the preferred output should be "Jeg liker mange byer for eksempel trondheim, bergen, oslo og tromsø.". The NER tokens returned for the sentence are shown below.

```
NER tokens returned for "Jeg liker mange byer for eksempel
trondheim bergen oslo og tromsø.":
[{'entity_group': 'GPE_LOC', 'score': 0.9080655, 'word': '
    trond', 'start': 34, 'end': 39},
{'entity_group': 'GPE_LOC', 'score': 0.73947406, 'word': '
    oslo', 'start': 51, 'end': 55},
{'entity_group': 'GPE_LOC', 'score': 0.99204683, 'word': '
    trom', 'start': 59, 'end': 63}, {
'entity_group': 'GPE_LOC', 'score': 0.9066798, 'word': '##sø
    ', 'start': 63, 'end': 65}]
```

The NER model was not able to detect "bergen" as a location name, and only partially located "trondheim". Because "bergen" was the second location name, none of the detected tokens followed each other. Consequently, the output sentence remained the same as the input sentence, with no added comma.

When the location names were written with an initial capital letter "Jeg liker mange byer for eksempel Trondheim Bergen Oslo og Tromsø." the NER model detected all proper nouns as location names. The NER tokens are shown below.

```
NER tokens returned for "Jeg liker mange byer for eksempel
```

```
Trondheim Bergen Oslo og Tromsø.":
[{'entity_group': 'GPE_LOC', 'score': 0.99856204, 'word': '
    Trondheim', 'start': 34, 'end': 43},
{'entity_group': 'GPE_LOC', 'score': 0.98276156, 'word': '
    Bergen', 'start': 44, 'end': 50},
{'entity_group': 'GPE_LOC', 'score': 0.9891607, 'word': '
    Oslo', 'start': 51, 'end': 55},
{'entity_group': 'GPE_LOC', 'score': 0.9987262, 'word': '
    Tromsø', 'start': 59, 'end': 65}]
```

In the tokens listed, the word "Trondheim" is followed by "Bergen" which is followed by "Oslo". Therefore, the output sentence becomes: "Jeg liker mange byer for eksempel Trondheim, Bergen, Oslo og Tromsø.".

5.5 Inverse Text Normalization

In Table 5.4 accuracy results achieved by applying the normalization rules including grammar for dates, years, numbers, and abbreviations are presented. The total average WER was improved by only 0.42%, but the total average CER was improved by almost 1%. The average time to apply the grammar was 0.35 seconds per file and the standard deviation was 0.07 seconds.

The results of capitalizing with NER and applying ITN sequentially are also shown in Table 5.4. The only difference between first capitalizing and then applying ITN, and first applying ITN and then capitalizing, is the initial letter of three words distributed to two files in the dataset, text 02 and text 15. The `NER_ITN` sequence has two correct initial letters that the `ITN_NER` sequence does not have, while the `ITN_NER` sequence has one correct initial letter that the `NER_ITN` sequence is missing.

Post-processing Step(s)	**Avg. WER**			**Avg. CER**		
	Tot.	**Nb**	**Nn**	**Tot.**	**Nb**	**Nn**
None	34.94	26.60	47.90	10.86	8.03	15.26
ITN	34.52	26.06	47.68	9.94	7.25	14.13
NER_ITN	33.33	24.91	46.44	9.68	7.00	13.86
ITN_NER	33.35	24.91	46.49	9.68	7.00	13.86

Table 5.4: Evaluation of ITN experiment compared to the nb-wav2vec2-1b-bokmaal model and two different sequences of applying ITN and utilizing NER for capitalization. Metrics are given in %.

5.6 Dividing Commonly Merged Words

The accuracy results of ITN after running the spitting code to divide commonly merged words by the ASR model is given in Table 5.5 with the model name `Divided_ITN`. The total average CER decreased by 0.48% from the evaluation of ITN alone. At the same time, the WER increased by 0.32%. This may be caused by, for instance, "attenhundretallet" being counted as one wrong word in the WER calculation of ITN while "1800 tallet" is counted as two wrong words in the calculation of WER for `Divided_ITN`. The characters, on the other hand, are more correct for the phrase after `Divided_ITN`. Despite the increase in WER, the readability of the phrase can be said to have been improved with the help of the division code. A total of 12 new numbers were converted with the help of the division code.

The two sequences, `NER_ITN` and `ITN_NER`, also see the same trend with increasing WER and decreasing CER when dividing number words from "tallet" and "årene" before applying ITN, the new sequences including the division step are named `NER_Divided_ITN` and `Divided_ITN_NER` respectively in Table 5.5.

Post-processing Steps	**Avg. WER**			**Avg. CER**		
	Tot.	**Nb**	**Nn**	**Tot.**	**Nb**	**Nn**
Divided_ITN	34.84	26.38	47.99	9.46	6.81	13.56
NER_Divided_ITN	33.65	25.22	46.75	9.20	6.56	13.30
Divided_ITN_NER	33.67	25.22	46.81	9.19	6.55	13.29
NER_Divided_ITN_NER	33.57	25.14	46.68	9.18	6.54	13.28
Divided_ITN_NER_NER	33.56	25.14	46.67	9.17	6.54	13.26

Table 5.5: Evaluation of ITN experiment and different sequences after the division of commonly merged number words. Metrics are given in %.

In addition to the three initial capital letters differing between `NER_ITN` and `ITN_NER`, the `Divided_ITN_NER` sequence has identified three more proper nouns that have gotten an initial capital letter. This may explain the small gap in CER between the `NER_Divided_ITN` and `Divided_ITN_NER` sequence. The ASR model's spelling of the relevant names differs from the spelling of the names in the reference transcription. For instance "Knud" in the reference becomes "Knut" in the ASR output and "Jacobine" in the reference corresponds to "Jakobine" in the ASR output. Since the spelling differs, the WER is not affected by the localization of these proper nouns.

Because of the difference in the localization of proper nouns in the different

sequences, the post-processed transcriptions were run through the NER and capitalization code one more time. These results are shown in Table 5.5 with the model names `NER_Divided_ITN_NER` and `Divided_ITN_NER_NER`. Both sequences get a decrease in WER and CER, and the `Divided_ITN_NER_NER` ended up with lower scores in both metrics. The recall of the capitalization for the last NER step in the `Divided_ITN_NER_NER` sequence is given in Table C.2. The average recall ended up at 48%, and all capitalized letters were still correct giving a precision of 100%. Together the recall and precision produce a F_1 score of approximately 65%, which is a 5% increase from the F_1 score the NER capitalization achieved alone.

5.7 Final Results

The best readability and lowest evaluation metrics scores are achieved by first dividing common number word mistakes, then running the NPSC grammars to apply ITN, and lastly capitalizing proper nouns by running the transcription through the NER capitalization model two times. The `Divided_ITN_NER_NER` sequence is also the only sequence tested that gets two commas added by the NER punctuation code. In Appendix B an example of text 02 from the Exhibition Corpus is presented, the reference text is shown alongside the output from the fine-tuned Wav2vec 2.0 ASR model and the final post-processed transcription.

In Figure 5.4, the baseline Wav2vec 2.0 evaluation and the final sequence's average measures are shown side by side. The measurements show a slight decrease in all values both for the Bokmål and Nynorsk texts.

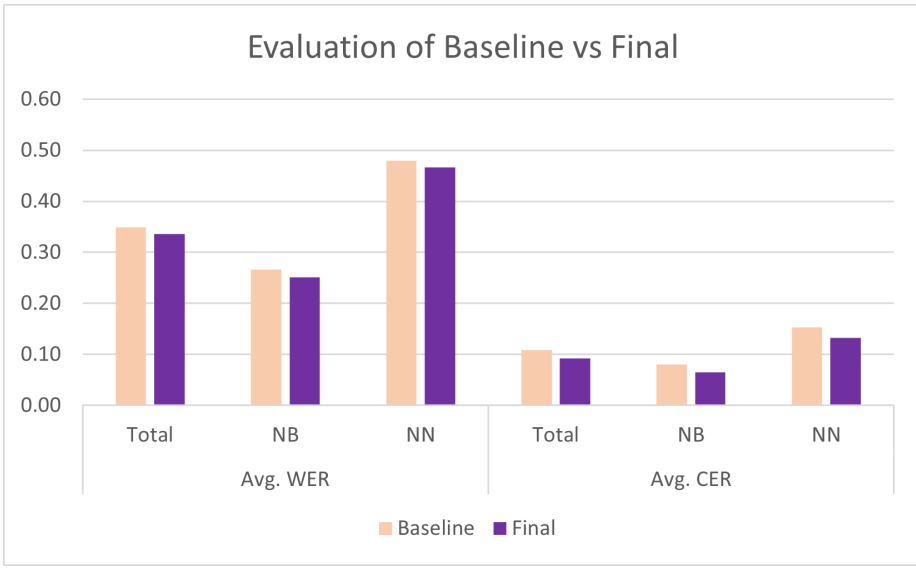


Figure 5.4: Average WER and CER of the Exhibition Corpus for the baseline nb-wav2vec2-1b-bokmaal model and the final sequence Divided_ITN_NER_NER.

Figure 5.5 shows the minimum, maximum, and median of the final sequence's WER evaluation results compared to the baselines. And Figure 5.6 shows the same for the CER. Compared to the box plots from the baseline evaluation (also shown in Figure 5.1 and Figure 5.2), some differences are observed. The maximum value of the WER has increased for Nynorsk and decreased for Bokmål from the baseline to the final sequence evaluation. Previously, the Bokmål and Nynorsk plots for the CER had some overlap, but for the final sequence, they do not overlap. The Bokmål CER box plot has been squeezed together, while the Nynorsk plot remains scattered. Overall, the median of all the texts CER has decreased by 2%.

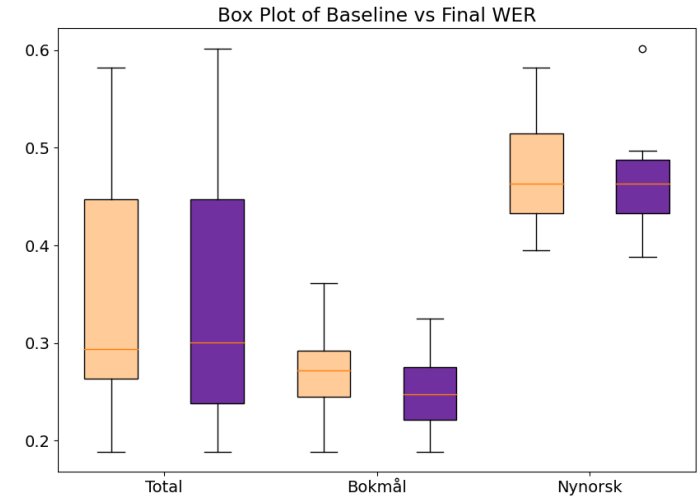


Figure 5.5: Box Plot of the Baselines (orange) vs Divided_ITN_NER_NER's (purple) WER for all 23 texts, and texts written in Bokmål and Nynorsk separately.

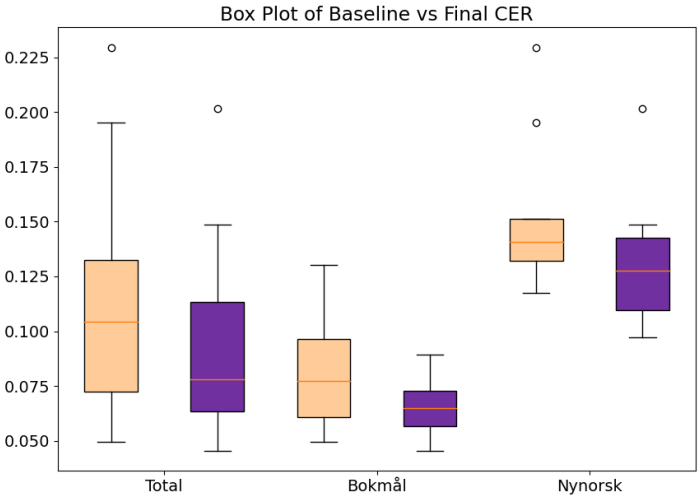


Figure 5.6: Box Plot of the Baselines (orange) vs Divided_ITN_NER_NER's (purple) CER for all 23 texts, and texts written in Bokmål and Nynorsk separately.

The final evaluation of processing time is presented in Table 5.6 as average times for applying the different post-processing steps alone, and as a total average time per file including all post-processing steps in the final sequence `Divided_ITN_NER_NER`. To divide the number words from the common neighboring words in the Wav2vec 2.0 output, approximately 0 seconds was used, this time is therefore not included in the table. It is worth noting that only the elapsed times of the transcription of the dataset with the fine-tuned Wav2vec 2.0 model were made with access to a GPU.

Post-processing Step	**Applied to**	**Average time [s]**	**Standard Deviation [s]**
Nb-wav2vec2-1b-bokmaal	Exhibition Corpus	5.25	1.15
NER	Nb-wav2vec2-1b-bokmaal output	0.66	0.22
ITN	Nb-wav2vec2-1b-bokmaal output	0.35	0.07
ITN	Divided nb-wav2vec2-1b-bokmaal output	0.35	0.06
ITN	NER	0.40	0.07
ITN	NER_Divided	0.38	0.08
NER	ITN	0.45	0.08
NER	Divided_ITN	0.49	0.11
NER	NER_Divided_ITN	0.47	0.09
NER	Divided_ITN_NER	0.49	0.11
Total	Nb-wav2vec2-1b-bokmaal+ITN+NER+NER	6.58	

Table 5.6: Average and standard deviation of the elapsed time of applying the post-processing steps and the total average time from transcription to finished post-processed transcriptions per file. The underlined values are summed to make the total transcription and post-processing time per file.

Chapter 6
Discussion

In this Chapter the experiments described in Chapter 4 and the results shown in Chapter 5 will be discussed to answer the objective and the RQs of this thesis. First, RQ2 will be partially answered in Section 6.1 and Section 6.2. Then RQ1 will be partially answered in Section 6.3 and Section 6.4. In Section 6.5 it will be discussed how the results benefit investigative interviews and investigations. Lastly, in Section 6.6 some limitations of the research will be described.

The overall objective of the thesis has been to *investigate to which extent it is possible to convert Norwegian text output from an ASR algorithm from oral text form to written text form.* The short answer to this is that this is possible to a large extent because the Whisper Medium and Large model is already on the market and deals with for instance converting numerical values to Arabic numerals, punctuation, and capitalization. The output of these models is already in written text form. As shown in Section 5.2 the Whisper Medium model achieves a WER of 16.64% and a CER of 5.60% on the Norwegian Bokmål texts of the Exhibition Corpus. The model used an average time of 14.31 seconds on a GPU to transcribe the audio files that have an average length of 73 seconds, so the model is still faster than real-time transcription. The Whisper models were published in the autumn of 2022 after the problem proposal for this master's thesis was presented. Nevertheless, the chosen ASR model Wav2vec 2.0 is still relevant, and therefore the objective is still interesting to investigate in the context of this ASR model. In the following Sections, some categories of ASR text post-processing are discussed.

6.1 Capitalizing

In this Section RQ2: "To what extent can we improve sentence structure in ASR transcriptions by improving punctuation, the spelling of compound words, abbreviations, and capital letters in proper nouns?" will be partially answered.

To capitalize letters in proper nouns, it was attempted to utilize a NER model

to first locate any proper nouns in the text. This could be the names of persons, organizations, and locations. The proposed solution was rule-based and included capitalizing the initial letter in all the words included in the tokens returned by the `nb-bert-base-ner` NER model. The result entails that the post-processing step of capitalizing proper nouns is only as good as the NER model used. The precision was measured to be 100%. Consequently, if the NER model performs well on locating proper nouns in the text then the nouns that should be capitalized will be.

The capitalization made it easier to locate proper nouns while reading the transcription (see for instance the example of text 02 in Appendix B). Capitalization of proper nouns can be interesting for the Police because the proper nouns can contain especially interesting information to the Police. For instance, information that can help answer "how, where, when, why, and by whom a crime was, or will be, committed". Nevertheless, the recall of the utilized NER model was low. The best recall of 48%, was achieved by iterating through the NER capitalization two times, and even then the recall is lower than all recall results mentioned in Section 3.3.1. The low recall means that some of the proper nouns were capitalized and some were not, so the investigator can not trust the capitalization in the transcription completely.

An observation made during the experiment was that the NER model did identify more tokens the last time the NER model was run than the first time, even though the only difference was the initial capital letter from the previous NER capitalization step. From the first iteration of the NER capitalization to the last one in the `Divided_ITN_NER_NER` sequence, the recall and F_1 score increased by 5%. Because of this, it is conceivable that running the NER capitalization code in sequence additional times can give further improvements to the metrics. As Mayhew *et al.* [MTR19] and Kuster *et al.* [KFM21] have shown, the amount of lowercased text in the training data of the NER model can have a big impact on the model's ability to detect proper nouns in lowercase text. Choosing another NER model trained in another way may therefore have an impact on the system's performance in locating and capitalizing proper nouns.

A drawback of the capitalization step is that the NER model does not differentiate between organization names where only the initial letter should be capitalized and organization names that are acronyms where all letters typically are capitalized. Therefore, all names of organizations located by the NER model are written with only the initial letter capitalized in the transcription.

To summarize RQ2 in the context of capitalization, the capitalization of proper nouns by the approach in this thesis did not do much to the sentence structure. Sentence structure of an ASR transcription can be improved to a limited extent by

capitalizing proper nouns. Mainly because the capitalization made it a little easier to locate proper nouns in the text. This can be a benefit to investigators trying to answer "how, where, when, why, and by whom a crime was, or will be, committed". The recall after iterating through the NER capitalization post-processing step is still low. Despite this, an observation that more words were identified by the NER model the second time makes it conceivable that the recall can improve by iterating through the NER capitalization additional times. In addition, choosing a different NER model may have an impact on the results.

6.2 Punctuation

A proposed step to add some punctuation (commas) was to add a comma between words if the “end” of one token was followed by the “start” of a new token. The results show that this solution could have worked well if the NER model was good enough to detect all proper nouns in a list correctly. In the example sentence “Jeg liker mange byer for eksempel Trondheim Bergen Oslo og Tromsø” the NER model correctly located four city names if the city names were written grammatically correct with the initial letter of the word in uppercase. When the initial letter of the words was written in lowercase, the same style as the Wav2vec 2.0 output, the NER model located some of the cities. But if the model only missed one of the cities, for instance, “Bergen”, no punctuation was added. The addition of commas is dependent on the NER model. If the recall of the NER model is low, the adding of commas also has a low impact on the transcription.

The test of applying commas to lists in the Exhibition Corpus without any other post-processing was disappointing. Unsatisfactorily the NER model did not detect the lists of proper nouns and therefore none of them got a comma between the phrases. In the sequence `Divided_ITN_NER_NER`, the running of the punctuation code returned the transcriptions with two added commas in total.

Another possible solution to this “add comma between phrases” task could be to look for common conjunctions like “og” and “eller”, and locate possible lists with this. In this way, it could also be possible to locate other lists than just the ones containing proper nouns. For instance, the Exhibition Corpus has texts where different languages are listed, but in Norwegian, language names are not written with initial capital letters.

Concerning RQ2, it is difficult to answer to what extent the insertion of punctuation would improve the sentence structure in ASR transcriptions because only two commas have been added to the transcriptions at this point. At least the insertion of punctuation to ASR transcriptions with the help of the approach explored in this thesis with inserting a comma between proper nouns located by a NER model, does

not improve the sentence structure to any extent. The recall of the utilized NER model is still low, causing only two pairs of proper nouns following each other to be found.

6.3 Numbers, Years, Dates

A proposed method to answer RQ1: "How can Inverse Text Normalization (ITN) be adapted to convert numbers, monitory values, dates, and abbreviations to the proper written format for the Norwegian language?" was to utilize the grammar and `searchString` function from the post-processing scripts developed by the Language Bank. They could locate words in the transcription to be normalized and propose a normalized phrase, this could be adapted to convert the ASR transcription to the written domain. This was chosen to be used instead of adapting the NeMo ITN tool to Norwegian because both methods were rule-based, and the post-processing scripts already were made for Norwegian and included important features like converting years, dates, and numbers that were interesting in the task of standardizing ASR text output.

The normalization code in Algorithm 4.2 successfully converted most of the years and numbers correctly from oral to written text form. Nevertheless, the post-processing step only improved the WER with 0.37% and the CER with a little under 1%. This can be due to the text not only consisting of relevant text with a lot of numbers. But as shown in Table A.2, some of the texts in the Exhibition Corpus had a higher WER and CER than the same text in the evaluation of the baseline Wav2vec 2.0 model shown in Table A.1, and therefore contributed to increasing the total average WER and CER. This is because some of the number words in the reference transcriptions were written in letters instead of Arabic numerals. For instance when it was mentioned "Hva tror du vil skje med språket vårt i de neste hundreårene?" in reference text 05. The ITN step converted the corresponding Wav2vec 2.0 output to "hva tror du vil skje med språket vårt i de neste 100 årene?", which will give the new post-processed transcription a higher WER and CER compared to the previous transcription. This increase in the metrics is contributing to the total average of the metrics, and therefore a way of handling these exceptions could have decreased the metrics further.

In addition, the method of utilizing the grammar to locate phrases to be normalized and propose a normalized value was fast. The elapsed time to convert the text from oral to written form was on average approximately 0.35 seconds per file (original audio length was 73 seconds on average).

In some of the texts, the Wav2vec 2.0 model had number words merged with other words, and therefore the ITN step alone did not work well, this can also explain the

low improvement in accuracy from the baseline Wav2vec 2.0 evaluation to the result of evaluating the individual ITN post-processing step. Number words merged with the words “tallet” and “årene” were repeated multiple times in the texts. Since the ITN step successfully converted the numbers when the number word was separated from other phrases, a possible solution to improve the post-processing step was to divide words where the number words were merged with common neighboring phrases. After dividing the commonly merged words, the WER increased while the CER decreased. To decrease the WER it could have been added a hyphen between the number and the phrase “tallet” or “årene” instead of a space. Then the phrase would have been counted as one hit instead of one substitution and one deletion by the WER metric evaluation.

A limitation of the NPSC grammars is that it is missing code for converting ordinal numbers. For instance, “femte til tiende” is still written with letters in text 02 (shown in Appendix B). A modification of the NPSC grammars to include more rules would be needed if the grammars should be used as a tool for ITN on a large scale.

To answer RQ1, for the Norwegian language the normalization grammars included with the NPSC corpus can be utilized to apply ITN to ASR output to convert numbers to the written format. The grammar needs to have the number words divided from the other words to be able to detect them, so for an ASR model that often outputs number words merged with neighboring words this ITN approach may not be the best. In addition, the grammar is missing code to convert ordinal numbers to the written format. Nevertheless, the efficiency of utilizing the NPSC grammars for the task of ITN on ASR transcriptions are good.

6.4 Abbreviations

The utilization of the post-processing scripts from the NPSC successfully converts all the abbreviations in the dataset, so the precision of the abbreviation grammar was good. This included abbreviations like “det vil si, dvs.”, “cirka, ca.”, “for eksempel, f.eks.”, and “blant annet, bl.a.”. Because the dataset did not include a lot of abbreviations it is difficult to say something about how the abbreviation grammar would scale for bigger texts, and how beneficial it is in reality. To answer RQ1 with respect to abbreviations, the utilization of the grammar and the `searchString` function from the NPSC dataset is both accurate and fast, and therefore can be utilized to convert abbreviations in ASR transcriptions to the proper written format for Norwegian.

With respect to RQ2, whether the conversion of abbreviations improved the sentence structure or not, is hard to answer. The conversion of the abbreviations in

the Exhibition Corpus improved the sentence structure to a limited extent because the dataset did not contain a lot of abbreviations. For transcriptions of investigative interviews, the use of the correct abbreviation can make the text shorter and more concise. However, the Language Council of Norway writes that as long as the abbreviations are not incorporated, writing the words out completely is the most reader-friendly in continuous text [Kla]. Though the approach of converting abbreviations has proven to work well, it can potentially worsen readability and sentence structure. Consequently, it may not be desirable to adopt it in the context of transcription of investigative interviews.

6.5 The Benefits of the Results for Investigative Interviews

The speed obtained by combining the time of the ASR model and the time to apply post-processing steps is still faster than the duration of the audio recordings and a lot faster than complete manual transcription (four times the duration of the recording [KLC+19]). It will therefore save the Police officers time to utilize an ASR system to transcribe investigative interviews, even if they have to add the punctuation themselves.

The accuracy is satisfactory for the transcriptions to be used as a template for the investigative report. But a manual correction of the transcriptions will be needed because of the still high WER of 33.56% and CER of 9.17%

6.6 Limitations

Firstly, the chosen dataset, the Exhibition Corpus, is short and therefore does not contain many examples for each post-processing case. This is a weakness because it makes it difficult to say something about the scalability of the research and exactly how accurate the proposed post-processing steps are. In addition, the recordings in the dataset were originally made to be used as an audio guide in the exhibition room, which is different from the speech in an interviewing context where the interviewee should speak freely. For instance, manuscript-read speech tends to not include hesitations and interruptions that occur in spontaneous speech.

In addition to the previously mentioned weaknesses of the dataset used, the narrators in the recordings are two men. The accuracy measures do not necessarily extend to correspond to accuracy measures of ASR transcriptions made by audio of women or children.

Another limitation of the research is the time measures shown in Chapter 5. The measurements are the elapsed time of only one iteration of the different post-processing steps. This makes the measurements inaccurate but gives an estimation

of how long the different post-processing steps take. For instance, does the step take milliseconds, seconds, or minutes to complete? To make the time estimates more accurate, an average value of for instance 1000 iterations of the same post-processing step could have been done but this is beyond the scope of this thesis.

Lastly, only one NER model (`nb-bert-base-ner`) was utilized for the capitalization and adding of commas to lists. Eventually combining different NER models could have improved the final result by identifying more of the proper nouns in the Exhibition Corpus texts.

Chapter 7
Conclusion

The purpose of this thesis has been to *investigate to which extent it is possible to convert Norwegian text output from an ASR algorithm from oral text form to written text form.* To answer the two RQs both sequences of different post-processing steps and individual applications of post-processing steps were evaluated. The results show a slight improvement in the WER and CER with the best average WER being 33.56% achieved by applying ITN and NER in sequence. The total average WER had an improvement of 1.38%, while the total average CER had an improvement of 1.69%.

Converting the ASR transcriptions to include numerals, abbreviations, commas in lists, and capitalization of proper nouns were investigated. The utilization of the normalization grammar developed by the Language Bank at the National Library of Norway demonstrates some improvements in the readability of the numbers and abbreviations in the dataset but is dependent on the number words being properly separated from the other words to convert them. The post-processing steps of applying commas in lists and capitalizing proper nouns are only as good as the NER model used. The final sequence of post-processing steps was able to improve text 02, the text with the most proper nouns, from 58.23% to 48.73% WER, and improve text 09, the text with the most numbers, from 22.96% to 9.72% CER.

The readability of ASR output is improved, especially in the texts with the most numbers and proper nouns, which are the features experimented with in this thesis. Nevertheless, a manual correction of the ASR output must be expected if the Police were to apply the `nb-wav2vec2-1b-bokmaal` ASR model and the final post-processing sequence `Divided_ITN_NER_NER` because a WER of 33.56% is still high. Further, the time to apply ASR and the different post-processing steps is very low, so the benefit of at least using the ASR model and post-processing steps as a template for the investigative interview transcription is high. A complete manual transcription takes longer than transcribing with an ASR model, applying post-processing steps and correcting it.

7.1 Future Work

Multiple challenges for Norwegian ASR are still to be investigated. Based on the results and discussion in this thesis some potential areas for future research are the following:

- Conversion of acronyms of organization names to uppercase.
- Improving ITN by including the conversion of ordinal numbers.
- Investigate if the amount of uppercase and lowercase training data for Norwegian NER has the same impact on the ability to detect proper nouns in lowercased texts as indicated by Mayhew *et al.* [MTR19] and Kuster *et al.* [KFM21].
- Addition of punctuation, for instance, punctuation added when the speaker takes a pause. For this task, Voice Activity Detection can be an interesting starting point.
- The division of the transcription into a dialog. The application of speaker diarization to identify different speakers can be interesting.
- Addition of timestamps in the transcriptions. In professional transcriptions, timestamps are often present like in the paper of Kim *et al.* [KLC+19].

References

[ABG22] A. Antonova, E. Bakhturina, and B. Ginsburg, «Thutmose tagger: Single-pass neural model for inverse text normalization», *arXiv preprint arXiv:2208.00064*, 2022.

[AI4] AI4Interviews, Artificial Intelligence in Innovation of Investigative Interviews Speech-To-Text and Text Analysis Using Machine Learning. [Online]. Available: https://prosjektbanken.forskningsradet.no/en/project/FORISS/331893?Kilde=FORISS&distribution=Ar&chart=bar&calcType=funding&Sprak=no&sortBy=score&sortOrder=desc&resultCount=30&offset=0&Fritekst=243497&source=FORISS&projectId=243497 (last visited: Mar. 23, 2023).

[And19] S. Andersen, «Technical report: A preliminary process model for investigation», 2019.

[BEL08] T. Bazillon, Y. Estève, and D. Luzzati, «Manual vs assisted transcription of prepared and spontaneous speech», in *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, European Language Resources Association (ELRA), 2008.

[BZMA20] A. Baevski, Y. Zhou, *et al.*, «Wav2vec 2.0: A framework for self-supervised learning of speech representations», *Advances in Neural Information Processing Systems*, vol. 33, pp. 12 449–12 460, 2020.

[Cam] Cambridge Dictionary, phoneme. [Online]. Available: https://dictionary.cambridge.org/dictionary/english/phoneme (last visited: Feb. 15, 2023).

[Cho20] K. R. Chowdhary, «Automatic speech recognition», in *Fundamentals of Artificial Intelligence*. Springer India, 2020, pp. 651–668.

[DBK+23] J. De la Rosa, R.-A. Braaten, *et al.*, «Boosting norwegian automatic speech recognition», in *Proceedings of the 24th Nordic Conference on Computational Linguistics (NoDaLiDa)*, 2023, pp. 555–564.

[DHS+20] X. Du, C. Hargreaves, *et al.*, «Sok: Exploring the state of the art and the future potential of artificial intelligence in digital forensic investigation», in *Proceedings of the 15th International Conference on Availability, Reliability and Security*, 2020.

[Eri13] P. K. F. Eriksen, *Avhørsrapporten som rekontekstualisering av avhøret*, 2013.

[Fah16] I. A. Fahsing, *The making of an expert detective. Thinking and deciding in criminal investigations*. 2016.

[GCD+22] A. Gupta, N. Chhimwal, *et al.*, «Indic-punct: An automatic punctuation restoration and inverse text normalization framework for indic languages», *arXiv preprint arXiv:2203.16825*, 2022.

[GKX+23] Y. Gaur, N. Kibre, *et al.*, «Streaming, fast and accurate on-device inverse text normalization for automatic speech recognition», in *2022 IEEE Spoken Language Technology Workshop (SLT)*, IEEE, 2023, pp. 237–244.

[Hug] Hugging Face, Models. [Online]. Available: https://huggingface.co/models?pipeline_tag=automatic-speech-recognition&language=no&sort=downloads (last visited: Apr. 19, 2023).

[Haa08] Å. Haaland, «A maximum entropy approach to proper name classification for norwegian», 2008.

[JAH+19] F. Jørgensen, T. Aasmoe, *et al.*, «Norne: Annotating named entities for norwegian», *arXiv preprint arXiv:1911.12146*, 2019.

[JHH+05] J. B. Johannessen, K. Hagen, *et al.*, «Named entity recognition for the mainland scandinavian languages», *Literary and Linguistic Computing*, vol. 20, no. 1, pp. 91–102, 2005.

[JM18] D. Jurafsky and J. H. Martin, «Regular expressions, text normalization, edit distance», *Speech and Language Processing*, pp. 1–28, 2018.

[Joh19] B. Johansen, «Named-entity recognition for norwegian», in *Proceedings of the 22nd Nordic Conference on Computational Linguistics*, 2019, pp. 222–231.

[JSL18] K. K. Jakobsen, U. Stridbeck, and Å. Langballe, «Objektivitet i avhør: Avhør av fornærmede i straffesaker i norge», *Tidsskrift for strafferett*, vol. 18, no. 2, pp. 74–101, 2018.

[Jus20] Justis- og beredskapsdepartementet, Politimeldingen - et politi for fremtiden , Meld. St. 29 (2019-2020), 2020. [Online]. Available: https://www.regjeringen.no/no/dokumenter/meld.-st.-29-20192020/id2715224/ (last visited: Jun. 18, 2023).

[KDWB21] P. E. Kummervold, J. De la Rosa, *et al.*, «Operationalizing a national digital library: The case for a norwegian transformer model», *arXiv preprint arXiv:2104.09617*, 2021.

[KFM21] A. Kuster, J. Filipek, and V. V. Muppirala, «Reproducing" ner and pos when nothing is capitalized"», *arXiv preprint arXiv:2109.08396*, 2021.

[Kla] Klarspråk, Forkortelser. [Online]. Available: https://www.sprakradet.no/klarsprak/om-skriving/sprak-i-lover-og-forskrifter/skriverad/rettskriving/forkortelser/ (last visited: Jun. 16, 2023).

[KLC+19] J. Y. Kim, C. Liu, *et al.*, «A comparison of online automatic speech recognition systems and the nonverbal responses to unintelligible speech», *arXiv preprint arXiv:1904.12403*, 2019.

[KLW19] U. Kamath, J. Liu, and J. Whitaker, *Deep learning for NLP and speech recognition*. Springer, 2019.

[MAHW13] J. Monckton-Smith, T. Adams, *et al.*, *Introducing Forensic and Criminal Investigation*. Sage, 2013, p. 46.

[MCC19] A. F. Miranda, P. Chitale, and J. Cohen, How to Build Domain Specific Automatic Speech Recognition Models on GPUs, Dec. 17, 2019. [Online]. Available: https://developer.nvidia.com/blog/how-to-build-domain-specific-automatic-speech-recognition-models-on-gpus/ (last visited: May 22, 2023).

[MMG04] A. C. Morris, V. Maier, and P. Green, «From wer and ril to mer and wil: Improved evaluation measures for connected speech recognition», in *Eighth International Conference on Spoken Language Processing*, 2004.

[Moe22] M. K. Moe, «Post-processing automatic speech recognition transcriptions: A study for investigative interviews», Department of Information Security and Communication Technology, NTNU – Norwegian University of Science and Technology, Project report in TTM4502, Dec. 2022.

[Moh14] B. Mohit, «Named entity recognition», *Natural language processing of semitic languages*, pp. 221–245, 2014.

[MPR02] M. Mohri, F. Pereira, and M. Riley, «Weighted finite-state transducers in speech recognition», *Computer Speech & Language*, vol. 16, no. 1, pp. 69–88, 2002.

[MTR19] S. Mayhew, T. Tsygankova, and D. Roth, «Ner and pos when nothing is capitalized», *arXiv preprint arXiv:1903.11222*, 2019.

[NB a] NB AI-lab, Models. [Online]. Available: https://ai.nb.no/models/ (last visited: Apr. 13, 2023).

[NB b] NB AI-lab, nb-bert-base-ner. [Online]. Available: https://huggingface.co/NbAiLab/nb-bert-base-ner (last visited: May 2, 2023).

[NbA22] NbAiLab, Norwegian Wav2Vec2 Model - 1B Bokmål, 2022. [Online]. Available: https://huggingface.co/NbAiLab/nb-wav2vec2-1b-bokmaal (last visited: Apr. 13, 2023).

[ND21] M. Negrão and P. Domingues, «Speechtotext: An open-source software for automatic detection and transcription of voice recordings in digital forensics», *Forensic Science International: Digital Investigation*, vol. 38, p. 301 223, 2021.

[Nua] Nuance, Dragon Law Enforcement police reporting software. [Online]. Available: https://www.nuance.com/dragon/industry/dragon-law-enforcement.html (last visited: May 31, 2023).

[NVI] NVIDIA CORPORATION, Text (Inverse) Normalization. [Online]. Available: https://docs.nvidia.com/deeplearning/nemo/user-guide/docs/en/main/nlp/text_normalization/wfst/wfst_text_normalization.html (last visited: Apr. 18, 2023).

[Ope22] OpenAI, Introducing Whisper, 2022. [Online]. Available: https://openai.com/blog/whisper/ (last visited: Feb. 13, 2023).

[Phi] Philips Dictation, Accomplishing critical tasks with a seamless voice solution. [Online]. Available: https://www.dictation.philips.com/gb/success-stories/success/accomplishing-critical-tasks-with-a-seamless-voice-solution/ (last visited: May 24, 2023).

[Pol16] Politidirektoratet, Handlingsplan for løft av etterforskningsfeltet, 2016. [Online]. Available: https://www.politiet.no/globalassets/05-om-oss/03-strategier-og-planer/handlingsplan-for-loft-av-etterforskningsfeltet.pdf (last visited: Jun. 18, 2023).

[Rik13] Riksadvokatens arbeidsgruppe, Avhørsmetodikk i politiet, 2013. [Online]. Available: https://www.riksadvokaten.no/wp-content/uploads/2017/10/Avh%C3%B8rsmetodikk-i-Politiet-med-vedlegg.pdf (last visited: Apr. 19, 2023).

[RKX+22] A. Radford, J. W. Kim, *et al.*, «Robust speech recognition via large-scale weak supervision», *arXiv preprint arXiv:2212.04356*, 2022.

[RW01] P. Roach and H. Widdowson, *Phonetics*. Oxford University Press, 2001.

[SO22] P. E. Solberg and P. Ortiz, «The norwegian parliamentary speech corpus», *arXiv preprint arXiv:2201.10881*, 2022.

[SOP+23] P. E. Solberg, P. Ortiz, *et al.*, «Improving generalization of norwegian asr with limited linguistic resources», in *Proceedings of the 24th Nordic Conference on Computational Linguistics (NoDaLiDa)*, 2023, pp. 508–517.

[Spr] Språkbanken, The Norwegian Language Bank, A national infrastructure for language technology. [Online]. Available: https://www.nb.no/sprakbanken/en/sprakbanken/ (last visited: May 4, 2023).

[Spr13] Språkbanken, "Exhibition Corpus" – Text, Sound, Sign, 2013. [Online]. Available: https://www.nb.no/sprakbanken/en/resource-catalogue/oai-nb-no-sbr-30/ (last visited: Apr. 26, 2023).

[SSBK21] M. Sunkara, C. Shivade, *et al.*, «Neural inverse text normalization», in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2021, pp. 7573–7577.

[TBK+23] S. Tan, P. Behre, *et al.*, «Four-in-one: A joint approach to inverse text normalization, punctuation, capitalization, and disfluency for automatic speech recognition», in *2022 IEEE Spoken Language Technology Workshop (SLT)*, IEEE, 2023, pp. 677–684.

[VÁ23] J. C. Vásquez-Correa and A. Álvarez Muniain, «Novel speech recognition systems applied to forensics within child exploitation: Wav2vec2. 0 vs. whisper», *Sensors*, vol. 23, no. 4, p. 1843, 2023.

[VB16] J. Vajpai and A. Bora, «Industrial applications of automatic speech recognition systems», *International Journal of Engineering Research and Applications*, vol. 6, no. 3, pp. 88–95, 2016.

[WDMH17] S. Watanabe, M. Delcroix, *et al.*, «New era for robust speech recognition», *Springer International Publishing. doi*, vol. 10, pp. 978–3, 2017.

[WHK+17] S. Watanabe, T. Hori, *et al.*, «Hybrid ctc/attention architecture for end-to-end speech recognition», *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, no. 8, pp. 1240–1253, 2017.

[WWL19] D. Wang, X. Wang, and S. Lv, «An overview of end-to-end automatic speech recognition», *Symmetry*, vol. 11, no. 8, p. 1018, 2019.

[YD16] D. Yu and L. Deng, *Automatic Speech Recognition, A Deep Learning Approach*. Springer, 2016.

[ZBGG21] Y. Zhang, E. Bakhturina, *et al.*, «Nemo inverse text normalization: From development to production», *arXiv preprint arXiv:2104.05055*, 2021.

Appendix A
Complete Evaluation Tables

Filename	Language	%WER	%CER
02.txt	nn	58.23	15.11
03.txt	nb	27.11	5.89
04.txt	nn	47.68	12.84
05.txt	nb	29.08	13.04
06.txt	nb	27.33	8.03
07.txt	nb	24.40	7.40
08.txt	nb	18.88	5.03
09.txt	nn	51.46	22.96
10.txt	nn	39.46	11.74
11.txt	nb	20.96	4.95
12.txt	nb	25.52	8.79
13.txt	nb	29.37	8.21
14.txt	nb	20.00	5.90
15.txt	nn	40.29	13.22
16.txt	nb	36.14	11.12
17.txt	nn	43.31	13.28
18.txt	nn	46.34	14.60
19.txt	nb	31.74	9.93
20.txt	nb	29.27	10.44
21.txt	nb	24.80	6.60
22.txt	nb	27.78	7.06
23.txt	nn	58.25	19.54
24.txt	nn	46.09	14.08

Table A.1: Evaluation of the Wav2vec 2.0 output from the Exhibition Corpus. The table also shows the written language, Norwegian Bokmål (nb) or Norwegian Nynorsk (nn), of the reference transcription.

Filename	Language	%WER	%CER
02.txt	nn	48.73	10.95
03.txt	nb	26.51	5.57
04.txt	nn	49.67	14.25
05.txt	nb	22.45	7.79
06.txt	nb	24.42	6.33
07.txt	nb	22.02	4.55
08.txt	nb	18.88	5.03
09.txt	nn	47.37	9.72
10.txt	nn	39.46	11.74
11.txt	nb	20.96	4.95
12.txt	nb	25.00	6.89
13.txt	nb	30.07	7.34
14.txt	nb	20.00	5.90
15.txt	nn	38.83	10.86
16.txt	nb	31.33	8.24
17.txt	nn	43.31	12.76
18.txt	nn	46.34	14.86
19.txt	nb	32.54	8.93
20.txt	nb	26.83	6.66
21.txt	nb	23.20	6.31
22.txt	nb	27.78	7.06
23.txt	nn	60.19	20.15
24.txt	nn	46.09	14.08

Table A.2: Evaluation of the transcriptions of the Exhibition Corpus after all post-processing steps were applied. The table also shows the written language, Norwegian Bokmål (nb) or Norwegian Nynorsk (nn), of the reference transcription.

Appendix B
Example of text 02

Reference Transcription: Om konkurransen
Hausten 2012 inviterte Nasjonalbiblioteket elevar på 5.–10. trinn og elevar på vidaregåande skular frå heile landet til å delta i ein stor språkkonkurranse. Temaet for konkurransen var språk og identitet, og elevane skulle dokumentere dei ulike språka og dialektane dei møter i kvardagen. Det var opp til elevane å avgjere kva dei ønskte å definere som språk.
Bidraga blei vurderte etter originalitet, kreativitet, om dei ga ny informasjon om språk, og tilrettelegging, altså om bidraga var lagt til rette for eiga aldersgruppe.
Juryen bestod av juryleiar Arnfinn Muruvik Vonen som er direktør i Språkrådet; Lisa Baal, Senter for samisk i opplæringa; Hanne Haugli, NAFO; Johanne Ostad, Språkbanken; Inger Johanne Sæterbakk, Språkåret 2013; Helene Uri, forfattar og språkvitar – og Liv Kristin Bjørlykke Øvereng, Nasjonalt senter for nynorsk i opplæringa.
Konkurransen blei utarbeidd saman med dei nasjonale sentra for fleirkulturell opplæring, framandspråk, matematikk, nynorsk og samisk.
Vi fekk 77 bidrag til konkurransen. I utstillinga ser du vinnerbidraga og fleire andre gode bidrag!

ASR Output Wav2vec 2.0: om konkurransen høsten tjuetolv invitert til nasjonalbiblioteket elever på femte til tiende trinn og elever på videregående skoler fra hele landet til å delta i en stor språkkonkurranse temaet for konkurransen var språk og identitet og elevene skulle dokumentere de ulike språka og dialektene de møter i hverdagen det var opp til elevene å avgjøre hva det ønste å definere som språk bidraga ble vurdert etter originalitet kreativitet om de ga ny informasjon om språk og tilrettelegging altså om bidraget var lagt til rette for eiga aldersgruppejuriem besto av jurileier arnfinn muruvik vonen som er direktør i språkrådet lisa bal senter for samisk i opplæringa hanne haugli nafo johanne ostad språkbanken inger johanne sæterbakk språkåret tjuetretten helene uri forfatter og språkviter og liv kristin bjørlykke øvereng nasjonalt senter for nynorsk i opplæringkonkurransen ble utarbeidet sammen med de

nasjonale sentre for flerkulturell opplæring framandspråk matematikk nynorsk og samisk vi fikk syttisju bidrag til konkurransen i utstillinga ser du vinnerbidragene og flere andre gode bidrag

Transcription After Post-processing: om konkurransen høsten 2012 invitert til nasjonalbiblioteket elever på femte til tiende trinn og elever på videregående skoler fra hele landet til å delta i en stor språkkonkurranse temaet for konkurransen var språk og identitet og elevene skulle dokumentere de ulike språka og dialektene de møter i hverdagen det var opp til elevene å avgjøre hva det ønste å definere som språk bidraga ble vurdert etter originalitet kreativitet om de ga ny informasjon om språk og tilrettelegging altså om bidraget var lagt til rette for eiga aldersgruppejuriem besto av jurileier Arnfinn Muruvik Vonen som er direktør i språkrådet Lisa Bal, Senter for samisk i opplæringa, Hanne Haugli Nafo Johanne Ostad språkbanken Inger Johanne Sæterbakk språkåret 2013 Helene Uri forfatter og språkviter og Liv Kristin Bjørlykke Øvereng nasjonalt senter for nynorsk i opplæringkonkurransen ble utarbeidet sammen med de nasjonale sentre for flerkulturell opplæring framandspråk matematikk nynorsk og samisk vi fikk 77 bidrag til konkurransen i utstillinga ser du vinnerbidragene og flere andre gode bidrag

Words converted by the ITN post-processing step are marked in magenta, text capitalized by the NER model are written in orange, while punctuation added by the NER model are shown in blue.

Appendix C
NER Precision and Recall

Filename	Proper Nouns in Reference	Correct Tokens by NER	Recall
02.txt	25	17	0.68
03.txt	3	3	1.00
04.txt	5	0	0
05.txt	6	6	1.00
06.txt	11	5	0.45
07.txt	7	5	0.71
08.txt	0	0	-
09.txt	6	3	0.50
10.txt	0	0	-
11.txt	0	0	-
12.txt	10	2	0.20
13.txt	5	0	0
14.txt	4	0	0
15.txt	9	5	0.56
16.txt	7	5	0.71
17.txt	5	0	0
18.txt	4	2	0.50
19.txt	3	1	0.33
20.txt	12	7	0.58
21.txt	2	2	1.00
22.txt	1	0	0
23.txt	3	0	0
24.txt	0	0	-
Avg. Recall			0.43

Table C.1: Proper nouns in reference transcription and correctly labeled tokens by the NER model for the Wav2vec 2.0 transcription. There were no falsely labeled tokens, therefore precision is 100% and no column for this is added in the table. The average recall is calculated by taking the average of the 19 texts with proper nouns in them.

Filename	Proper Nouns in Reference	Correct Tokens after Divided_ITN_NER_NER	Recall
02.txt	25	20	0.80
03.txt	3	3	1.00
04.txt	5	0	0
05.txt	6	6	1.00
06.txt	11	6	0.55
07.txt	7	6	0.86
08.txt	0	0	-
09.txt	6	4	0.67
10.txt	0	0	-
11.txt	0	0	-
12.txt	10	3	0.30
13.txt	5	0	0
14.txt	4	0	0
15.txt	9	4	0.44
16.txt	7	5	0.71
17.txt	5	0	0
18.txt	4	2	0.50
19.txt	3	1	0.33
20.txt	12	11	0.92
21.txt	2	2	1.00
22.txt	1	0	0
23.txt	3	0	0
24.txt	0	0	-
Avg. Recall			0.48

Table C.2: Proper nouns in reference transcription and correctly capitalized words in the final post-processed transcription. The average recall is calculated by taking the average of the 19 texts with proper nouns in them.

Appendix D
Code

Algorithm D.1 Code for utilizing the nb-bert-base-ner model.

```
# Following line in a separate code cell.
!pip install transformers

from transformers import AutoTokenizer,
    AutoModelForTokenClassification
from transformers import pipeline

tokenizer = AutoTokenizer.from_pretrained("NbAiLab/nb-bert-
    base-ner")
model = AutoModelForTokenClassification.from_pretrained("
    NbAiLab/nb-bert-base-ner")
group_entities = True

ner = pipeline("ner", model=model, tokenizer=tokenizer,
    grouped_entities=group_entities)

# The model is ready to be used by a function call to
    ner(text) with a string as input.
```

Algorithm D.2 Code for capitalizing proper nouns in the dataset utilizing a ner model.

```
import os

input_directory = "wav2vec_text"
output_directory = "ner"

# Iterate through all text files in the dataset.
for file in os.listdir(input_directory):
    filenamepath = os.path.join(input_directory, file)
    text = open(filenamepath, "r", encoding="utf8").read()

    # Apply NER model.
    results = ner(text)

    # Make list to index the different characters.
    output = list(text)
    for token in results:
        hashtag_count = token["word"].count("#")
        # More than one word, capitilizes start of every
            word.
        if "␣" in token["word"]:
            # First word should be capitilized
            capitilize = True
            # First word starts in the middle of the word,
                should not be capitilized.
            if hashtag_count.
                capitilize = False
            for i in range(token["start"], token["end"]):
                if capitilize:
                    output[i] = output[i].upper()
                    capitilize = False
                if output[i] == "␣":
                    capitilize = True

        # Only one word.
        else:
            # Only end of a word, should not be capitilized.
            if hashtag_count:
                continue
            output[token["start"]] = output[token["start"
                ]].upper()
    # Saves capitalized text to a new text file.
    with open(os.path.join(output_directory, file), "w") as
        txtfile:
        txtfile.write("".join(output))
```

Algorithm D.3 Code for adding commas to a list of proper nouns.

```
# Example text to add commas to.
text = "Jeg liker mange byer for eksempel Trondheim Bergen
    Oslo og Tromsø."

# Apply NER model.
ner_tokens = ner(text)
output = list(text)

for i in range(len(ner_tokens) - 1):
    current_token = ner_tokens[i]
    next_token = ner_tokens[i + 1]

    if current_token["end"] == next_token["start"]-1 and not
        next_token["word"].startswith("#"):
      output[current_token["end"]] = ", "

new_text = "".join(output)
```

Algorithm D.4 Simplified code for timing a function: function().

```
import time

# Dicitionary will consist of filenames as keys and elapsed
    time as values.
speed = {}

for filename in directory:
    start = time.perf_counter()
    result = function(filename)
    end = time.perf_counter()

    # Saves elapsed time with 3 decimals.
    speed[filename] = round(end-start, 3)

print(speed)
```

Algorithm D.5 Code for calculating the average and standard deviation of the speed dictionary.

```
from statistics import mean, stdev

# Example of elapsed time of running ITN on Wav2vec 2.0
    output.
speeds = {'02.txt': 0.349, '03.txt': 0.361, '04.txt': 0.343,
     '05.txt': 0.446, '06.txt': 0.377, '07.txt': 0.36, '08.
    txt': 0.295, '09.txt': 0.382, '10.txt': 0.398, '11.txt':
    0.389, '12.txt': 0.449, '13.txt': 0.348, '14.txt': 0.304,
     '15.txt': 0.461, '16.txt': 0.347, '17.txt': 0.336, '18.
    txt': 0.504, '19.txt': 0.289, '20.txt': 0.413, '21.txt':
    0.267, '22.txt': 0.265, '23.txt': 0.218, '24.txt': 0.255}

average = mean(speeds.values())
standard_deviation = stdev(speeds.values())
```

Algorithm D.6 Code for evaluating one text file.

```
import os
from jiwer import wer, cer

def evaluate_file_in_directory(filename, hypothesisdir,
    referencedir):
    # Reading reference transcription of the text file.
    r = open(os.path.join(referencedir, filename), "r",
        encoding="utf8").read()
    # Reading text file to be evaluated.
    h = open(os.path.join(hypothesisdir, filename), "r",
        encoding="utf8").read()

    # Calculating WER.
    wer_metric = wer(r,h)

    # Calculating CER.
    cer_metric = cer(r,h)

    # Printing results.
    print("WER: "+str(wer_metric)+" CER: "+str(cer_metric))

# Directory to find reference transcription in.
refdir = "tekst"

# Directory to find text to be evaluated in.
hypdir = "itn_divided"

# Name of the text file to be evaluated.
filename = "05.txt"

evaluate_file_in_directory(filename, hypdir, refdir)
```

Algorithm D.7 Code for evaluating multiple text files in a directory.

```
import os
from jiwer import wer, cer

def evaluate_whole_directories(referencedir, hypothesisdir,
    save_as):
    # Overview of which reference texts are in Bokmål and
        Nynorsk (1=nb, 0=nn).
    is_bokmaal =
        [0,1,0,1,1,1,1,0,0,1,1,1,1,0,1,0,0,1,1,1,1,0,0]

    with open(save_as+".txt", "a") as evaluation_file:
        evaluation_file.write("Filename\tWER\t\tCER\tNB/NN\n
            ")
        for filename in os.listdir(hypothesisdir):
            hypothesispath = os.path.join(hypothesisdir,
                filename)
            referencepath = os.path.join(referencedir,
                filename)
            if os.path.isfile(hypothesispath):
                h = open(hypothesispath, "r", encoding="utf8
                    ").read()
                r = open(referencepath, "r", encoding="utf8"
                    ).read()
                wer_metric = wer(r,h)
                cer_metric = cer(r,h)
                # Index of list starts at 0, filename number
                     starts at 2.
                if is_bokmaal[int(filename.strip(".txt"))
                    -2]:
                    label = "nb"
                else:
                    label = "nn"
                evaluation_file.write(filename +","+ str(
                    wer_metric) +","+ str(cer_metric) +","+
                    label +"\n")

# Directory to find reference transcriptions in.
refdir = "tekst"

# Directory to find texts to be evaluated in.
hypdir = "divided_itn_ner_ner"

# Name to save the evaluation table to.
save_as = "e_divided_itn_ner_ner"

evaluate_whole_directories(refdir, hypdir, save_as)
```

Algorithm D.8 Code for calculating average or standard deviation from evaluation table file.

```
from statistics import mean, stdev

def calculate(file, type):
    dict = {}
    dict["nb"] = {"wer":[], "cer":[]}
    dict["nn"] = {"wer":[], "cer":[]}

    # Loading data from file, stores in dict corresponding
        to written language of the text file
    with open(file, "r") as baseline:
        for line in baseline:
            if line.startswith("Filename"):
                continue
            values = line.split(",")
            if values[3] == "nb\n":
                dict["nb"]["wer"].append(float(values[1]))
                dict["nb"]["cer"].append(float(values[2]))
            else:
                dict["nn"]["wer"].append(float(values[1]))
                dict["nn"]["cer"].append(float(values[2]))

    if type == "mean": # Average
        wer = mean(dict["nb"]["wer"]+dict["nn"]["wer"])
        nb_wer = mean(dict["nb"]["wer"])
        nn_wer = mean(dict["nn"]["wer"])

        cer = mean(dict["nb"]["cer"]+dict["nn"]["cer"])
        nb_cer = mean(dict["nb"]["cer"])
        nn_cer = mean(dict["nn"]["cer"])

    elif type == "stdev": # Standard deviation
        wer = stdev(dict["nb"]["wer"]+dict["nn"]["wer"])
        nb_wer = stdev(dict["nb"]["wer"])
        nn_wer = stdev(dict["nn"]["wer"])

        cer = stdev(dict["nb"]["cer"]+dict["nn"]["cer"])
        nb_cer = stdev(dict["nb"]["cer"])
        nn_cer = stdev(dict["nn"]["cer"])

    return wer, nb_wer, nn_wer, cer, nb_cer, nn_cer

filename = "e_divided_itn_ner_ner.txt"
type = "mean"

totwer, nbwer, nnwer, totcer, nbcer, nncer = calculate(
    filename, type)
```

Algorithm D.9 Code for generating box plot of baseline.

```
import matplotlib.pyplot as plt

dict = {}
dict["nb"] = {"wer":[], "cer":[]}
dict["nn"] = {"wer":[], "cer":[]}

# Loading data from file, stores in dict corresponding to
    written language of the text file.
with open("e_baseline.txt", "r") as baseline:
    for line in baseline:
        if line.startswith("Filename"):
            continue
        values = line.split(",")
        if values[3] == "nb\n":
            dict["nb"]["wer"].append(float(values[1]))
            dict["nb"]["cer"].append(float(values[2]))
        else:
            dict["nn"]["wer"].append(float(values[1]))
            dict["nn"]["cer"].append(float(values[2]))

# Change font of plot.
font = {'size':14}
plt.rc('font', **font)

# Choose "wer" or "cer".
type = "cer"

data = [dict["nb"][type]+dict["nn"][type], dict["nb"][type],
    dict["nn"][type]]
fig,ax = plt.subplots(figsize =(10, 7))

# Creating plot.
bp = ax.boxplot(data, labels=["Total", "Bokmål", "Nynorsk"])

# To make the title.
plt.title("Box Plot of Baseline " + type.upper())

# Show plot.
plt.show()
```

Algorithm D.10 Code for transcribing audio files with the Whisper Medium model in Google Colab.

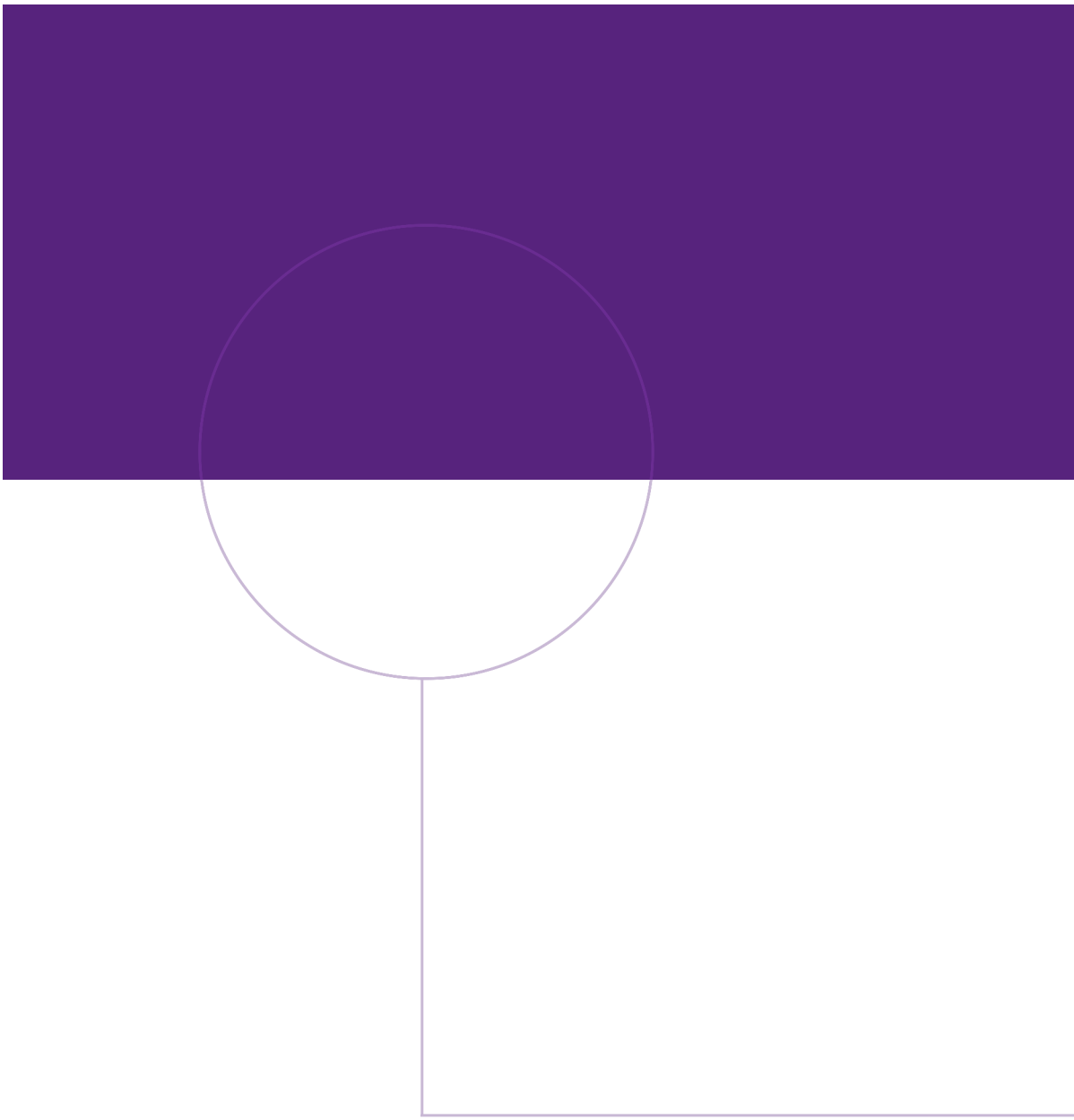
```
# The two following lines were run in a separate code cell.
!pip install git+https://github.com/openai/whisper.git
!sudo apt update && sudo apt install ffmpeg

import whisper, os
model = whisper.load_model("medium")

input_directory = "audio"
output_directory = "medium"

for filename in os.listdir(input_directory):
    f = os.path.join(input_directory, filename)
    if os.path.isfile(f):
        result = model.transcribe(f)
        with open(os.path.join(output_directory, filename.
            replace(".mp3",".txt")), "w") as txtfile:
            txtfile.write(result['text'])
```

Appendix E

Overview of the Exhibition Corpus

Filename	Proper Nouns	Numerals	Numbers in Letters
02.txt	25	5	0
03.txt	3	0	1
04.txt	5	0	1
05.txt	6	7	1
06.txt	11	2	1
07.txt	7	2	0
08.txt	0	0	0
09.txt	6	12	0
10.txt	0	0	0
11.txt	0	0	0
12.txt	10	3	2
13.txt	5	1	2
14.txt	4	0	0
15.txt	9	2	0
16.txt	7	2	0
17.txt	5	1	0
18.txt	4	1	0
19.txt	3	1	2
20.txt	12	3	0
21.txt	2	0	0
22.txt	1	0	0
23.txt	3	1	2
24.txt	0	0	0

Table E.1: Counts of proper nouns and numbers (written with numbers or letters) for the reference texts in the Exhibition Corpus.



NTNU

Norwegian University of
Science and Technology