Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Electronic Systems

Solveig Reppen Lunde

Modeling the Interpretability of an End-to-End Automatic Speech Recognition System

Adapted to Norwegian Speech

Master's thesis in Electronics System Design and Innovation
Supervisor: Giampiero Salvi
Co-supervisor: Pablo Ortiz
June 2022

NTNU
Norwegian University of Science and Technology

telenor

Solveig Reppen Lunde

Modeling the Interpretability of an End-to-End Automatic Speech Recognition System

Adapted to Norwegian Speech

Master's thesis in Electronics System Design and Innovation
Supervisor: Giampiero Salvi
Co-supervisor: Pablo Ortiz
June 2022

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Electronic Systems

NTNU
Norwegian University of
Science and Technology

Norwegian University of Science and Technology

TFE4940

Electronics Systems Design and Innovation,
Master's thesis

Modeling the Interpretability of an End-to-End Automatic Speech Recognition System Adapted to Norwegian Speech

Author:
Solveig Reppen Lunde

Supervisor:
Giampiero Salvi

Co-supervisor:
Pablo Ortiz

June 6, 2022

Abstract

This project aims to model the interpretability of an automatic speech recognition (ASR) system adapted to Norwegian speech. The ASR system is an end-to-end (E2E) deep neural network that takes speech signals as input and is trained to output orthographic text. By performing phoneme and grapheme classification, we investigated how phonetic and orthographic information is encoded in the model layers. Our results show that the phonetic representation improves for the lower layers but is degraded for the higher layers. The orthographic representation improves gradually for the higher layers, with a considerable improvement for the last layer. This indicates that the last layers are more geared towards graphemes, which is reasonable since it is trained to output orthographic text. Our results indicate that the model learns phonetic representation implicitly, even though there is nothing with the training method that forces it to learn phonetic representation.

For further analysis of the model's interpretability, we investigated whether the information encoded in the model is dialect-dependent, by testing the ASR model on spontaneous speech from twelve different dialect groups. The dialects from the southeastern parts of Norway achieved the lowest error rates (and thus the best results), while the dialects from the middle and western parts of Norway gave the highest error rates. The results seem to have a clear correspondence with dialectal variation since the best recognized dialects are the ones being closest to Bokmål, while the worst recognized are those that are most distinct from Bokmål. This was verified by analyzing the recognition of some functional words and verbs that have distinctive dialectal forms and a high occurrence in the Norwegian language. The results show that the model achieves a low error rate when the words are pronounced similarly to the Bokmål form, while it struggles with transcribing dialectal forms that deviate clearly from the Bokmål form. We find this reasonable since the ASR model is fine-tuned on read-aloud Bokmål text. For improving ASR systems for Norwegian dialects, we propose including more training data from spontaneous speech, which involves more dialect-specific words, and having a more balanced amount of Nynorsk and Bokmål data for training the language model.

Sammendrag

Formålet med dette arbeidet var å modellere tolkbarheten til et automatisk talegjenkjenningssystem trent på norsk tale. Systemet er et ende-til-ende dypt nevralt nettverk som tar inn taledata og er trent for å gi ut ortografisk tekst. Gjennom fonem- og grafemklassifikasjon analyserte vi den fonetiske og ortografiske representasjonen av de ulike lagene til det dype nettverket. Resultatene våre viser at den fonetiske representasjonen forbedres for de nederste lagene, før den degraderes for de øverste lagene av systemet. Den ortografiske representasjonen forbedres gradvis jo høyere opp i systemet vi kommer, med en betydelig forbedring for det siste laget. Dette indikerer at de øverste lagene er mer fokusert på grafemer, som er plausibelt siden systemet er trent til å gi ut ortografisk tekst. Resultatene våre indikerer at systemet implisitt lærer fonetisk representasjon, selv om dette ikke blir lært direkte gjennom treningsmetoden - den fokuserer på den ortografiske representasjonen.

For en videre analyse av systemets funksjonalitet undersøkte vi om informasjonen som er lagret i modellen er dialektavhengig, ved å teste systemet på spontan tale fra tolv ulike dialektgrupper. Dialektene fra de sørøstlige områdene av Norge ga lavest feilrate, mens dialektene fra Vest- og Midt-Norge ga høyest feilrate. Resultatene ser ut til å ha en tydelig sammenheng med dialektvariasjoner, da dialektene som ga lavest feilrate er de som har flest fellestrekk med Bokmål, mens de med høyest feilrate er de som skiller seg mest fra Bokmål. Dette ble verifisert ved å analysere enkelte funksjonsord og verb som brukes mye i det norske språket, og som varierer betydelig mellom ulike dialekter. Resultatene viser at systemet oppnår en lav feilrate når ordene er uttalt på lignende måte som den skriftlige Bokmålsformen, men har betydelige problemer med å gjenkjenne ordet for dialektvarianter som skiller seg ut fra Bokmålsformen. Dette er rimelig da systemet er finjustert på taledata fra opplest Bokmåltekst. For å forbedre talegjenkjenningssystemer for norske dialekter, anbefaler vi at treningsdataen består av mer spontan tale som innebærer flere dialektuttrykk, og at språkmodellene som brukes til dekoding burde bli trent med en mer balansert mengde Nynorsk og Bokmål tekst.

Contents

List of Figures

List of Tables

Acronyms

AM acoustic model.

ANN artificial neural network.

ASR automatic speech recognition.

BLSTM bidirectional long short-term memory.

CER character error rate.

CNN convolutional neural network.

CTC connectionist temporal classification.

DNN deep neural network.

E2E end-to-end.

GMM Gaussian mixture model.

GRU gated recurrent unit.

HMM hidden Markov model.

LM language model.

LSTM long short-term memory.

ML machine learning.

NN neural network.

NPSC Norwegian Parliament Speech Corpus.

NST Nordisk Språkteknologi.

PM pronunciation model.

RNN recurrent neural network.

seq2seq sequence-to-sequence.

WER word error rate.

Chapter 1

Preface

This report is part of the SCRIBE project[1], which is working with improving the performance of ASR systems adapted to the Norwegian language. It is a collaboration between NTNU, Telenor Research, the National Library of Norway, NRK, and the Norwegian Open AI Lab. The aim of the project is to develop a Norwegian speech-to-text transcription system for multi-party conversations in realistic recording conditions.

The report is a continuation of the work done by Lunde *et al.* [1], where we analyzed the phonetic representations of an ASR model adapted to Norwegian speech. The ASR model we analyzed was an E2E model based on Deep Speech 2 [2], a family of ASR systems based on deep neural networks (DNNs) and the CTC framework [3], which is trained to predict a sequence of graphemes based on a given input speech signal. The model architecture consists of three strided convolutional layers, nine 1200-dimensional, bidirectional GRU layers, and a fully connected layer with softmax. The AM is sequentially trained with 300h of NST data[2], 40h of NPSC data[3], and 12h of internal Telenor data from their customer service center.

E2E models may be easier to implement than conventional ASR systems, but it can be hard to model their interpretability. We conducted the study by analyzing the activations from the nine recurrent layers of the deep model for investigation of how well the different layers represent phonetic information. The activations were fed into a supervised classifier for frame-level phoneme classifica-

[1]More information available at `https://scribe-project.github.io/`.

[2]Datasets and documentation are available at `https://www.nb.no/sprakbanken/en/resource-catalogue/oai-nb-no-sbr-13/`.

[3]Datasets and documentation are available from `https://www.nb.no/sprakbanken/en/resource-catalogue/oai-nb-no-sbr-58/`. Note that at the time of training the base model, only 58h were available, in contrast to the 140h available at the time of writing.

tion. Our results showed that the phonetic representation was improved for the first layers, but then the performance decreased monotonically after the middle layers. The results are indicating that the model is doing better with more parameters in terms of phoneme recognition for the first layers, but then it changes to accommodate the target classes.

Chapter 2

Introduction

Norway is a country with approximately 5.4 million citizens, and the majority speak Norwegian. Despite the country's small population, spoken Norwegian consists of a large variety of different dialects. This is mainly caused by the country's topography. Norway has an elongated landscape, and people have been separated by mountains, forests, and fjords until modern times, which naturally has led to dialectal variations. There is no official spoken form of Norwegian. Thus, people are talking in their dialect in casual speech, broadcasts, teaching, and in the Parliament, to give a few examples. Though many dialects are very distinctive, Norwegians can understand other dialects than their own, since they are exposed to different dialects in all kinds of settings. There are two official written forms of Norwegian: Bokmål and Nynorsk. Bokmål originates from the Danish language because Norway and Denmark were in a union for more than 400 years until 1814. After the union was dissolved, many Norwegians wanted to "take back" the original Norwegian language. Ivar Asen traveled across the whole country to gather different dialects, which he sorted and systematized to create a new written form, namely Nynorsk. Dialects from the urban eastern parts of Norway are usually more similar to Bokmål, while dialects from the rural southern parts of Norway, especially those from the west, are more consistent with Nynorsk [4].

Knowledge about the Norwegian language is important when working with automatic speech recognition (ASR) systems. An ASR system takes audio data as input and tries to recognize the spoken utterance. For example, ASR systems can be used to transcribe speech data into text. State-of-the-art ASR systems are now performing well for the most spoken languages in the world under the right conditions (e.g., noise-free conditions and read speech), but suffer from a lack of performance for smaller languages. One of the main reasons for this is

that ASR systems require a huge amount of data for training. There are many corpora consisting of hundreds or even thousands of hours of annotated English speech. This is unfortunately not the case for smaller languages, like Norwegian. Though many people are talking English as their second language, there is important to develop ASR systems that perform well in smaller languages. In many applications, it is necessary to have a well-working system in the local language. For example, many immigrants have problems learning Norwegian due to the wide dialect variety. In Norwegian language courses, it is most common to learn to speak similarly as they do in the urban eastern parts of Norway and to write Bokmål. If the immigrants live in another part of the country, they are exposed to a different dialect when meeting people in their hometown. The vocabulary of those dialects may be quite distinct from Urban East Norwegian. A well-working ASR system implemented as a mobile application would be a good way of learning how words are pronounced in different dialects. Additionally, people with physical disabilities can benefit from such ASR systems. Automatic real-time transcription of TV broadcasts and speech-to-text applications for people who are unable to write are two examples.

2.1 Research goal

The goal of this thesis is to model the interpretability of an end-to-end ASR system that transcribes speech signals into graphemes. This is a fundamental task in improving end-to-end models, as will be presented in Section 3.1. The system is adapted to the Norwegian language and its many dialects, as it has been trained with corpora that contain a rich variety of dialects. Since it is a deep end-to-end model, we will try to model what information is encoded in the model layers. Firstly, we will compare the phonetic and orthographic representations in the layers. To do this, we will extract phonetic and orthographic information from the different layers separately, to see if intermediate features computed by the network are well aligned with phonetic or orthographic classes, respectively. The model is not explicitly optimized for modeling phonemes. However, we hypothesize that the internal representations must contain phonetic information in order to solve the transcription task. Since the model takes in acoustic features and outputs graphemes, we expect that the model switches from focusing on phonemes to focusing on graphemes in the last layers.

The work of analyzing the model's phonetic representations was done in our previous work, Lunde *et al.* [1], using framewise classification. In this report, we describe our analysis of the model's orthographic representations. We use the activations from different layers of the model to train supervised classifiers to output orthographic transcriptions. We examine the different layers' performance in terms of word error rate (WER) and character error rate (CER), which will give

information about the layer's orthographic representations. The obtained results are compared with the phonetic representations of the ASR model, to obtain a deeper understanding of how information is encoded in the model layers.

For further analysis of the information encoded in the model, we investigate whether the information is dialect-dependent. We test the model on both native and non-native speakers and analyze the outputs. The main emphasis is placed on analyzing the impact of dialectal Norwegian speech in terms of model performance. We compare the resulting error rates between different dialect groups to see if some dialects are more easily recognized by the model. Further, we examine whether the differences are caused by dialectal variations. Hopefully, the information obtained from the analyses can be used in future work to improve ASR systems for Norwegian speech, e.g., in terms of what kind of data is necessary to train the ASR model.

Based on the preceding introduction, we formulate the following research questions.

Research question 1: To what extent does the model capture phonetic and orthographic information?

We analyze the activations from different layers of the model and investigate if there is a difference in how well they recognize graphemes and phonemes. This analysis is useful for modeling the ASR model's interpretability, which is a fundamental task in improving end-to-end models. Based on the results from Belinkov and Glass [6] (see details in Section 3.1) and Lunde *et al.* [1] (summarized in Chapter 1), we expect that the higher layers of the model capture orthographic information better than the lower levels and that the higher layers are more concerned with graphemes than phonemes.

Research question 2: Is the information encoded in the model dialect-dependent?

For a further analysis of the information encoded in the model, we will test the model on different dialects and compare the resulting error rates. We will examine whether differences between the dialects are caused by dialectal variations, by examining certain words that can differ substantially between dialects. The analysis aims to gain knowledge about how ASR systems can be improved to handle dialectal speech.

2.2 Thesis Structure

This report is organized as follows: Chapter 3 introduces relevant theory and background information. It covers the basics of speech recognition, speech production, Norwegian dialects, and neural networks applied for speech recognition.

Chapter 4 presents the methods used to analyze the E2E ASR model in terms of grapheme classification and dialect analysis. The data sets and implementation details used to conduct the experiments are given in Chapter 5. Chapter 6 presents and discusses the results. A conclusion of the study and suggestions for future work are given in Chapter 7.

Chapter 3

Background & Theory

This chapter presents the background and theory on which this thesis is based. The level of difficulty is chosen so that other master's students with a background in electrical engineering, but not necessarily from machine learning or signal processing, can follow the deductions. Some parts of the text from this chapter were reported in Lunde *et al.* [1].

3.1 Related work

Traditional ASR systems are based on hidden Markov models (HMMs) combined with Gaussian mixture models (GMMs). These systems achieve satisfactory performance in terms of low word error rate (WER) but can be quite hard to implement and train. This is mainly because they contain an acoustic model (AM) which includes a pronunciation model (PM), since high expert knowledge is required to define the PM and the phonetic inventory. The introduction of neural networks (NNs) in ASR systems introduced more flexible systems. Initially, the neural networks were used as acoustic models, replacing the GMM in conventional systems. The so-called hybrid DNN-HMM models resulted in models with higher accuracy. Recently, E2E systems, which consist solely of NNs, have been explored to reduce the need for expert intervention in designing ASR systems. Several studies have obtained better performance with E2E models compared to conventional ASR systems. For instance, [3] achieved a CER of 30.51 % with a connectionist temporal classification (CTC) E2E system compared to 33.84 % with a bidirectional long short-term memory (BLSTM)-HMM model. The CTC is one of the two main structures of the E2E model, which was introduced in 2006 [3]. Compared to traditional neural networks, the CTC tries to map input acoustic features directly to graphemes without the need for frame-level alignments.

This will be further explained in Section 3.8.

One drawback with E2E models is that it can be hard to interpret them since they are trained all at once. Belinkov and Glass [5] investigated if and to what extent a deep end-to-end CTC model implicitly learned phonetic representations, with the intent of modeling the interpretability of end-to-end models. Their model was based on Deep Speech 2 [2] and consisted of two convolutional neural networks (CNNs) and seven recurrent neural networks (RNNs). They used data from the TIMIT data set [7], which contains English speech, and conducted the study by feeding activations from different layers of the model to a supervised classifier that was trained to recognize phonemes. The analysis showed that the first successive RNNs improved the representations, but after the fifth layer the accuracy decreased, which indicates that the top layers do not preserve all the phonetic information coming from the bottom layers. This is intuitive considering that the ASR system was trained to predict graphemes, and thus the phonetic representations must necessarily transform into adequate representations for the task at hand. This is a side effect of trying to recognize graphemes - there is nothing with the training method that forces the model to learn a phonetic representation.

Belinkov and Glass [5] extended their studies by evaluating several classification tasks in multiple languages (English and Arabic) and three different data sets [6]. In this study, they used the Deep Speech 2 light model, consisting of two CNN layers and five RNN layers. They evaluated the representation quality by comparing the results from framewise classification of phonemes and graphemes, as well as evaluating different articulatory features. Their results showed that the classification performance of graphemes followed the same layer-wise pattern as phonemes, though grapheme classification achieved slightly better results. The gaps between phoneme and grapheme classification were bigger at the top recurrent layers, and the relative performance drop at the top layer was smaller for graphemes than phonemes. This indicates that the top layers are more concerned with graphemes than with phonemes. They also found that over many different configurations, like languages, data sets, and linguistic properties, the E2E models exhibited strikingly similar behavior across layers. This suggests that such models may benefit from sharing information, for example using multilingual systems.

Dialectal variation is one of the biggest challenges regarding ASR. Prasad and Jyothi [8] conducted a detailed investigation of how accent information is reflected in the internal representations of an end-to-end ASR system. By analyzing several English accents, they found that most accent information was encoded within the first recurrent layer. This information is useful for adapting E2E models to learn accent-invariant representations.

Neural networks usually require a huge amount of training data to perform

well. Many speech analyzes require knowing the exact timing of phonemes. Labeling speech data manually is a time-consuming task, which makes automatic alignment of text to audio a fundamental task in speech research [9]. Though ASR systems based on neural networks have shown better performance than HMM systems [10], NNs have not had an improving influence on phone-to-audio alignment. This is partly caused by the increased use of end-to-end models like CTC, which disregards precise frame alignment. Zhu *et al.* [9] investigated this problem by designing frameworks for both text-dependent and text-independent phone-to-audio alignment using the wav2vec 2.0 model [10], which is an E2E neural network. Their results suggested that both proposed methods generate highly close results to the traditional forced alignment tools.

As introduced, the scarcity of training data for smaller languages is a fundamental challenge in ASR. The Norwegian Parliament Speech Corpus [11] was the first publicly available data set containing unscripted, Norwegian speech designed for training ASR systems. NPSC is an open data set containing 140h of recordings of meetings at Stortinget, the Norwegian parliament, with orthographic transcriptions in Nynorsk and Bokmål. The speech data contains partly read-aloud speech and partly spontaneous speech. Solberg and Ortiz [12] compared the performance of an ASR system based on Deep Speech 2 trained on the NPSC and Nordisk Språkteknologi (NST)[1] corpus, with a baseline system trained on clean, manuscript-read speech. They tested both systems on an independent data set containing spontaneous, dialectal speech, namely the NB Tale module 3 data set[2]. The NPSC-trained system performed significantly better than the baseline system, with a 22.9% relative improvement in terms of WER. This indicates that using spontaneous speech for training makes the system more robust for recognizing realistic speech data. Additionally, they evaluated the performance of the 12 dialect groups of the NB Tale module 3 data set. The dialect groups from the southeastern parts of Norway achieved the lowest WERs, with the group from Oslo performing best. The groups from the western and the middle parts of Norway achieved the highest error rates, with the group from Sogn og Fjordane performing worst.

3.2 Automatic Speech Recognition

ASR is based on finding the most likely sequence of words W given an input speech signal X. This is given by

[1] Datasets and documentation are available at `https://www.nb.no/sprakbanken/en/resource-catalogue/oai-nb-no-sbr-13/`.

[2] Datasets and documentation are available at `https://www.nb.no/sprakbanken/ressurskatalog/oai-nb-no-sbr-31/`.

$$\hat{W} = \underset{W}{\operatorname{argmax}} P(W|X)\,. \tag{3.1}$$

Usually, ASR systems are used to recognize spoken words and translate them into written text.

Speech can be transcribed into text in different forms. Often, we use either a phonetic or an orthographic representation. A phonetic transcription consists of phonemes, which are symbols representing how the speech signal is pronounced. A phoneme is the smallest unit of sound that distinguishes one word from another word in a language [13]. The phonetic transcriptions give close to a one-to-one relationship between sounds and symbols. The orthographic form consists of sequences of graphemes (letters) which form words using the standard spelling rules of the target language. In the Norwegian language, there are two official written languages, namely Nynorsk and Bokmål. There is no unequivocal correspondence between spoken dialect and orthographic representation. This can be illustrated with the word *hjørne* (orthographic form), which is annotated phonetically as *j2:n‘@*. The *h* is silent, and thus not included in the phonetic form. The *r* and *n* are pronounced together as the phoneme *n‘*. Additionally, the *e* is not pronounced as a normal e, but as a *schwa*[3] (the phoneme *æ*).

Several phonemes can represent the same grapheme, i.e., there are more phonemes than there are graphemes in the Norwegian language. There are 29 graphemes in the Norwegian alphabet, 9 of them are vowels and 20 are consonants. The exact number of phonemes in the Norwegian language differs for different dialects. According to Husby and Høyte [4], there are for example 53 phonemes in the dialect of Sandessjøen, 42 in the dialect of Bergen, and 39 in the Stavanger dialect. To see the difference between phonemes and graphemes, take as an example the phonemes *i* and *i:* in the Norwegian words *litt* and *lit*, respectively. These words are phonetically transcribed as *lit* and *li:t*. Both phonemes represent the grapheme *i*, but with different pronunciation lengths. Having a model that correctly identifies these phonemes will help the same system to correctly predict the written form of the word (with single or double *t*, in this case), which in turn contributes to producing more meaningful transcriptions.

3.2.1 Challenges related to ASR

As mentioned in Chapter 2, ASR is now accurate enough in terms of word error rate (WER) for the most frequently spoken languages in the world under the right conditions. But there are still many challenges that make speech recognition a complicated task. Some of the most important ones are listed below.

[3] You can read more about the schwa sound at `https://pronunciationstudio.com/schwa-pronunciation-guide/`

1. **Speaker variability:** Spectral characteristics are affected by age, gender, anatomy, and dialect, which result in variations between different speakers. Different samples from the same speaker can also vary due to emotion, stress, health condition, etc.

2. **Environment:** The speech recordings may be affected by e.g., background noise, channel noise, microphone distance, the type of microphone used, and room acoustics.

3. **Data:** ASR systems require a huge amount of training data to obtain sufficient performance. Additionally, the quality of the data must be good, in terms of realistic conditions like noisy environments and spontaneous speech. It is difficult to obtain enough training data for languages with fewer speakers, like Norwegian.

4. **Accent and dialect variations:** Covering accent and dialect variations is a difficult task that needs to be investigated further. There is improving work on English dialects, but there is little research done on smaller languages. Adapting the systems to non-native speakers is also an important work.

3.3 Norwegian dialects

As introduced in Chapter 2, in pre-modern times the Norwegian people were separated by natural barriers like fjords and mountains. This resulted naturally in dialectal variations across the country. Today, the differences between dialects are slowly diminishing. This is mainly because we get exposed to people with different dialects, due to relocation to the biggest cities and social media. Additionally, we are more exposed to other languages, like English, which introduces loanwords [14]. Nevertheless, there are still significant differences between the Norwegian dialects. This section introduces the main dialect clusters and their typical characteristics.

As mentioned previously, there are two written forms of the Norwegian language: Bokmål and Nynorsk. Bokmål is the most widely used form and is associated with an "urban style". It is the language of weekly magazines, the biggest newspapers, and the world of technology [4]. The spoken dialects in the urban eastern part of Norway, i.e., in the capital Oslo and the surrounding areas, are quite close to Bokmål. This way of speaking is called Urban East Norwegian and can be interpreted as a spoken form of Bokmål. But recall that there is no official way of speaking Norwegian. Nynorsk, on the other hand, is related to culture, tradition, and national values. There is a law that enforces the use of Nynorsk in media. Additionally, a minimum of 25 % of publicly available documents from

government agencies must be written in Nynorsk. Since it is created based on the systematization of many different dialects, we cannot say that there is a spoken form of Nynorsk. The spoken language in news readings and stage performances is the closest we get to a spoken form. The closest dialects are those from the rural southern parts of Norway, especially those from the west [4]. In most of Norway's 11 counties, more than 90 % of the pupils in first and secondary school have Bokmål as their first written language [15]. Only in the Western parts of Norway do a considerable amount of the pupils have Nynorsk as their first language; around 50 % in Vestland (former Hordaland and Sogn og Fjordane) and Møre og Romsdal, and around 20 % in Rogaland.

Separation of Norwegian dialects is a complicated task. There are no clear borders between different dialects, and the borders for the dialectal characteristics are not consistent. For example, the borders for first personal plural pronoun do not follow the borders for palatalization: e.g., the dialects in the north and the southeast use the same form of first personal plural pronoun (*vi*). Palatalization, on the contrary, occurs in many dialects in the north, but not in the southeast [16]. Additionally, there are often variations within a dialect, for example, due to socioeconomic differences. It is common to say that there are two dialects (or sociolects) in cities: one for the upper classes and one for "normal people". Due to all these varieties, we must look at the most distinctive dialectal characteristics to make the best possible grouping of dialects. This section presents the groups determined by Skjekkeland [16]. Some of the most important characteristics they used for the separation were "jamvektsregelen"[4], thick *L* (also called voiced retroflex flap[5]), and pitch accent. In terms of pitch accent, most Norwegian dialects are classified as either *low-pitch* or *high-pitch* dialects, which tells whether they rise the pitch on the stressed or the unstressed syllable in accent 1 words and accent 2 words[6]. Based on these characteristics, Norwegian dialects are roughly divided into two groups: *Vestnorsk* (West-Norwegian) and *Østnorsk* (East-Norwegian). Based on other characteristics, the two main groups can be divided into two smaller groups. Some important characteristics are the conjugation of infinitive verbs, personal pronouns, definite suffix, palatalization[7], and "soft consonants" (which means that *p*, *t*, and *k* are pronounced as *b*, *d*, and *g*, respectively, in certain words). We refer to Skjekkeland [16] and Christiansen [17] for a comprehensive review of these characteristics. Maps with the geographical distribution of the most important characteristics are displayed in Appendix A. Comparing the maps illustrates the difference in the geographical distribution of

[4]Read more about this at `https://snl.no/jamvektsloven`.

[5]The thick *L* is pronounced with the tongue being postalveolar and retroflex, i.e., it is flat or curled and touching the back of the alveolar ridge. More details can be found at `https://en.wikipedia.org/wiki/Voiced_retroflex_flap`.

[6]Read more about low-pitch and high-pitch dialects at `https://snl.no/tonelag`.

[7]Read more about palatalization at `https://snl.no/palatalisering`.

the different characteristics.

It is common to divide Vestnorsk into *Vestlandsk* (western and southern part of Norway) and *Nordnorsk* (Northern Norway), while Østnorsk is often divided into *Trøndersk* (middle area of Norway) and *Østlandsk* (eastern part of Norway). These four groups can again be divided into finer groups. The geographical distribution of these four groups is displayed in Figure 3.1. A summary of some of the most important characteristics of the four groups is given below.

Vestlandsk

1. Thick *l*: No, except for the Romsdal area.
2. Definite suffix: Typically *-o* or *-å* ending.
3. First personal pronouns: Singular: usually *e* or *eg*, but *i*, *æg*, and *ej* occur in some places. Plural: Usually *me* or *mi*.
4. Pitch accent: High-pitch dialects.
5. Additional characteristics: *Guttural R* is present in the southwestern parts of Norway. The southern parts also have *soft consonants*.

Nordnorsk

1. Thick *l*: Present in the southern part of Northern Norway.
2. Definite suffix: Usually *-a* ending, but *-o, -å*, and *-æ* also occur.
3. First personal pronouns: Singular: mostly *æ* or *e*, but *eg* and *æg* also occur. Plural: *Vi/ve*.
4. Pitch accent: High-pitch dialects.
5. Additional characteristics: palatalization and apocope in the southern parts of Northern Norway.

Østlandsk

1. Thick *l*: Yes.
2. Definite suffix: *-a* ending is most typical, but *-e* ending also occurs.
3. First personal pronouns: Singular: *je, jæi, e*. Plural: *vi, ve, oss*.
4. Pitch accent: Low-pitch dialects.

Trøndersk

1. Thick *l*: Yes.

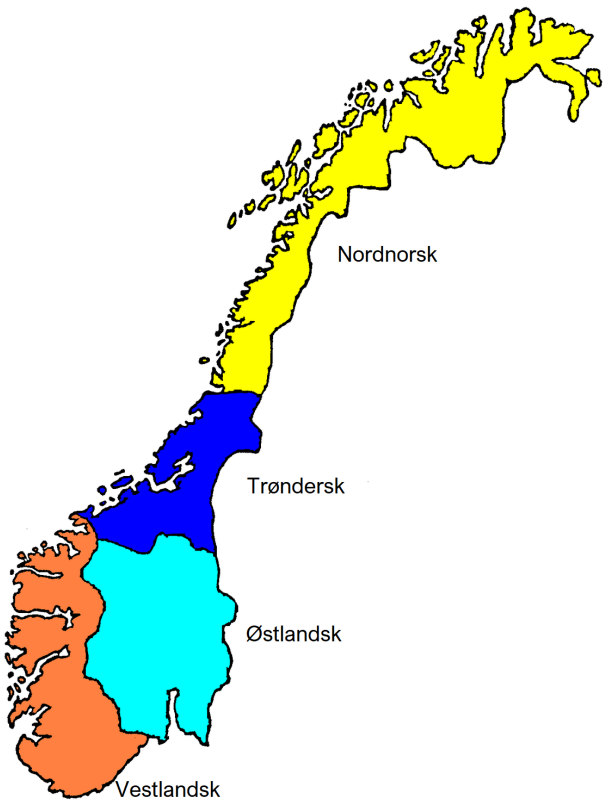


Figure 3.1: The four main groups of Norwegian dialects. Picture taken from `https://no.wikipedia.org/wiki/Tr%C3%B8ndersk`.

2. Definite suffix: *-a* is most typical.

3. First personal pronouns: Singular: *æ*. Plural: *vi, åss, mi, me*.

4. Pitch accent: Low-pitch dialects.

5. Additional characteristics: apocope, palatalization.

As we can see, there are both acoustic and lexical variations between the clusters. Many irregular lexical variations are not mentioned here - many words can look completely different in different dialects. One example is the Bokmål form *bedre*, which typically is spoken as "likar" in the Trøndersk dialects. Such variations are challenging in terms of ASR since the ASR model must learn to transcribe the dialectal forms as their Bokmål form, even though they can be phonetically very different.

The variations across one main dialect cluster can be big, and there is not a distinct separation between the clusters. Though the most important characteristics usually are shared between the dialects in the Nordnorsk & Vestlandsk groups, and the Østlandsk & Trøndersk groups, this distribution is a bit random. For example, the dialects from the southern parts of North Norway and Trøndersk have a lot in common, and there are many lexical differences between Trøndersk and Østlandsk. Though the northernmost dialects differ from the southeastern dialects in terms of pitch accent and phonetics, their lexical inventories are quite similar. Thus, the separation of Norwegian dialects is a complicated task in terms of speech recognition.

Skjekkeland [16] divided the Norwegian dialects into finer groups based on the theory above. The geographical distribution of these groups is illustrated in Figure 3.2.

3.4 Speech production

ASR is a complex task. There are many sources of variation in speech, which can make the same sentence sound different when pronounced either several times by the same speaker, or when pronounced by different speakers. We will in this section introduce the main concepts of speech production and sources of variation.

The main components of the speech production apparatus are the lungs, trachea, larynx (which contains the vocal cords), pharyngeal cavity (throat), oral cavity, and nasal cavity. The pharyngeal and oral cavities are typically referred to as the vocal tract and the nasal cavity as the nasal tract. The speech production components are illustrated in Figure 3.3. Normally, speech is created by air pressure from the lungs which is led through the glottis with the vocal cords and then modified in the vocal tract. When the air-pressure waves are emitted

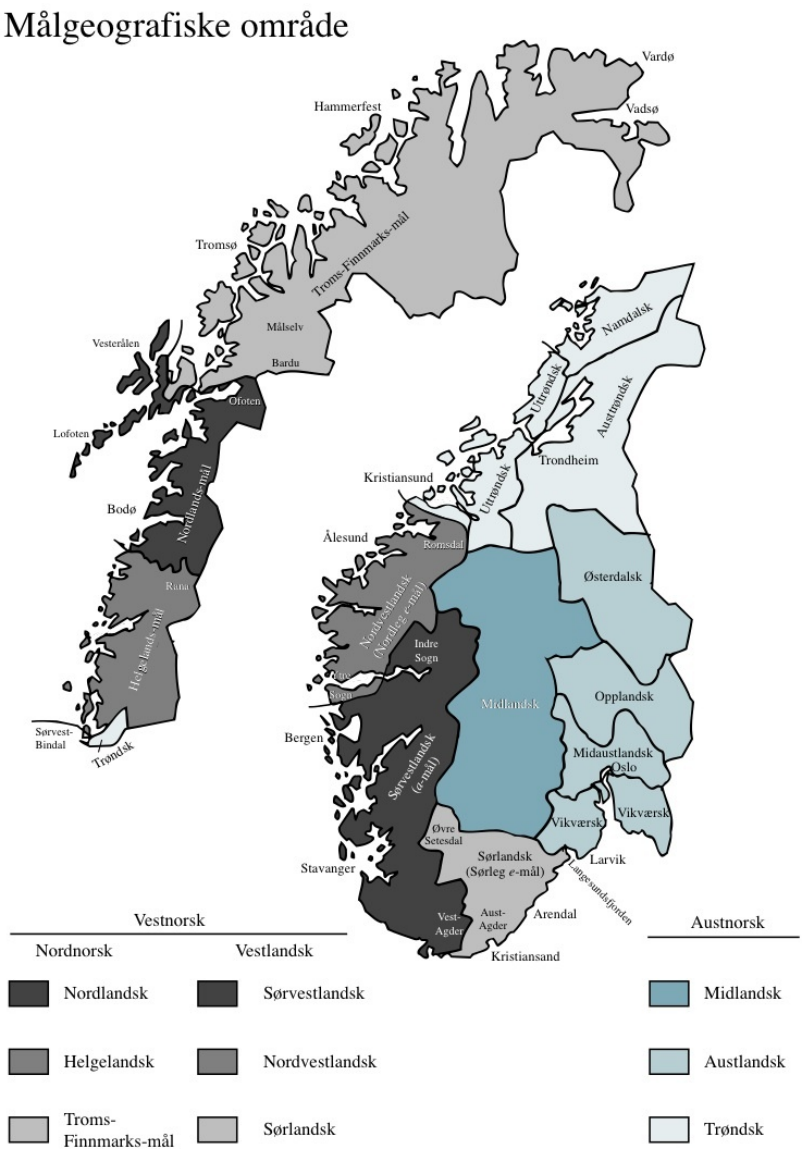


Figure 3.2: Finer clustering of the Norwegian dialects [14].

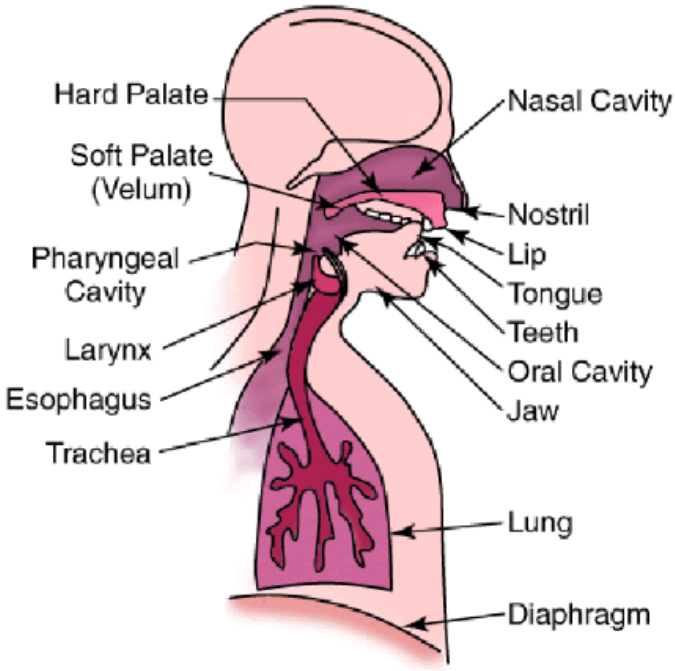


Figure 3.3: The human speech production system [18].

from the mouth and the nostrils of a speaker, this can be perceived as sound by a listener.

Often, we classify a sound as being either voiced or unvoiced. Voiced sound is created when the vocal folds vibrate during phoneme articulation, otherwise, the sound is unvoiced [19]. The spectrograms of voiced sounds (read more about spectrograms in Section 3.5) consist of a quite regular pattern, while the spectrograms of unvoiced sounds are more irregular. The vibration rate of the vocal folds is called the fundamental frequency, $F0$. $F0$ contributes highly to the perception of pitch, the relative highness or lowness of voice tones. Pitch can contribute to the meaning of a sentence, by indicating the mood of the speaker or whether the speaker is asking a question or is ironic. The fundamental frequency differs between individuals and depends on physiological factors. It is usually inversely proportional to body size, i.e., the smaller you are, the higher $F0$ you have. $F0$ can be as low as 60 Hz for large men and as high as 300 Hz for children and small women [19].

The resonances of the vocal tract change when the vocal tract changes (i.e.,

Vowel Labels	Mean F1 (Hz)	Mean F2 (Hz)
*iy (**fee**l)*	300	2300
*ih (**fi**ll)*	360	2100
*ae (g**a**s)*	750	1750
*aa (**fa**ther)*	680	1100
*ah (c**u**t)*	720	1240
*ao (d**o**g)*	600	900
*ax (**co**mply)*	720	1240
*eh (p**e**t)*	570	1970
*er (t**ur**n)*	580	1380
*ow (**to**ne)*	600	900
*uh (g**oo**d)*	380	950
*uw (t**oo**l)*	300	940

Figure 3.4: Typical formant values for American English vowels [19].

due to tongue placement and the shape of the vocal tract). Harmonics (integer multiples of $F0$) near the resonances of the vocal tract are emphasized. These harmonics are called formants and give rise to the perception of vowels. The first and second formants are called $F1$ and $F2$ and determine the characteristic quality of the vowel.

As an example, the mean values of $F1$ and $F2$ of American English vowels are listed in Figure 3.4.

3.4.1 Variability in speech

Coarticulation

The value of the formant frequencies is highly affected by different factors, like speech rate and neighboring phonemes. Thus, the formant values may vary a lot from their "standard" mean values, which complicates the speech recognition task. The phenomenon where neighboring phonemes influence each other is called coarticulation. A phoneme that has different realizations based on its phonetic neighbors is called an allophone. An example of an allophone is the phoneme *p*. It is pronounced differently in the words *pin* and *spin*. For the first word, most speakers produce a small puff of air, which is considerably reduced when an *s* is placed in front of the *p*, for example in the word *spin* [19].

Continuous vs. read-aloud speech

Whether the input speech signal consists of spontaneous, or read-aloud speech, clearly affects the recognition rate of an ASR system. Several factors influence the recognition rate negatively in the case of spontaneous speech. The speaker may for example utter vocal or nasal hesitation (*eeh* or *mmm*). It can be difficult for the model to recognize these sounds correctly and interpret the intent behind the sounds, e.g., whether the speaker is thinking about what to say or if they actually are part of a meaningful utterance. For spontaneous speech, it is also more common to "swallow words", merge words, and cut the ending of words. Thus, it is less probable that formant targets are fully achieved. Naturally, these factors make speech recognition more difficult. If the speaker reads from a manuscript, the speech is more likely to be clear and understandable. There is usually less hesitation, and the spoken language is often more similar to written language, so the transcribing task is more straightforward. Dialect-specific words, which may not be known to the language model (see Section 3.6), are usually omitted in read-aloud speech. Additionally, the speaking rate (e.g., the number of words uttered per minute), affects the recognition accuracy. Usually, a higher speaking rate leads to a higher error rate, due to many of the same factors that yield spontaneous speech.

Context dependency

Capturing the context of a speech signal is still a big challenge in automatic speech recognition. With today's technology, machines are outperformed by humans. Human listeners manage to capture the main context of an utterance even if some words are unclear, and we manage to distinguish between words that sound equal based on the context. This makes us good speech recognizers. One challenge is that people often use tone to indicate context: whether they ask a question, whether they are ironic, etc. Most Norwegian dialects use tone to distinguish between words that contain the same speech sound [4]. For example, the word *tømmer* can have two different meanings, based on the toneme. It can mean either *timber* or *emptying*, based on whether the tone is rising or falling on the first syllable. In these cases, it is not enough to transcribe the spoken words correctly to understand the whole context, which can affect the recognition of the following words.

The sentence "Do you wanna go to Lisa at two o'clock too?" is an example of a challenging utterance in terms of speech recognition. *To*, *two* and *too* are pronounced almost identically. Without understanding the context of this sentence, it would be very difficult to transcribe it correctly.

In addition to coarticulation, speech rate, and context variability, there are many other sources of variability that complicate the speech recognition task, as

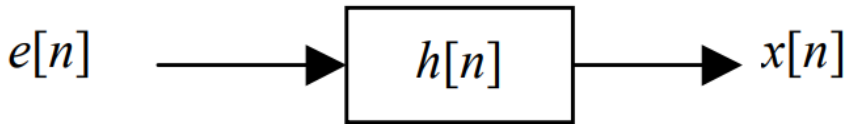


Figure 3.5: Source-filter model [19]. The excitation $e[n]$ (the source) is passed through a linear filter $h[n]$, which results in the filtered speech signal $x[n]$.

introduced in Section 3.2.1.

3.5 Analyzing speech signals

It is useful to make a simplified model of the human speech production system for the analysis of speech signals. The most common representation is the source-filter model, where the source represents the air-pressure waves emitted from the lungs and the filter represents the resonances of the vocal tract [19]. The resulting filtered signal represents the speech signal that can be perceived by a listener. The process is illustrated in Figure 3.5. The excitation $e[n]$ (the source) is passed through a linear filter $h[n]$, which results in the filtered speech signal $x[n]$.

We are interested in separating the source and the filter. Estimating the source and/or the filter is useful in many applications, e.g., for phoneme classification (and thus speech recognition) and speech synthesis. Traditionally, speech recognizers estimate the filter characteristics and ignore the source [19]. By estimating the filter, the source can be estimated by sending the speech signal through the inverse estimated filter.

As a tool for separating source and filter, we often represent the speech signal $x[n]$ by its spectrogram, so that we can apply analyses in the frequency domain. This will be introduced in the following section.

3.5.1 Short-time Fourier transform and spectrograms

When applying the Fourier transform to a signal, we go from analyzing the signal in the time domain to analyzing it in the frequency domain. In the frequency domain, we can analyze the frequency content of the signal. This is for example useful in acoustic modeling, for recognizing which phonemes have been pronounced. One criterion for applying the Fourier transform to a signal is that the signal must be stationary, which means that its statistical properties do not change over time. Speech signals change a lot over time and are thus not stationary. But if we analyze a speech signal over a very short time frame, the signal can be regarded as stationary, and thus the Fourier transform can be applied.

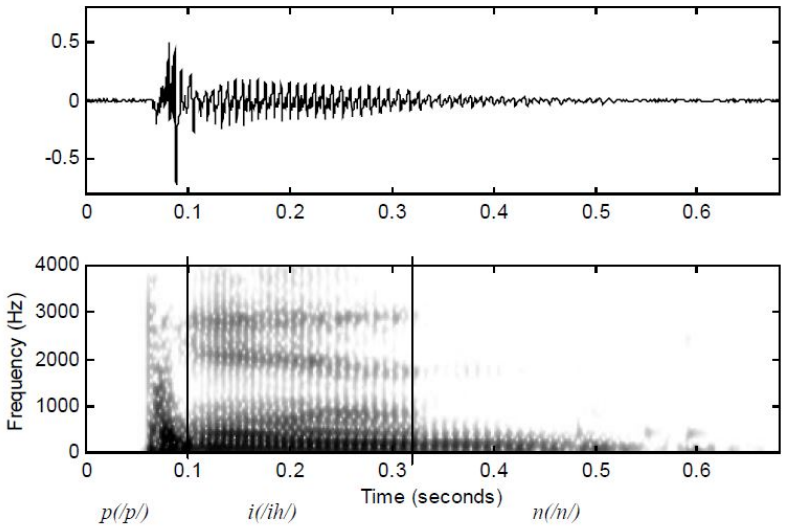


Figure 3.6: The waveform (upper figure) and spectrogram (lower figure) of a realization of the word *pin* [19]. The approximate phoneme boundaries are indicated by the vertical bands.

This is the idea behind spectrograms, which is a tool for visualizing speech signals, through a two-dimensional plot showing time and frequency on the figure axes. The spectrogram of a speech signal is obtained by computing the Fourier transform of short, overlapping frames of the speech signal. This method is called short-time Fourier transform (STFT). The de facto standard in speech recognition is to use time frames of length 25 ms and an overlap (also called stride) of 10 ms between successive frames. For comparison, the duration of phonemes is in the order of milliseconds, where the exact length varies between phonemes and due to speaker variations (e.g., speech rate). In speech recognition, the most frequently used window frames are the rectangular, Hann and Hamming window functions.

Spectrograms are very useful for analyzing phonemes. Figure 3.6 shows the waveform of the word *pin* and its corresponding spectrogram, which captures the transitions between each successive phoneme in the utterance.

The relative energy present at a given frequency is displayed in the spectrogram. The darker the color, the more energy is contained. This is intimately related to the phonemes spoken and the signal received by our ears. The two dark horizontal bands in Figure 3.6 indicate the first and second formants of the vowel *ih*. We can see that there is no energy present at the beginning of the spectrogram. This is due to the pronunciation of the phoneme *p*, which involves

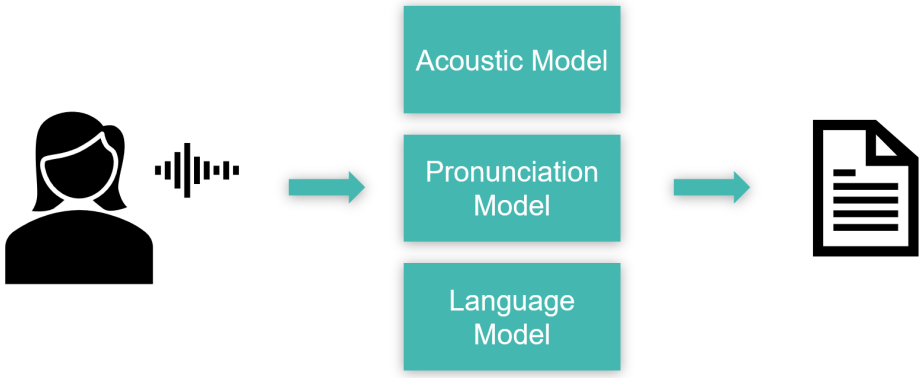


Figure 3.7: Illustration of the functionality of a conventional ASR system that consists of three modules that are trained separately. Speech signals are given as input to the model, which tries to recognize it and transcribe it correctly into text format.

a nearly complete blockage of airflow from the oral cavity.

3.6 Conventional ASR system

As mentioned previously, a conventional ASR system consists of an acoustic model which also includes a pronunciation model and a language model which are trained independently. Figure 3.7 illustrates the functionality of a conventional ASR system. The following sections explain the different parts in more detail.

By using Bayes' theorem, (3.1) can be written as

$$\hat{W} = \underset{W}{\operatorname{argmax}} \frac{P(X|W)\,P(W)}{P(X)}. \tag{3.2}$$

Since $P(X)$ is fixed for all possible word sequences W, Eq. (3.2) can be simplified to

$$\hat{W} = \underset{W}{\operatorname{argmax}}\, P(X|W)P(W)\,. \tag{3.3}$$

$P(X|W)$ is modeled using an AM and $P(W)$ is modeled using an LM. It is challenging to design precise AMs and PMs that reflect the input speech signal satisfactorily so that it can be recognized correctly.

3.6.1 Acoustic model

Combined with a PM, the acoustic model (AM) is used to compute $P(X|W)$, the emission probability, which is the probability of observing an acoustic feature X given a word or a word sequence W. $P(X|W)$ should consider speaker variations, pronunciation variations, environmental variations, and context-dependent phonetic coarticulation variations [19]. In conventional ASR systems, $P(X|W)$ is most often modeled using a combination of an HMM and a GMM. The GMM is used to estimate the emission probabilities, by assuming that the observations are drawn from a multivariate Gaussian distribution. The GMM is used to model the dynamics of the speech signal, i.e., finding the most probable word sequence and placing the predicted phonemes in the correct order. As introduced in Section 3.5.1, speech signals are non-stationary signals, which means that their distribution changes with time. Thus, assuming that the observations are drawn from a multivariate Gaussian distribution does not hold for the whole speech signal, only for some parts of it. Replacing the GMM with an artificial neural network (ANN) for acoustic modeling has led to improved results [3], since ANNs are more flexible and thus better at discriminative learning (i.e., learning how to distinguish different classes without knowing their probability distributions).

Pronunciation model

In conventional ASR systems, the acoustic model also includes a pronunciation model (PM). The PM maps a sequence of phonemes produced by the acoustic model to words. It is based on a pronunciation lexicon, which contains keywords - the orthographic form of a word - and their corresponding pronunciation information. The pronunciation information contains all possible phonetic realizations of the keyword, which can range from one to several dozen. Traditionally, pronunciation lexicons are generated manually by a skilled phonetician. This method is both expensive and very time-consuming. Additionally, it is difficult to cover all possible words, especially in the Norwegian language which has many compound words.

3.6.2 Language model

The aim of the language model (LM) is to model the word sequence of the speech input, i.e., choose the correct word among words that sound similar and put the chosen words in the correct order to form a meaningful sentence. Traditional LMs are dependent on the training corpus, which means that they are unable to infer new words that are not present in the corpus. This can be problematic when dealing with Norwegian speech since the Norwegian language is highly affected by compound words, which leads to a huge number of realizable words. This

gives a high probability of out-of-vocabulary words, namely that some words are not present during training.

The language model should reflect how frequently a word string W occurs in a language. For the English language, $P(hi)$ might have a relatively high probability, e.g., 0.01, since people are saying the word *hi* quite often [19]. Conversely, $P(chichi\,pli\,pu) \approx 0$, because the probability that someone will say this is extremely small. The probability of observing W, $P(W)$, can be represented by

$$\begin{aligned} P(W) &= P(w_1, w_2, ..., w_n) \\ &= P(w_1)P(w_2|w_1)...P(w_n|w_1, w_2, ..., w_{n-1}) \\ &= \prod_{i=1}^{n} P(w_i|w_1, w_2, ..., w_{i-1}). \end{aligned} \tag{3.4}$$

For most values of n, $P(W)$ will be approximately impossible to estimate since there are so many possible combinations of word strings that must be considered. Even with a massive training set and modest n, most word sequences will not occur in the training set. n-gram models handle this problem, as explained below.

n-gram model

The n-gram model assumes that word number i in a sequence, w_i, only depends on the $n-1$ previous words in a sequence for context inference. The most common n-grams are the unigram, bigram, and trigram. The trigram assumes that w_i depends on the two previous words, i.e., $P(w_i) = P(w_i|w_{i-2}\,, w_{i-1})$. Similarly, the bigram gives $P(w_i) = P(w_i|w_{i-1})$ and the unigram gives $P(w_i) = P(w_i)$. The probability of a certain word sequence is estimated by counting how often it occurs in a corpus.

Even though we have a big amount of training data, likely word sequences may occur very seldom, or even never. For example, for a corpus containing several-million words of English text, more than 50% of trigrams occur only once, and more than 80% of trigrams occur less than five times [20]. This problem is handled by n-gram smoothing, which estimates more robust probabilities of unseen sequences. This is done by adjusting the probability of infrequent sequences upwards and frequent sequences downwards. We refer to Chen and Goodman [21] for a detailed review of smoothing techniques.

3.7 Machine Learning basics

We will in this section introduce the most important machine learning basics, which will be essential for the reader for understanding the concept and discussion

of our experiments.

3.7.1 Machine learning algorithm

A machine learning (ML) algorithm is an algorithm that can learn from data [20]. This means that if you give an ML algorithm a lot of data, it will improve its *performance* in solving a certain *task* by *experience* from training. Often, the terms machine learning algorithm and machine learning model are used interchangeably. In this report, when we use the term *ASR model*, we are referring to the algorithm driving the ASR system. We will now explain each part of the training task in more detail.

Many kinds of tasks can be solved with machine learning. Some of the most common machine learning tasks are classification, regression, transcription, and machine translation.

The performance measure is a quantitative measure of the machine learning algorithm's performance. For classification tasks, we normally use accuracy or error rate to measure the algorithm's performance. In speech recognition, word error rate (WER) and character error rate (CER) are common performance measures, as will be explained in Section 3.7.2.

The experience says something about how much information the algorithm is given during the learning process. This can be ordered into two classes: supervised or unsupervised learning. In the case of supervised learning, the algorithm gets a set of input observations x and corresponding output values y. That is, the algorithm knows the true representation of each input observation and tries to estimate the conditional distribution $p(y|x)$. An example of a supervised machine learning algorithm is the support vector machine (SVM). For unsupervised learning, on the contrary, the algorithm does not know the correct representation of each input vector, hence the name. The algorithm tries to learn the probability distribution of the given input observations. Examples of unsupervised machine learning algorithms are principal component analysis (PCA) and k-means clustering.

Usually, we use a train set for training the machine learning algorithm and a test set to evaluate its performance. The train and test set should fulfill the i.i.d. assumption: the samples in both sets should be independent of each other, and the train and test set should be identically distributed, e.g., drawn from the same probability distribution. This is because we are interested in how well the algorithm performs on unseen data, which would be the case when applied in a real-world situation. The ability to perform well on unseen data is called *generalization*, which is a central challenge in machine learning.

3.7.2 Performance measure in speech recognition

Error rate is the most popular performance measure of a speech recognition system, either in terms of word error rate (WER) or character error rate (CER). The error rates can be computed by comparing the true transcription with the model's predicted output. There are typically three types of recognition errors in speech recognition which are used to compute the error rates [19]:

1. Substitution: An incorrect word was substituted for the correct word.
2. Deletion: A correct word was omitted in the recognized sentence.
3. Insertion: An extra word was added to the recognized sentence.

Word Error Rate (WER)

When calculating the word error rate, we look at the amount of wrongly recognized words in the predicted output. The WER is given by

$$\text{WER} = 100\% \cdot \frac{Subs + Dels + Ins}{\text{No. of words in the correct sentence}}. \tag{3.5}$$

Eq. (3.5) tells us that we cannot compare the word sequences word-by-word (e.g., first comparing the first word in each sentence, then the second word, and so on). For example, the sentence "the girl is kind" may be recognized as "girl is very kind". If we compare the sentences word-by-word, we get an error rate of 75 % (the vs. girl, girl vs. is, is vs. very, kind vs. kind). Using Eq. (3.5), we have one deletion (the) and one insertion (very) of in total four words of the true transcription, which gives an error rate of $100\% * 2/4 = 50\%$.

Character error rate (CER)

The character error rate is calculated similarly to the WER, except that we are comparing the characters (i.e., graphemes) instead of words of the true and predicted transcription. This is given by

$$\text{CER} = 100\% \cdot \frac{Subs + Dels + Ins}{\text{No. of characters in the correct sentence}}. \tag{3.6}$$

3.7.3 Overfitting and underfitting

During the learning process, the machine learning algorithm first samples the train set and updates its parameters accordingly to reduce the training error, i.e., the proportion of wrong predictions when the algorithm is run on the train

set. Next, the algorithm samples the test set. The generalization error, or test error, is defined as the expected error on new, unseen input, namely the test set.

We expect the generalization error to be greater than or equal to the training error. Both the training error and the generalization error should be as low as possible. If the machine learning model is not able to obtain a sufficiently low error rate on the train set, we call this *underfitting*. *Overfitting* is when the gap between the train and test error is too big, which means that the model is not good at generalizing, i.e., it does not adapt properly to new, unseen data. Overfitting and underfitting tell us how well the model performs and are central challenges in machine learning.

By adjusting the *capacity* of the model, we can control the balance between overfitting and underfitting. If the model has a high capacity, it can fit a wide variety of functions, which means that it overfits the train set. With a low capacity, the model struggles to fit the training set, which leads to underfitting. Several techniques can be applied to tackle the problems of underfitting and overfitting. Regularization is a method that can be used to reduce overfitting. Some examples of regularization are dropout, early stopping, and data augmentation.

When training a model, it is common to split the training data into two separate subsets, which are called the training set and the validation set. Often, the training set contains 80 % of the training data, while the validation set contains 20 %. The training set is used to learn the model parameters during training. The validation set estimates the generalization error during or after training and uses this estimate to update the hyperparameters (we refer to Goodfellow *et al.* [20] for a detailed review of hyperparameters). After the hyperparameter optimization is complete, the generalization error can be estimated using the test set. Since the validation set is used to "train" the hyperparameters, we expect the validation set error to be smaller than the generalization error, but it will give a better estimate of the generalization error than the training error would give.

3.7.4 Gradient descent and batches

Often, the parameters of the machine learning algorithm cannot be computed analytically. In such cases, we define a loss function $f(x)$[8] and use an optimization algorithm for finding the model parameters that minimize the loss function. One of the most widely used optimization algorithms is gradient descent. A practical explanation of how gradient descent works is that we compute the gradient of $f(x)$, $f'(x)$, and then move with small steps along with $f(x)$, in the decreasing direction of the gradient, to find the global minimum of the loss function $f(x)$. If f(x) has several minima, it is important to find the global minimum, or at least a local minimum not far from it. This is illustrated in Figure 3.8. The

[8]Note that the loss function often is annotated by the symbol $\mathcal{L}$.

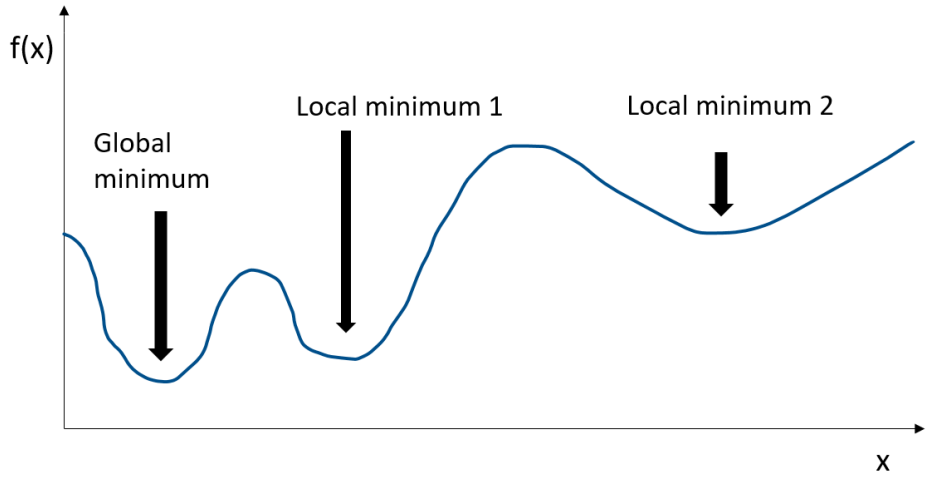
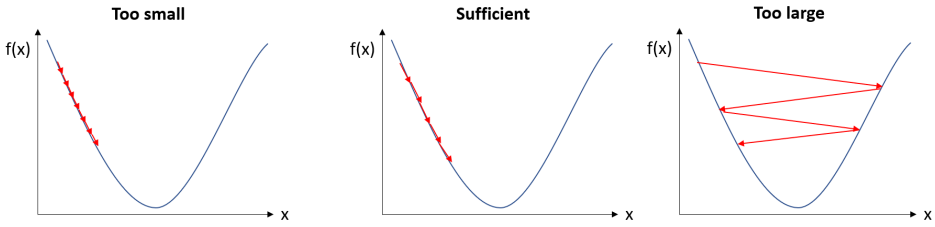


Figure 3.8: Illustration of the importance of finding the global minima of the loss function $f(x)$. The value of the first local minimum is close to the global minimum and would thus give quite good results. The second local minimum is much larger than the global minimum and would thus lead to a poor loss calculation.

learning rate determines the step size when moving along $f(x)$. There is a trade-off between using a too small step size, such that it takes a very long time to reach the minima, or using a too big step size, where we risk "jumping" over the minima. This is illustrated in Figure 3.9. A *learning rate scheduler* can be used to optimize the learning rate, which means that the learning rate is not fixed during training but is gradually decreased with increasing training epochs[9]. Using a scheduler when training neural networks often gives slightly better results, since using smaller steps at the end of the training allows for better fine-tuning of the model parameters.

As introduced previously, machine learning algorithms need a lot of data to obtain sufficient training results. Applying gradient descent to the whole training set is thus computationally expensive and time-consuming. This problem is solved by dividing the training set into smaller subsets, called mini-batches. We apply the same procedure for computing the gradient to find the minima, but we only use one mini-batch at a time for updating the parameters. For each step, a number of random samples equal to the mini-batch size are chosen. Typical mini-batch sizes are of the n-th power of two, typically 32 (2^5), 64 (2^6), or 128

[9]One epoch means that the ML algorithm has one complete pass through the training set.



(a) A too small step size leads to slow convergence towards the minimum.

(b) Sufficient step size for finding the minimum.

(c) A too large step size may lead to that the algorithm fails in finding the minimum.

Figure 3.9: Comparison of the architectures of the original ASR model and the supervised classifier.

(2^7). Mini-batch gradient descent has shown to be both an effective and accurate way of finding the global minima of the loss function [20].

3.8 Deep Learning

Machine learning algorithms can be used to solve many kinds of problems, but they fail in solving complex tasks like speech recognition and object recognition. This challenge was one of the motivations for the development of deep learning.

Deep learning is a subset of machine learning. The use of deep learning for ML tasks has increased significantly in the last decade. This means using Artificial Neural Networks (ANNs) for solving a given task. ANNs are models that are inspired by the human nervous system, hence the name *neural network*. ANNs consist of one input layer, one or more hidden layers, and an output layer. The hidden layer(s) contains numerous nodes with behavior inspired by the neurons in the brain. The nodes are connected with weighted links, which are updated and optimized by training the neural network. Each node can either be connected to all other nodes in the next layer, or they can be connected to a subset of the other nodes. A classical configuration is feedforward neural networks, which means that the information only propagates forward from one layer to the next, i.e., the connections between the nodes do not form a cycle.

Deep learning methods were first explored several decades ago but have become more useful in recent times as the amount of training data has increased since neural networks require a huge amount of data for sufficient training. The improvement of computer infrastructure, which means faster computers with larger memory, has allowed for bigger deep learning models, i.e., the number

of nodes can be increased. Since the introduction of hidden units, artificial neural networks have doubled in size roughly every 2.4 years [20]. These factors have resulted in networks that achieve higher accuracy on more complex tasks. Even though we consider today's networks large, they are far from the size of the human brain. Today's networks are smaller than the nervous system of even relatively primitive animals like frogs [20]. With the current pace of increase of neural networks, they will have the same number of nodes as the human brain earliest in the 2050s.

In the following sections, we will introduce some types of ANNs that are widely used in speech recognition and relevant to this thesis.

3.8.1 Recurrent Neural Network

An RNN is a type of ANN which is suited for modeling sequential data or time-series data.

A big difference between RNNs and feedforward networks is that while feedforward networks assume independence between inputs and outputs, the output of recurrent neural networks depends on the prior elements within the sequence. This makes RNNs powerful sequence learners. Another discriminating characteristic of RNNs is that they share parameters across different parts of the network. This stands in contrast to feedforward networks, which have different weights across each node. Parameter sharing makes it possible to extend and apply the model to examples of different lengths and generalize across them [20]. An example of an RNN architecture is given in Figure 3.10.

Vanishing gradient is a common problem when training RNNs using back-propagation. This makes the RNN have short-term memory, i.e., RNNs have problems with capturing long dependencies [23]. This can be a problem when there are many words between two context-informing words in a sentence. For example, in the sentence "Sara's mother is from *Italy*, so even though Sara is born and raised in France, she speaks *Italian* fluently", a model with short-term memory would have problems with capturing that Sara can speak Italian. Several kinds of RNN architectures are developed with the intent of better capturing long-term dependencies. We will introduce the most important ones here.

Bidirectional RNN

A bidirectional RNN (BRNN) consists of two RNNs: one moves forward through time, starting at the beginning of the sequence, while the other moves backward through time, starting at the end of the sequence. This makes BRNNs able to take both past and future data into consideration to predict the current state, while unidirectional RNNs can only take previous inputs into consideration when predicting the current state. When predicting the output at a current time, the

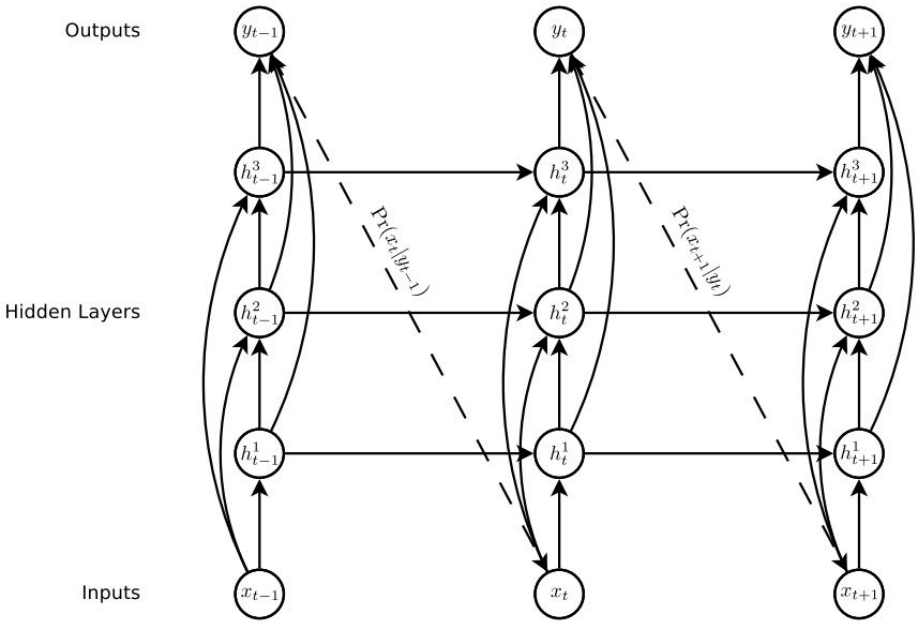


Figure 3.10: Deep recurrent neural network architecture. The circles represent network layers, the solid lines represent weighted connections, and the dashed lines represent predictions [22].

BRNN is most sensitive to the input values around that time but is also taking the past and future into consideration. This is very advantageous when working with sequential data where the predicted output may depend on the whole input sequence. As introduced in Section 3.4.1, speech is affected by coarticulation and context, both on a phonetic level and between words. Neighboring phonemes and words may affect how the current phoneme or word is pronounced. Additionally, we may need to listen to a whole utterance to fully interpret its meaning. Take the sentence "I am very happy since I won the lottery" as an example. A BRNN would use the section "won the lottery" to predict the mood of the speaker. This ability makes BRNNs better at capturing content and predicting the correct state. BRNNs have achieved incredible results in sequence recognition and learning tasks such as handwriting recognition [24]; [25], speech recognition [26], and bioinformatics [27].

A typical BRRN is illustrated in Figure 3.11.

3.8.2 Gated RNNs

Gated RNNs include long short-term memory (LSTM) and networks based on the gated recurrent unit (GRU). These components are solving the vanishing gradient problem, by creating paths through time that have derivatives that neither vanish nor explode [20]. Thus, they are addressing the short-term memory problem of RNN models. Information can be stored in these networks for a long time. When the information is used, the network forgets the old state by setting it to zero. Gated RNNs learn to decide when to forget the old state. The LSTM is out of scope for this thesis, so we will only give a brief introduction to the GRU in the following section.

Gated Recurrent Unit

Introduced in Cho *et al.* [28], the GRU is addressing the short-term memory problem of RNN models. It uses hidden states to store information. The information flow is controlled by two gates - a reset gate and an update gate. The same gating unit simultaneously controls the forgetting factor and the decision to update the state unit [20].

3.8.3 Convolutional Neural Network

CNN is a class of ANNs mainly used as a feature extractor, especially for image processing. In the context of speech recognition, spectrograms can be interpreted as images fed to the CNN. CNNs consist of one or more pairs of a convolutional layer and a pooling layer, followed by a fully connected layer that performs classification. An illustration of a CNN architecture is displayed in figure 3.12.

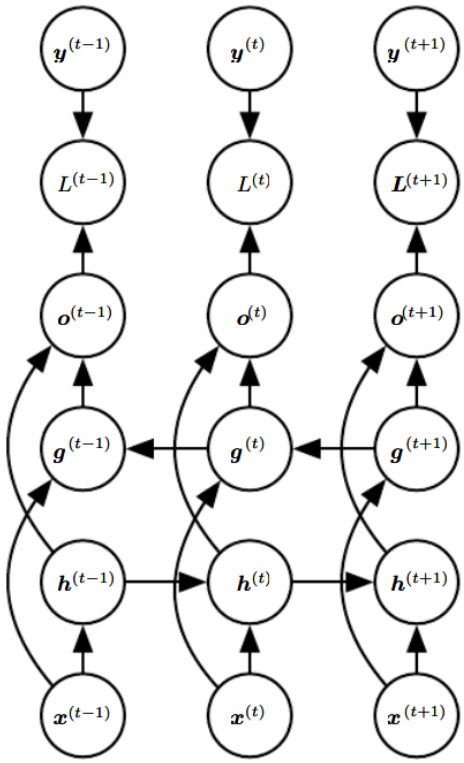


Figure 3.11: Typical configuration for a BRNN network. h represents the RNN which propagates forward through time, while g represents the RNN which propagates backward through time. The output units o learn to map input sequences x into target sequences y for each time step t, with loss L [20].

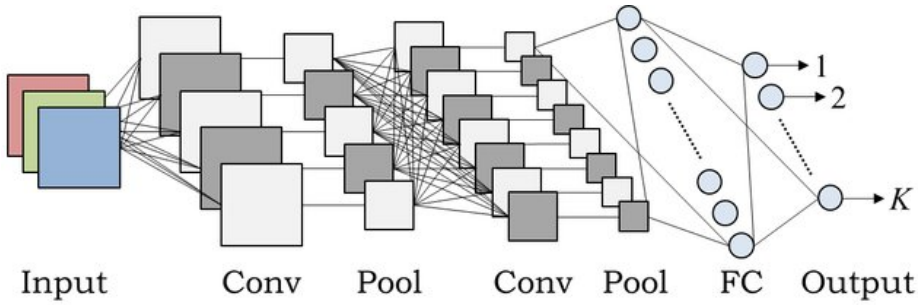


Figure 3.12: A possible CNN architecture [29]. The convolutional and pooling layers are followed by a fully connected layer (FC) that outputs probabilities for each of the K classes.

The convolutional layers are based on the convolution operation and are used to extract high-level features from the input data. The pooling layer takes several inputs and outputs a statistical value, e.g., mean or max value. This leads to dimensionality reduction, which reduces the computational cost and controls overfitting [30]. The fully connected layer applies a nonlinear function to each element, like sigmoid, tanh, or ReLU.

3.8.4 End-to-End ASR Model

The end-to-end ASR model was introduced in the attempt of mapping speech input directly to an output sequence of graphemes or words. It replaces most modules from the conventional ASR system with a single ANN. The network can automatically learn language or pronunciation information, but it needs a large amount of training data to achieve sufficient results. One drawback with E2E models is that it can be hard to model their interpretability because it is trained all at once. This contrasts with conventional ASR systems, where each part of the model is trained independently. The main structures for E2E modeling are Connectionist Temporal Classification (CTC) and sequence-to-sequence (seq2seq) modeling. We do not make use of seq2seq modeling in our work, so it is out of scope for this thesis. We will explain the main principles of CTC in the following section.

Connectionist Temporal Classification

Temporal classification means labeling unsegmented sequence data. This is a common challenge in sequence-learning tasks like speech recognition and hand-

writing recognition. The CTC was introduced with the aim of using RNNs to perform temporal classification [3]. It has a softmax layer that outputs probability scores. Thus, the CTC network works as a classifier. We are interested in training the network by 1) calculating a loss function and 2) decoding the character scores to find the most probable utterance. Both tasks are solved with the CTC operation.

CTC models take acoustic features as input and are trained to predict a symbol for each frame [3]. The symbols can either be graphemes (for orthographic transcription), or phonemes (for phonetic transcription). The softmax output layer has $L+1$ units, where L is the label set size. The first L units represent the probability of observing the corresponding labels, while the last unit represents the probability of observing a blank, whose role is explained below.

CTC networks output probability scores for each $L+1$ unit at each time step. This results in a matrix of size $(L+1) \times T$, where T is the number of time steps in the input sequence. This matrix, or trellis, represents all possible ways of aligning all possible label sequences with the input sequence. The probability of one specific label sequence is found by summing the probabilities of all its possible alignments, or paths, through the trellis. CTC assumes that the network outputs at different times are conditionally independent. This is ensured by requiring that no feedback connections exist from the output layer to itself or the network [3]. This feature can somewhat limit the modeling capabilities of CTC, which is addressed by the seq2seq models.

When the CTC network encodes the input data, it outputs the most probable label at each time step. It may output blanks to support the decoding task, for example, to mark double letters. The blank symbol can also be placed between distinctive graphemes. All blanks are removed during decoding for the final output. We use the word *hello* as an example. For example, it can first be encoded as *hh_eeellll_ll_oo* (depending on the duration of each grapheme). For the decoding task, the network reduces all duplicate characters to one single character if there is no blank in between. This would give the current output *h_el_lo*. Then it removes all blanks and puts the graphemes together to form a word, i.e., *h el lo* $\rightarrow$ *hello*. As we have seen in this example, there might be many more observations than output labels, since multiple time steps can correspond to the same grapheme (or phoneme). Since the CTC reduces duplicate characters, it means that it disregards frame alignment. It does not care about the duration of each grapheme/phoneme but is only interested in outputting a correct final transcription.

Given an input sequence $\mathbf{x}$, the CTC should output the most likely label sequence $\hat{\mathbf{l}}$. This is given by

$$\hat{\mathbf{l}} = \underset{\mathbf{l} \in \mathcal{B}}{\operatorname{argmax}}\, p(\mathbf{l}|\mathbf{x}), \tag{3.7}$$

where $\mathcal{B}$ is the set of all possible label sequences l.

Instead of maximizing this likelihood directly, we can optimize the model parameters by minimizing the loss function $\mathcal{L}$. The loss function minimizes the negative log likelihood of Equation (3.7) and is given by

$$\mathcal{L} = -\sum_{l \in \mathcal{B}} \log p(\mathbf{l}|\mathbf{x}). \tag{3.8}$$

Using backpropagation, the model parameters are chosen so that $\mathcal{L}$ is minimized.

Finding the most probable label sequence is known as decoding. Several decoding algorithms can be used to find the most likely label sequence. Greedy decoding computes the argmax, i.e., it chooses the label with the highest probability at each time step and combines these labels to output a label sequence. It only looks at one label at a time and does not care if the output consists of just random characters that do not form lexical words. Beam search decoding is more complicated. It finds several likely labels (or words) at each time step and uses a language model to compare successive labels to find the most probable sequence. Since the greedy decoder is outputting the most probable label at each time step independently of the neighboring labels, it will not necessarily output vocabulary words, i.e., it may output words that do not exist in a lexicon. Since the beam search decoder is using a language model, it will opt to output vocabulary words. Thus, it is more likely that the output from beam search decoding consists of lexical words, and again more likely that the word error rate is lower compared to greedy decoding.

In the case of transcribing Norwegian speech, it is challenging to choose an appropriate language model for beam search decoding. As introduced in Section 3.3, some dialects are more like Nynorsk, while others are closer to Bokmål. Thus, which LM to use depends on which dialect is spoken. Often, the ASR system should be able to handle all Norwegian dialects, so the LM should be trained with both Nynorsk and Bokmål data.

3.8.5 Forced alignment

In speech recognition, we are often interested in the exact alignment of transcription to its corresponding audio. Knowing the exact transitions between the phonemes in an utterance can be a practical debugging tool for improving a speech recognition system. Traditionally, frame-level aligned transcriptions are obtained by manually transcribing audio files. This is time-consuming and requires linguistic expertise. Forced alignment is an algorithm that can solve this problem automatically. An audio file and its corresponding transcription are given to the algorithm that aligns them by time. This is illustrated in Figure

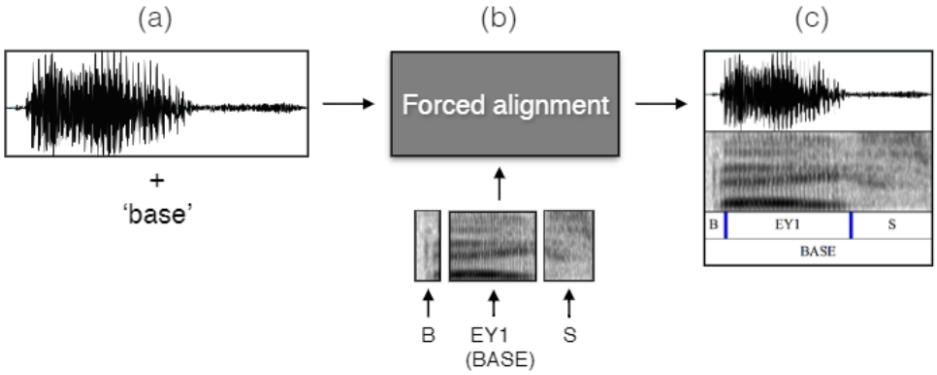


Figure 3.13: Illustration of the functionality of the forced alignment algorithm [31].

3.13. The aligned transcriptions obtained from forced alignment are not as precise as those obtained from manual transcription, but if time is restricted for a certain alignment task, forced alignment may be preferred. Forced alignment contrasts with "typical" automatic speech recognition tasks, where the correct transcription of an input speech signal is unknown, and the ASR system tries to predict the correct transcription.

Most forced alignment tools use an ASR system to extract acoustic features from the audio file. Next, it uses either HMM or dynamic time-warping to find the most likely alignment [9]. In the case of dynamic time-warping, the ASR model uses ground truth transcription to synthesize audio and computes MFCC features from the real audio data and the synthesized data. The next step is to use dynamic time-warping to find the minimum cost path transforming the synthesized audio file into the real audio file. This gives the most probable alignment of the text to the audio.

From the theory given in Section 3.8.4, we see that using ASR models trained with the CTC framework to extract acoustic features for forced alignment can be challenging since CTC disregards frame alignment information.

Chapter 4

Methods

As introduced in Chapter 2, the aim of this project is to gain knowledge about what information is encoded in an E2E ASR model adapted to Norwegian speech. This chapter presents the methods used to conduct the analysis. Section 4.1 explains how the internal representation of the model layers is analyzed in terms of grapheme classification. Section 4.2 explains the analysis of whether the information encoded in the ASR model is dialect-dependent.

Some parts of the text from this chapter were reported in Lunde *et al.* [1].

4.1 Modeling the ASR model's internal phonetic and orthographic representations

In order to gain knowledge about the E2E ASR model's interpretability, we compared the orthographic and phonetic representations of its layers. The E2E ASR model trained with the CTC framework consists of several CNNs for acoustic feature extraction, followed by several recurrent layers and, lastly, a fully connected layer with softmax, as introduced in Section 3.8.4. Since we modeled the phonetic representations in Lunde *et al.* [1], we needed to model the orthographic representations in this experiment in order to do the comparison of phonetic and orthographic representations. We used the hidden activations from the different recurrent layers of the model to perform grapheme classification, for gaining knowledge about the orthographic information encoded in each layer. We tried several methods and chose the best solution for the analysis of the model's interpretability. The attempted methods are presented in the following sections.

4.1.1 Framewise classification

To maintain consistency with the phoneme classification conducted in Lunde *et al.* [1], we wanted to conduct the grapheme classification task in the same manner, i.e., with framewise classification. This is a challenging task if the orthographic transcriptions are not frame-level aligned, such that the exact alignment of the graphemes to the audio is unknown.

Phoneme-to-grapheme mapping

There are several ways to solve this problem. One solution for obtaining frame-level aligned orthographic transcriptions is to apply phoneme-to-grapheme mapping to the phonetic frame-level aligned transcriptions. This is a complicated task, since there is not a one-to-one mapping from phonemes to graphemes, in addition to other irregularities. Some challenges are given below.

1. Silent consonants: many Norwegian words have a silent consonant. E.g., the word *godsnakk*. In phonetic form, this is transcribed as *gusnak*. Thus, there is no symbol in the phonetic transcription that can be mapped to the grapheme *d*.

2. Many of the vowel phonemes can be mapped to different kinds of graphemes, for example:

   (a) The phoneme *u* can be mapped to the graphemes *u* and *o*.
   For example: b*u*kk vs. sm*o*kk.

   (b) The phoneme { can be mapped to the graphemes *e* and *æ*.
   For example: v*e*rre vs. f*æ*rre.

   (c) The phoneme *O:* can be mapped to the graphemes *o* and *å*.
   For example: *o*g vs. l*å*t.

3. There are also several examples of consonant phonemes than can be mapped to various graphemes, for example:

   (a) The phoneme *C* can be mapped to the graphemes *kj*, *k* and *kkj*: *kj*øpe vs. *k*yrne vs. i*kkj*e

   (b) The phoneme *S* can be mapped to the graphemes *sj* and *sk*: *sj*al vs. *sk*ildring

   (c) The phoneme *j* can be mapped to the graphemes *gj* and *g*: *gj*ennom vs. *g*i.

4. When a phoneme can be orthographically transcribed by two graphemes, e.g., $S \rightarrow sj$, it does not make sense to align the letter *s* with one part of

the segment and the letter *j* with another because the combination of the two graphemes represents a single sound [20].

As we have seen, many factors make phoneme-to-grapheme mapping a difficult task. Thus, we decided to use another method to align the graphemes to the audio.

Forced alignment

Forced alignment can be applied to align the graphemes from the true transcriptions to the audio for obtaining frame-level aligned transcriptions. We used the ASR model to extract acoustic features from the training data. Using dynamic time warping, we found the most probable path through the trellis for aligning the graphemes, as explained in Section 3.8.5.

The obtained alignments were too poor to use as ground truth for the grapheme classification task, so we concluded that this method was insufficient for our experiments. An example of an obtained alignment using forced alignment is given in Appendix B, to illustrate that this method was insufficient.

4.1.2 Sequence classification

Since the results from the forced alignment were poor, we investigated a second method for performing grapheme classification that did not require frame-level transcriptions. This was done through sequence classification, which means classification of the whole input sequence, not per frame. To compare the orthographic representations of the model's RNN layers, we designed supervised classifiers consisting of only the last part of the ASR model, i.e., a fully connected layer with a softmax. We designed several such classifiers, the number of classifiers corresponding to the number of RNN layers in the ASR model. Each classifier was fed with activations from the corresponding RNN layer of the ASR model. That is, when the ASR model is trained, it computes intermediate activations from each RNN layer. We stored these activations and used them as training sets for the supervised classifiers. E.g., classifier one was fed with activations from the first recurrent layer, classifier two was fed with activations from RNN layer two, and so on.

Using the orthographic transcriptions as ground truth, the classifiers are trained to output orthographic text (i.e., sequences of graphemes). Through the softmax layer, the supervised classifiers give out probability scores for each grapheme at each timestep. A decoder can then used to find the most probable output sequence. We compare the predicted outputs of the simplified model with the ground truth orthographic transcriptions for calculating the WER and the CER of the classifiers' outputs. The error rates are used to compare the

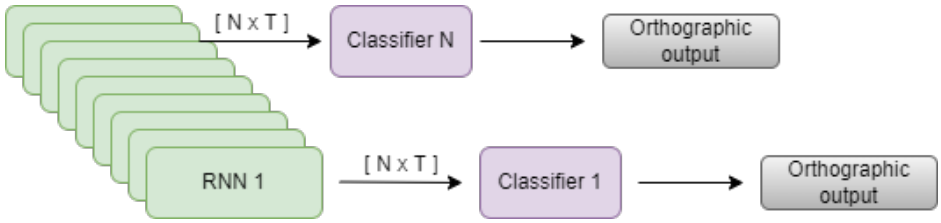


Figure 4.1: Illustration of how the supervised classifiers are trained. The activations from the ASR model's RNN layers are fed to the classifiers that are trained to output orthographic text. The size of the hidden activations is the number of hidden units in each RNN layer, N, times the number of time steps T in the sampled input audio file.

orthographic representations of the ASR model's recurrent layers, and to compare these results with the recurrent layers' phonetic representations (which was computed in Lunde *et al.* [1]). Figure 4.1 illustrates how the supervised classifiers are trained.

The supervised classifiers aim to reproduce either the total ASR model (the last classifier) or a part of the ASR model (the other classifiers) and performing grapheme classification. The results from the grapheme classification can give us information about the orthographic information encoded in each model layer. The supervised classifier trained with the activations from the last RNN layer is in fact equal to the original ASR model, since they give the same input (activations) to their final layer (the fully connected layer with softmax). Thus, we expect approximately equal performances for these two models. For the other models, which are trained with activations from the lower layers of the ASR model, we expect worse performance, but this is a way to examine the information encoded in each of the recurrent layers.

We trained each classifier for several epochs by dividing the train set into mini-batches. We divided the test set into smaller subsets and computed the WER and CER of each subset. These results were then used to compute the mean and standard deviation of the total test set. In this way, we could analyze whether the performance differences between the layers were statistically significant[1], by comparing the mean and standard deviation of each layer.

[1]I.e., there is $1\sigma = 68\%$ probability that the differences are not caused by statistical fluctuations

4.2 Analyzing dialect dependency

For the investigation of whether the information encoded in the ASR model were dialect-dependent, we tested the ASR model on different dialect groups and computed the WER and CER of each group. The results were compared to find out which dialects are best recognized by the model. The same methodology was used to compare the error rates between native and non-native speakers.

Further, we investigated whether the performance differences between the dialect groups were caused by dialectal variations. This was done by analyzing the model's output thoroughly and looking for errors that occur more often. We conducted the analysis by comparing the ground truth orthographic transcriptions with the model's predicted transcriptions. For this comparison, it is crucial to align the true and predicted transcriptions correctly when analyzing errors. Remember the example from Section 3.7.2, "the girl is kind" vs. "girl is very kind". If several sentences are aligned wrongly, the wrong words are compared and thus the results will not be valid for a detailed analysis. Since it is difficult and time-consuming to make a script that automatically aligns all sentences correctly, we chose to analyze the transcriptions manually. We did not perform a quantitative analysis by counting all true and wrong predictions of each word of interest, but did rather look for consistency in when the model fails or succeeds in recognizing a certain word. For each specific word we were interested in, we printed all ground truth transcriptions containing that word and the corresponding predictions from the model. In cases where we were interested in how some specific words were pronounced by the speaker, we listened to the corresponding sound file in addition to reading the transcriptions. For example, we analyzed all examples where the word *hvor* (English: where) were present in the true transcriptions. Since people from the north usually say "kor", the model had problems with recognizing these samples. If the model managed to recognize someone from the north saying *hvor*, we listened to the audio file to check if the speaker pronounced the word as "kor" or more similar to the Bokmål form, like "vor".

Chapter 5

Experiments

In this chapter we present the data sets and implementation details used to conducted the experiments.

Some parts of the text from this chapter were reported in Lunde *et al.* [1].

5.1 Data sets

This section gives a brief overview of the data sets used for training and testing the E2E ASR model.

5.1.1 NB Tale

The NB Tale data set consists of three modules. The corpus contains speech data from 380 different informants. In modules 1 & 2, the speakers read manuscript sentences in a noise-free environment. Module 1 contains audio from 240 native Norwegian speakers, while module 2 contains audio from 120 non-native speakers and a validation group with 20 native speakers. There are both orthographic and phonetic corresponding transcriptions, where the phonetic transcriptions contain frame-level annotations. There are in total 7600 sentences in the corpus, where 2163 of them are unique. Phoneme distribution was considered when selecting sentences to include as many phonetic sounds as possible. The speakers are chosen so that there is a balanced variety of dialects, gender, and age. Table 5.1 shows an overview of the two modules.

The native speakers in module 1 are ordered in 12 different dialect groups and the speakers in module 2 are ordered in 12 different geographic areas. An overview of the dialect groups is given in Table 5.5 and the geographical distribution of the non-native speakers in Table 5.4 in Section 5.4.

Table 5.1: Overview of NB Tale modules 1 and 2.

	Module 1	**Module 2**	**Total**
Native/non-native speakers	Native	Both	
Total sentences	4800	2800	7600
Unique sentences	2163	1263	3426
Speakers	240	140	380

NB Tale module 3 contains orthographically transcribed spontaneous speech. 365 of the speakers from modules 1 & 2 are recorded in module 3, and the recordings are done in the same acoustic environment. Each speaker is recorded on average for two minutes, including pauses, while talking about a self-chosen topic. The recording of each speaker is divided into several audio files, each file representing a sentence spoken by the speaker. The corresponding transcriptions are annotated in Bokmål.

5.1.2 Nordisk Språkteknologi (NST)

The NST corpus is made for ASR development. It contains audio recordings of read-aloud speech sampled in a noise-free environment at a sampling frequency of 16 kHz. Phoneme distribution is considered when selecting manuscript sentences, which are based on sentences from NST's Norwegian corpus. The sentences are written in Bokmål. The corpus is divided into an official test set and a training set. Approximately 1000 speakers are recorded. They were selected with the intent of covering a balanced variety of gender, dialects, and age. The total NST corpus for Norwegian speech consists of approximately 540h of audio.

Since the speakers are reading manuscript texts aloud that are written in Bokmål, all speech is planned. The data set is thus very clean and unrealistic because phenomena that are typical for spontaneous speech, like hesitation and dialectal forms, occur seldom.

5.1.3 Norwegian Parliamentary Speech Corpus (NPSC)

The Norwegian Parliament Speech Corpus (NPSC) contains audio recordings and corresponding transcriptions from meetings at Stortinget, the parliament of Norway, sampled at a sampling frequency of 48 kHz. The corpus contains in total 140h of audio recordings, including pauses, based on 65.000 sentences. The recordings contain partly read-aloud speech and partly spontaneous speech, which mainly is monologues and to a little degree dialogues. Thus, this data set contains more realistic speech than the NST. Since a part of the data is

Table 5.2: Training sets from NB Tale modules 1 & 2 for the supervised grapheme classification task.

	Train set	**Test set**
No. of samples	6057	352

spontaneous, the NPSC consists more likely of dialectal forms and hesitations than the NST corpus.

5.2 ASR model

The ASR model takes as input a sequence of spectrograms extracted from the training samples. The spectrograms are obtained using a Hann window with a window length of 20 ms and a window stride of 10 ms. The sampling rate is $F_s = 16$ kHz.

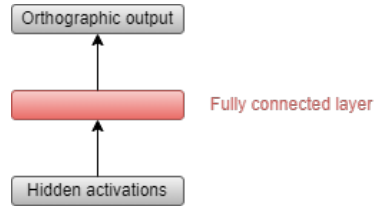
The model is based on Deep Speech 2 [2], a family of ASR systems based on deep neural networks and the CTC framework [3], which is trained to predict a sequence of graphemes for a given input speech signal. It processes each audio feature in the context of neighboring features using convolutional layers and then combines them using bidirectional RNN layers. We use a model pre-trained by Telenor Research based on an existing PyTorch implementation[1].

For transcription purposes, the model relies on an external n-gram LM during decoding. However, we are only using an LM in the dialect analysis task, not for the grapheme classification task. In that case, we are only interested in understanding the encoding of orthographic and phonetic information.

The ASR model has equal architecture to the model used for the phoneme recognition in Lunde *et al.* [1]. The architecture consists of three strided convolutional layers, nine 1200-dimensional, bidirectional GRU layers, and a fully connected layer with softmax. This is illustrated in Figure 5.1a. The AM is sequentially trained with 300h of NST data, 112h of NPSC data, and fine-tuned on data from NB Tale modules 1 & 2. The model achieved a CER of 8.0 % and a WER of 34.3 % on the test set after being trained for 30 epochs. The number of samples in the training and test sets from NB Tale modules 1 & 2 are displayed in Table 5.2.

[1] Documentation available at `https://github.com/SeanNaren/deepspeech.pytorch`.

(a) Original ASR model [1].



(b) Supervised classifier for grapheme classification.

Figure 5.1: Comparison of the architectures of the original ASR model and the supervised classifier.

Table 5.3: Overview of the grapheme label set.

Vowels	Consonants	Special symbols
a	b	_
e	c	' '
i	d	é
o	f	ü
u	g	
y	h	
æ	j	
ø	k	
å	l	
	m	
	n	
	p	
	q	
	r	
	s	
	t	
	v	
	w	
	x	
	z	

5.2.1 Grapheme label set

The grapheme label set gives the possible characters the ASR model can output. The set contains the 29 graphemes in the Norwegian alphabet, in addition to the blank symbol, _, for the CTC decoding, a space symbol, ' ', and the special characters é and ü. This gives 33 labels in total. An overview of the label set is given in Table 5.3.

É and ü are occurring very seldom in the Norwegian language, and the ASR model is confused to output these labels more often than it should. Thus, we would recommend not including this in the label set for future work. We used the same label set when training the supervised classifiers to maintain consistency. Changing the label set would require training the ASR model all over again with the new label set, which would take several weeks. Due to time restrictions, we found this label set sufficient, and leave the improvement of it for future work.

5.3 Sequence classification of graphemes

Since the ASR model consists of nine RNN layers, we trained nine supervised classifiers for the sequence grapheme classification. The supervised classifiers' architecture is illustrated in Figure 5.1b, and the final training setup is illustrated in Figure 5.2. We used a greedy decoder for decoding the character scores. It does not include a language model, as discussed in Section 3.8.4. Thus, our model predicts each grapheme independently and is not affected by an LM to output grammatically correct words, which results in more utterance-like transcriptions and thus reliable classification results in terms of orthographic representations. We used the train sets from NB Tale mod. 1 & 2 (Table 5.2) for training and testing the classifiers. Using the corresponding true, *normalized* transcriptions as ground truth, the supervised classifiers were trained to output correct transcriptions given the hidden layer activations as input. The normalized transcriptions contain no capital letters and all punctuation marks are removed. Nasal and vocal hesitations are replaced with the blank symbol _.

Training parameters

We trained each supervised classifier for 20 epochs, using batches of size 32 (which gives $6057/32 = 190$ batches per epoch). Due to time restrictions, we chose 20 instead of 30 epochs, which was the number of epochs used to train the ASR model introduced in Section 5.2. Since used a relatively small training set, the models are learning the representations quickly, so that the model performance converges quickly for the first epochs. This makes the performance difference between epochs 20 and 30 negligible, so we found 20 epochs sufficient for our analyses. We divided the test set into 11 equal subsets containing 32 samples and computed the WER and CER of each subset for computing the mean and standard deviation of each recurrent layer.

For training the supervised classifiers, we used the training parameters listed below:

1. Loss criterion: CTC loss[2]
2. An optimizer based on stochastic gradient descent with
   (a) Weight decay = 10^{-5}
   (b) Momentum = 0.9
   (c) Learning rate = $3 \cdot 10^{-4}$, optimized with a linear scheduler

[2]Documentation is available at `https://github.com/SeanNaren/warp-ctc`.

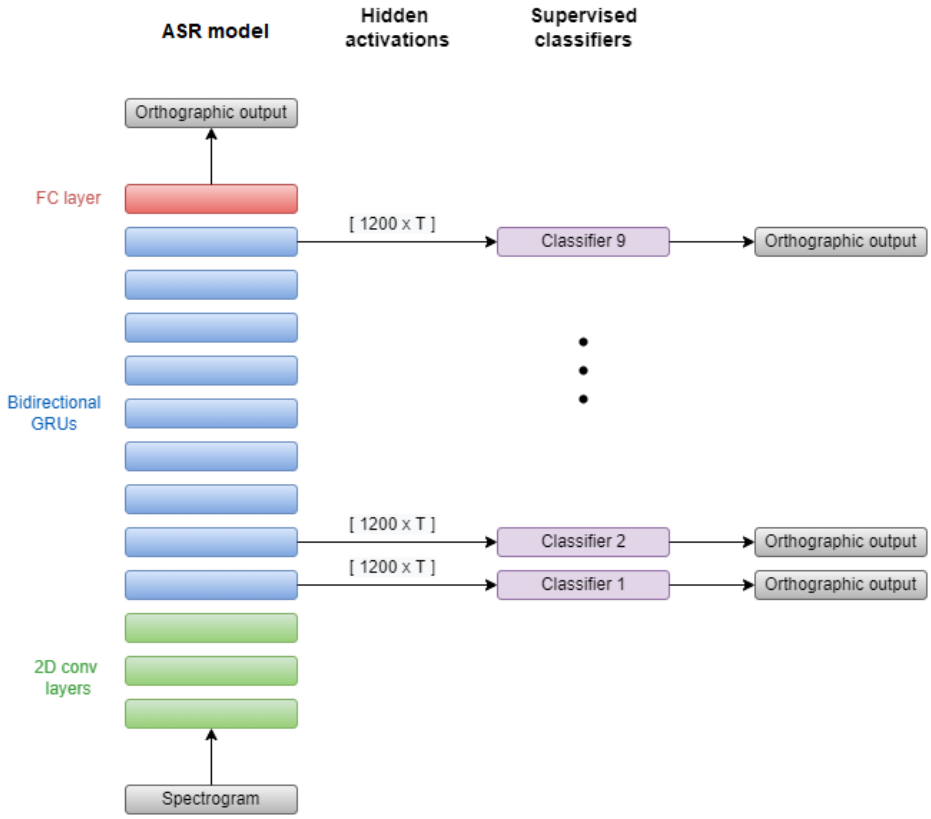


Figure 5.2: Illustration of how the nine supervised classifiers were trained, by extracting activations from the nine RNN layers of the ASR model.

The linear scheduler was the only parameter that differed between our training setup and the training setup for the original ASR model when fine-tuned on NB Tale. By including the scheduler, we expect a slightly better performance than when training without, as discussed in Section 3.7.4. We did not divide the training set into train and validation sets but tested directly on the test set after each training epoch. This was done in order to reproduce the results from the original ASR model, which was trained and tested in the same manner, to ensure that we get a valid analysis of the interpretation of each model layer. The motivation for omitting the validation set when the ASR model was fine-tuned, was to use as much data as possible for training.

5.4 Analyzing dialect dependency

5.4.1 Native vs. non-native speech

Experimental setup

We tested the ASR model on the test set from NB Tale modules 1 & 2 (Table 5.2) and used a greedy decoder to decode the output. Using the normalized true transcriptions as ground truth, we compared the error rates between the different dialect groups or geographical groups, respectively.

Table 5.4 shows the distribution of the non-native speakers from the test set of module 2. Recall that the last group, group 24, is a validation group that contains native Norwegian speakers. Table 5.5 displays the geographical distribution of the native speakers in module 1.

Expected results

There are only between 4-14 samples per group in the test set, based on between 1-4 speakers. This is not enough to conclude whether performance differences between the groups are caused by dialect-specific variations. Since there are that few speakers in the test set, differences may as well be caused by speaker variations instead of dialect variations. Anyways, we find it sufficient to analyze the test set to look at the differences between native and non-native speakers and to get the impression of which ethnicities are best recognized by the model.

We assume that the model achieves lower error rates for native speakers than non-native speakers since Norwegian speakers are more likely to pronounce Norwegian phonemes and words correctly compared to non-native speakers. Additionally, we expect that speakers with a native language similar to Norwegian have fewer problems with pronouncing typical Norwegian phonemes, which results in lower error rates. European languages are in general more similar to

Table 5.4: Overview of the origin of the non-native speakers in NB Tale module 2.

Group nr.	Native language	No. of samples
13	Greek, Tamil, Indonesian	9
14	Russian	11
15	Korean/Spanish	10
16	Thai/Japanese, Chinese	14
17	Kurdish, Persian	11
18	Danish, Swedish	13
19	German	4
20	French, Italian, Spanish	9
21	English	12
22	Hungarian, Finnish, Turkish	8
23	Maninka, Oromo, Swahili	11
24	Norwegian (Bergen, Stord, Oslo)	14
Total		**126**

Norwegian than those from other continents. Since Norwegian is part of the Germanic languages, it shares many characteristics with other languages from that group. It has most in common with the other Scandinavian languages (Danish and Swedish, also in the North-Germanic group). One can in fact say that these languages are mutually intelligible [4]. The languages from the West-Germanic group, like English, German and Dutch, are also relatively similar.

5.4.2 Dialect analysis

In terms of the dialect analysis, we tested the model on NB Tale module 3. Since module 3 contains the same speakers as those in modules 1 & 2, it means that the ASR model is trained on the speakers it is tested on (but of course not the same recordings), which should be an advantage in terms of recognition accuracy. A big difference between modules 1 & 2 and module 3 is that module 3 contains spontaneous speech. As discussed in Section 3.4.1, it is considerably more challenging to recognize spontaneous speech than read-aloud speech. This is emphasized when working with dialects, since many dialect-specific words are left out when the speakers are reading selected sentences, but will occur when the speakers are talking freely. Thus, we expect that the error rates will be noticeably worse when the model is tested on module 3.

Table 5.5: Samples per dialect group in the test sets from NB Tale modules 1 and 3.

Group	Counties	Mod.1	Mod.3
1	Finnmark & Nord-Troms	21	314
2	Troms & North of Nordland	21	374
3	Nordland	24	311
4	Brønnøysund, Trøndelag, Nordmøre	14	440
5	Trøndelag	17	383
6	Møre og Romsdal	18	409
7	Sogn og Fjordane	23	396
8	Rogaland & Bergen	16	382
9	Rogaland & Hordaland	18	392
10	Agder	20	367
11	Østlandet	18	420
12	Oslo & Akershus	16	394
Total		**226**	**4582**

Experimental setup

As introduced in Section 5.1, the native speakers from the NB Tale data set are divided into 12 broader dialect groups and the non-native speakers are divided into 12 groups based on geographical origin. This division is done by Lingit [32] and is mainly in line with the groupings of Skjekkeland [16] in Figure 3.2. We chose to keep these groups for our analysis since Lingit has linguistic expertise, so we assume they have made reasonable choices for the divisions. An overview of the dialect groups and their distribution in the test set from module 3 are given in Table 5.5. The groups are mainly consistent with the nine groups in Figure 3.2, but there are some changes. There are more than 300 samples per dialect group as displayed in Table 5.5, which we consider a sufficient number for analyzing dialect variations. Approximate borders between the 12 groups used for the dialect analysis are given in Figure 5.3

We used both greedy decoding and beam search decoding when testing the ASR model on module 3. The beam search decoder relies on a five-gram language model that is trained on approximately 13 million sentences from the NST newspaper corpus[3]. The corpus contains Bokmål text from Norwegian newspapers published between 1984-2011. Thus, the corpus does not contain, or contains to a very little degree, dialectal and Nynorsk forms.

[3]Documentation is available at `https://www.nb.no/sprakbanken/ressurskatalog/oai-nb-no-sbr-12/`

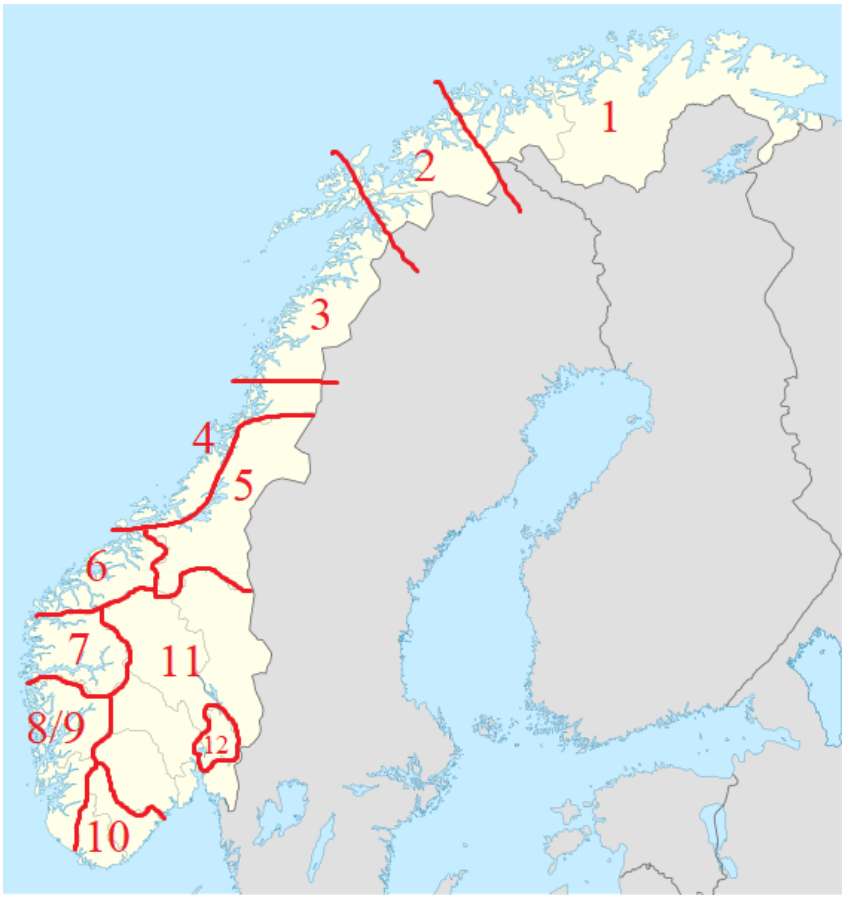


Figure 5.3: The borders between the 12 dialect groups used for our dialect analysis. The original map is taken from `https://no.wikipedia.org/wiki/Fil:Norway_location_map.svg`.

Using the normalized true transcriptions as ground truth, we computed the WER and CER for each dialect group and both decoding schemes. In the normalized transcriptions from module 3, most dialectal words are substituted with their Bokmål form, e.g., "jæmmantifra" is transcribed as "hjemmefra" (from home). Thus, the normalized transcription might differ noticeably from the spoken utterance.

We compared our results with the findings from Solberg and Ortiz [12], which tested an ASR model with a similar configuration as ours on NB Tale module 3, as introduced in Section 3.1. Recall that they achieved lowest the WERs for the dialects from the southeastern parts of Norway, and the highest error rates for the dialects from the western and middle parts of Norway.

In terms of the words analysis from the model's predicted outputs, we chose to look at some of the linguistic phenomena that differ the most between dialects, which were introduced in Section 3.3. This is for example verbs (where apocope occurs in some dialects) and function words like pronouns and interrogative words. These word classes usually have a high occurrence in languages. Thus, it is likely that they will appear often in the test data, so that we have many samples to analyze. The specific words we analyzed are listed below.

1. First personal singular pronoun, ***jeg*** (I), which has plenty of dialectal variants, e.g., "æ", "e", "i" and "je". The distribution of the dialectal variants is given in Figure A.5 in Appendix A.

2. The interrogative words ***hva*** (what), ***hvordan*** (how), and ***hvor*** (where) have many dialectal variants. In the urban eastern parts of Norway, they pronounce these words very similarly as they are written, with a v-sound (note: the *h* is silent and thus not pronounced in either of the words). In other parts of the country, the v-sound is often replaced with a kv-sound or k-sound. *Hva* is often pronounced as "ka" or "kva", *hvordan* as "koss", "kossn", "korleis" or "kordan" and *hvor* as "kor". Figure A.7 in Appendix A shows the geographical distribution of the pronunciation of words beginning with *hv*.

3. The verbs ***se*** (see), ***holde*** (hold), ***komme*** (come), ***være*** (be) and ***dra*** (go). Some of these words can sound completely different in dialectal form, for example "fær" (which means *dra*) and "sjå" (se).

Expected results

In the case of beam search decoding, since the LM is trained on Bokmål data, we expect the ASR model to perform better on the Bokmål-like dialects, especially those from the southeastern parts of Norway (which include the urban eastern parts). Ideally, we would use a language model trained on more Nynorsk data as

well, since we are testing on a wide variety of dialects, many of them being more similar to Nynorsk. Available language models trained with a more balanced amount of Bokmål vs. Nynorsk data are trained with considerably less data than the chosen one, which would probably result in higher error rates. We favored higher model accuracy so that errors are less likely caused by the model instead of dialect variations, though the model might be biased towards the eastern dialects.

As introduced in Section 3.4.1, spontaneous speech is more difficult to recognize than read-aloud speech. Since the ASR model is fine-tuned on read-aloud speech (NB Tale modules 1 & 2), we must think about which factors change when switching to spontaneous speech under testing (module 3), since we expect these changes to cause errors. When people with different dialects read Bokmål-written texts aloud, we suggest that they usually read the Bokmål words as they are and to a little degree substitute them with dialectal variants. Still, we expect that they pronounce the words with their dialect's characteristic phonemes. For example, we expect that people from Stavanger read with guttural R and soft consonants, which are not present in many other dialects. Thus, we assume that the phonetic inventory is the biggest distinction between dialects for read-aloud speech.

When people talk freely, we believe that they will still talk with the same phonetic sounds as when reading. I.e., the ASR model will get similar sounds as input for training and testing. The lexical inventory, on the contrary, will change a lot. Dialect-specific words are not present in Bokmål texts and will thus occur considerably more often in spontaneous speech. Hence, we chose to analyze the words listed in Section 5.4.2, which include some of the linguistic phenomena that differ the most between dialects,

Chapter 6

Results and Discussion

This chapter presents and discusses the results from the analyses of the E2E ASR model's internal representations. Section 6.1 shows the analysis of the model's phonetic and orthographic representations, while Section 6.2 shows the analysis of whether the information encoded in the model is dialect-dependent.

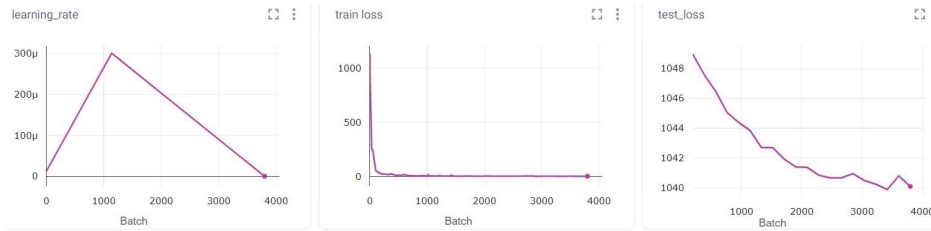
6.1 Modeling the ASR model's internal phonetic and orthographic representations

6.1.1 Results

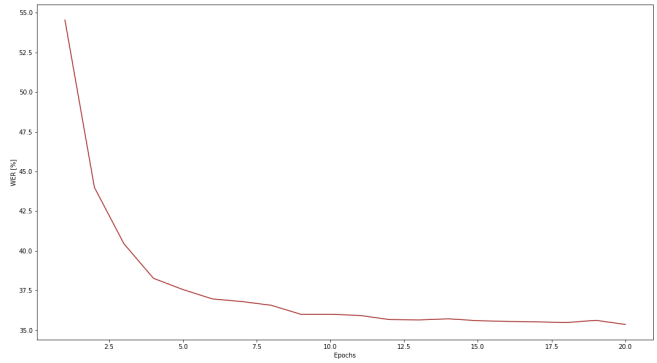
We start with representing the results from the grapheme classification of each recurrent layer, before comparing them with the previously obtained results from the phoneme classification in Lunde *et al.* [1].

The results from training and testing the supervised classifier with activations from RNN layer nine are given in Figure 6.1. For the remaining parts of the report, when we write "the results from layer X", we refer to to *the results from the classifiers trained on activations from layer X*, since these results reflect the orthographic representation of layer X. The training process goes through 190 batches per epoch, which gives $20 \cdot 190 = 3800$ batches in total. Figure 6.1a displays the learning rate and training loss per batch during training, and the test loss from testing the model after each training batch. Figure 6.1b and Figure 6.1c show the test WER and CER after each training epoch, respectively.

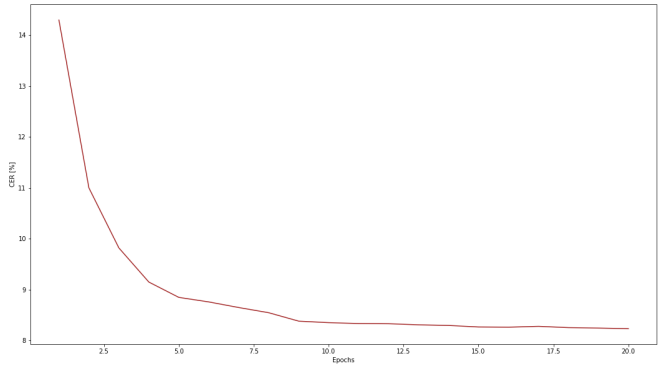
As presented in Section 5.2, the original ASR model achieved a word error rate of 34.3 % and a character error rate of 8.0 % on the test set after being trained for 30 epochs. Figure 6.1b and Figure 6.1c show that the supervised classifier trained on activations from RNN layer nine achieved a WER of 35.4 % and a



(a) Learning rate, train loss and test loss per batch during training.



(b) Resulting test WER.



(c) Resulting test CER.

Figure 6.1: The training and test results after training the model with activations from RNN layer nine for 20 epochs.

CER of 8.2 % after being trained for 20 epochs, which is almost as low as the original results. Our results would probably have been even closer to the original results if we had kept training for 30 epochs. Thus, we conclude that our method is sufficient for conducting a valid analysis of the different layers' orthographic representation and thus the model's interpretability.

Figure 6.1a shows that the model is overfitting to the training data. This could have been improved, e.g., by including regularization parameters like dropout. Due to time restrictions and the fact that we are interested in comparing the performance difference between the layers rather than achieving the best possible classification result, we chose not to focus on this improvement. Additionally, the full Deep Speech model was overfitted as well. Since we are interested in reproducing the original training conditions to get the most optimal conditions for the analysis, we find the conducted training method and the results sufficient.

The different layers' performances from the grapheme classification in terms of WER and CER are visualized in Figure 6.2. The results are summarized in Table 6.1 together with each layers' achieved CER from the phoneme classification conducted in Lunde *et al.* [1]. The graphs of the test error rates for each layer can be found in Appendix C.

	Graphemes		**Phonemes**
RNN layer	**WER (%)**	**CER (%)**	**CER (%)**
RNN 1	96.7	35.3	63.6
RNN 2	95.9	31.2	60.4
RNN 3	95.7	33.5	58.6
RNN 4	93.0	31.1	58.1
RNN 5	89.1	29.0	58.0
RNN 6	87.4	27.7	59.4
RNN 7	75.0	19.0	60.9
RNN 8	75.4	24.1	63.0
RNN 9	35.4	8.2	65.3

Table 6.1: Obtained WER and CER for each layer from the grapheme classification and the obtained CER from the phoneme classification.

In terms of grapheme classification, Figure 6.2 and Table 6.1 show that overall, both the CER and the WER decrease gradually with increasing layers. Additionally, it seems like the standard deviation increases for the higher layers, especially in terms of CER. In terms of WER, the performance of the lower layers is very poor, almost all words are wrongly predicted. There is a slight decrease in error rate going upward the layers, with a small performance jump when going to layers seven and eight, before a prominent improvement for the last layer, layer

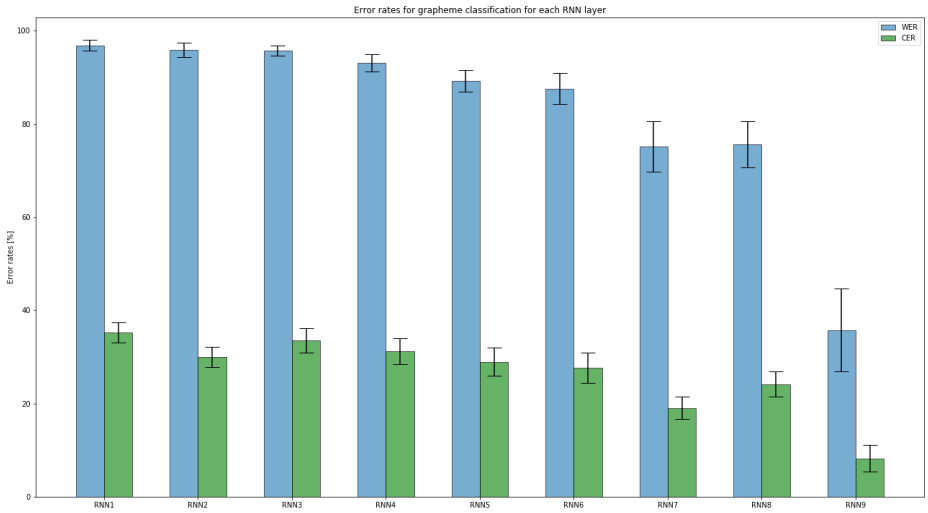


Figure 6.2: The test WER (blue) and CER (green) of the different RNN layers from the grapheme classification. The standard deviation of $\pm 1\sigma$ is displayed on each bar.

nine. The same trend is present in terms of CER, but layer two and layer seven stand out with a noticeably lower error rate. In this case, as well, the last layer achieves a considerably lower error rate than the preceding layers.

A comparison of the results from the grapheme classification with the previously obtained CERs from the phoneme classification is given in Figure 6.3.

Figure 6.3 shows that the performance varies more between the layers for the grapheme classification than the phoneme classification. While the performance improves with increasing layers for grapheme classification, it improves until layer five for phoneme classification, before it decreases for the last layers. Additionally, the character error rate is considerably better for the grapheme classification in all layers compared to the phoneme classification.

A comparison between true and predicted outputs for some samples from layers one, seven, and nine is presented below. We chose to display the output from these layers because they represent the three "performance groups": the first six layers have approximately equal performance, then layers seven and eight have a noticeable improvement, and layer nine performs considerably best.

Layer 1:

- **true:** karlsson skal dessuten ha mange plusspoeng for en fantasifull intrige

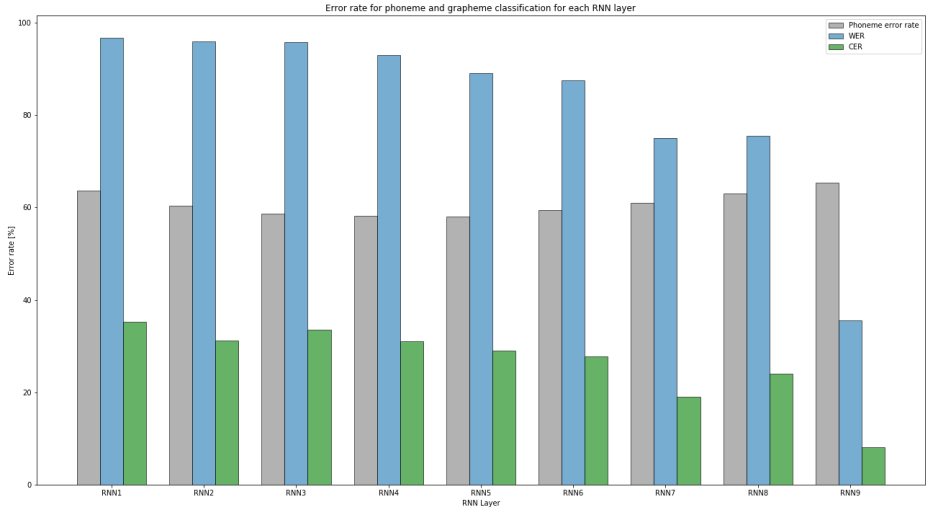


Figure 6.3: Comparison of the error rates for the phoneme frame classification (CER, grey) and the grapheme sequence classification, with WER (blue) and CER (green).

med action som bærende prinsipp
greedy: karllsanskeldesuten hammaniplus panggforen fanpsiføllintige m csen sombærandeprinsi

- **true:** i avtalen som ble undertegnet i går ble kun varaordfører og ordførertittelen fordelt
  greedy: itatarlensomble unetanetiorblkore oforrler o oforrler titrlen fordent

- **true:** men i brønnøysundregistrene står arnt selmer smeby oppført som kontaktperson for firmaet
  greedy: men i blnesune egestenesta arlt sel bme sme by o ført smktak p jo fog fima

Layer 7:

- **true:** karlsson skal dessuten ha mange plusspoeng for en fantasifull intrige med action som bærende prinsipp
  greedy: karelsan skal desuten har mang pluspoing for en fantasifulen trige med action som bæeren prinsip

- **true:** i avtalen som ble undertegnet i går ble kun varaordfører og ordførertittelen fordelt

greedy: i atalen som ble unnetenet iår blekun vre ordfører og ofører tintellen fordent

- **true:** men i brønnøysundregistrene står arnt selmer smeby oppført som kontaktperson for firmaet
  greedy: men i brnayysundregstrenesto art tselmeg smeby opført som kotaktbasjon for fyigma

Layer 9:

- **true:** karlsson skal dessuten ha mange plusspoeng for en fantasifull intrige med action som bærende prinsipp
  greedy: karlsson skal dessuten har mange plusspoeng for en fantasifullen trig med action som bærende prinsipp

- **true:** i avtalen som ble undertegnet i går ble kun varaordfører og ordførertittelen fordelt
  greedy: i avtalen som ble undertegnet i går ble kun væraorfører og orfører tiltelen fordet

- **true:** men i brønnøysundregistrene står arnt selmer smeby oppført som kontaktperson for firmaet
  greedy: men i bønnøysundregistrene stog arnt selmeks meby oppført som kontaktperson for filmae

Looking at the predicted transcripts, we see that the classifier trained on activations from layer one often outputs either very long or very short words. The long words seem to be composed of several words from the true transcription. E.g., the true transcription *"karlsson skal dessuten ha mange plusspoeng for en..."* is predicted as *"karlsanskeldesuten hammaniplus panggforen"*. Next follows an example where the predicted output consists of several short words: *"arnt selmer smeby oppført som kontaktperson for fimraet"* is predicted as *"arlt sel bme sme by o ført smktak p jo fog fima"*. Anyways, we can often see the contour of the true words since many graphemes are decoded correctly. This shows why the WER is that poor for most layers, while the CER is quite good. The classifier trained on activations from layer seven outputs transcriptions much closer to the true transcriptions. The number of predicted words is approximately equal to the number of words in the true transcription: they are not merged into very long words or split into shorter words like in layer one. Sometimes, the model wrongly divides a word into two separate ones, or it compounds two words into one word. E.g., *"ble kun"* is predicted as *"blekun"* and *"i brønnøysundregistrene står"* is predicted as *"i brnayysundregstrenesto"*. In the predictions from the classifier trained on activations from layer nine, only a few graphemes differ from the true transcription. Many words are predicted correctly. Sometimes the model wrongly

divides a word into two separate ones, or it compounds two correctly predicted words into one word. E.g., *"fantasifull intrige"* is predicted as *"fantasifullen trig"* and *"ordførertittelen"* is decoded as *"orfører tiltelen"*.

6.1.2 Discussion

Firstly, we will discuss the results from the grapheme classification, before we compare them with the previously obtained results from the phoneme classification.

We find it interesting that the test loss is approximately constant for all layers, while the WER and CER decrease, as can be seen in Figure 6.1. One possible explanation for this is that the model learns a better representation while still experiencing the same loss. Another explanation is that we have done something wrong when calculating the test loss. From Figure 6.1a, we see that the train and test loss start at approximately the same value. The train loss decreases quickly, but the test loss is approximately constant. Due to equal starting values, it seems like we calculate the test loss correctly, but it does not decrease during training as the training loss.

Overall, we see that the activations from the higher layers lead to better grapheme recognition. This is expected since the model is trained to output orthographic information from an acoustic input, and thus we expect the higher layers of the model to be more geared towards orthographic information. Interestingly, there is a small, noticeable improvement in terms of CER for layer two and layer seven. One possible explanation for why they are performing better than their next layer is that the intermediate layers are trying to be more generalized. I.e., they are designed not to output the best representation for their layer, but rather to output hidden activations that optimize the output of the last layer. Thus, the representations will not necessarily improve for each higher layer, except for the last one. We did not have time to investigate this further but leave it for future work.

Another interesting observation from the grapheme classification is that the lower layers obtain roughly the same error rates before a substantial improvement for the last layer. We ask ourselves why there is not a gradual improvement with increasing layers, including the last one? One explanation might be that since the error is computed using backpropagation, starting at the last layer, the gradient is diminishing when propagating to the first layers. This will lead to bad error computation. If the gradient is negligible for several of the first layers, the difference is probably too small to make a difference in classification performance. We ask ourselves if we would achieve the same error rates for the last RNN layer with a model consisting of only a few recurrent layers instead of nine. Are the first layers useful in terms of orthographic representation? Due to

time restrictions, we leave this for future work.

Looking at Figure 6.3, we see that in terms of phoneme classification, the model performs better towards the middle layers, while it performs better for the upper layers in terms of grapheme classification. We find these results reasonable since the model takes acoustic features as input and is trained to output graphemes (i.e., orthographic information). There is nothing with the training that forces the model to learn phonetic representation, it is optimizing for orthographic representation. This can explain why the *difference* in error rates between the layers is considerably bigger in terms of orthographic representation compared to phonetic representation. We find it reasonable that the orthographic representation has a bigger effect on the model layers than the phonetic representation, since the training is optimized for graphemic output, and that this effect leads to a bigger variation between the model layers. In the Norwegian language, there is a relatively close relationship between graphemes and phonemes. One could expect that due to this correlation, the phonetic representations improve in line with the improvement of the orthographic representations. Since the trend of the phonetic representation between the layer does not follow the trend of the orthographic representation (i.e., gradual improvement for higher layers), it seems like the model implicitly learns phonetic representation while being trained to output graphemes. These findings indicate that the model learns a better phonetic representation for the lower layers but then switches to accommodate the target classes for the upper layers. Thus, it seems like the model needs to learn some phonetic representation in order to output graphemes, which we find reasonable since it gets phonetic information as input.

We cannot directly compare the *value* of the error rates from the grapheme classification with the results from the phoneme classification since the methodology differs (sequence vs. framewise classification) and since the ASR models are trained on different data sets. Looking at Figure 6.3, we see that the CER is considerably lower for graphemes than phonemes in all RNN layers. This is natural since the ASR model used for the grapheme classification is fine-tuned on the NB Tale train set (modules 1 & 2), while the model used for phoneme classification is not, and both models are tested on data from NB Tale (modules 1 & 2). It would be interesting to conduct the phoneme and grapheme classification with the same ASR model (and preferably equal methodology) for a fairer comparison of the error rates of phonemes vs. graphemes. There are considerably more phoneme classes than there are grapheme classes (52 vs. 33 in these two studies), so we expect the phoneme classification to be more difficult. Due to time constraints, we leave this for future work.

Our results differ a little from the results of Belinkov *et al.* [6], which observed a performance drop for the last layers also in terms of grapheme classification. But their results showed that the last layers were more geared towards graphemes

Table 6.2: Error rates per group from the test set from NB Tale modules 1 & 2.

Module 1			Module 2		
Group	**WER (%)**	**CER (%)**	**Group**	**WER (%)**	**CER (%)**
1	26.7	6.7	13	56.9	13.1
2	19.2	4.2	14	47.7	12.6
3	30.3	6.2	15	64.0	13.2
4	26.7	5.4	16	62.6	21.6
5	34.0	7.7	17	36.3	7.9
6	25.9	5.1	18	34.4	6.8
7	20.2	3.7	19	46.8	16.0
8	31.4	6.5	20	45.4	13.1
9	32.6	6.6	21	34.8	7.9
10	37.1	8.7	22	44.0	8.6
11	29.4	6.3	23	45.0	9.6
12	25.5	6.0	24	29.9	7.2
Average	**28.3**	**6.1**		**45.7**	**11.5**

than phonemes, which correlates with our findings. We should keep in mind that we did not use the same models (3 CNNs and 9 RNNs vs. their model of 2 CNNs and 5 RNNs), which were trained and tested on different data, and we performed sequence classification while they were performing framewise classification. Thus, we cannot expect equal results.

6.2 Analyzing dialect dependency

6.2.1 Results

Native vs. non-native speech

This section presents the results from the analysis of the native and non-native speakers in the test set from NB Tale modules 1 & 2. The results from testing the model on each group in the test set are given in Table 6.2.

Table 6.2 shows that the mean value of module 1 in terms of CER and WER is lower than the error rates achieved when the model is tested on the whole test set (6.1 % CER and 28.3 % WER, vs. 8.0% CER and 34.4% WER on the total test set). When tested on module 2, the model achieved higher mean WER and CER compared to the total test set (11.5 % WER and 45.7 % CER), and thus higher than module 1 as well. There are big variations between the groups in module 2. Groups 17 (Kurdish and Persian), 18 (Swedish and Danish), 21

Table 6.3: Error rates achieved when testing the ASR model on module 3 with greedy and beam search decoding. The last two columns displays the relative improvement in terms of WER and CER when going from greedy to beam search decoding. Numbers in red represent the biggest value in the column, while blue numbers represent the lowest value.

	Greedy		**Beam Search**		**Rel.improv.(%)**	
Group	WER(%)	CER(%)	WER(%)	CER(%)	WER	CER
1	52.9	21.0	40.5	20.4	23.6	2.8
2	54.4	20.9	41.1	20.3	24.5	2.6
3	56.9	23.0	43.5	22.0	23.6	4.4
4	59.7	24.0	47.2	23.6	20.9	1.7
5	61.7	24.8	48.7	24.7	21.1	0.3
6	60.6	24.9	48.3	24.4	20.3	2.0
7	62.1	25.4	49.6	24.6	20.1	3.1
8	60.1	24.1	47.3	23.4	21.3	2.9
9	58.9	23.0	45.5	22.6	22.5	1.8
10	53.6	20.0	40.4	19.3	24.7	3.7
11	49.9	19.3	39.0	19.1	21.8	0.9
12	48.5	17.9	36.4	17.4	25.0	2.5
Average	56.6	22.4	44.0	21.8	22.4	2.4
St.dev.	4.6	2.5	4.4	2.5		
Rel.st.dev.(%)	8.2	11.1	10.0	11.8		

(English), and 24 (the Norwegian validation group) achieved low CERs, lower than the mean CER of the total test set. Group 16 (East-Asia) achieved the considerably highest CER. In terms of WER, group 24 achieved the considerably best result with 29.9%. Group 17, 18, and 21 also achieved a low WER. The groups with the highest WERs are group 15 (Korean/Spanish) and group 16.

Dialect analysis

The error rates from testing the ASR model on the dialect groups from NB Tale module 3 with two different decoding schemes (greedy and beam search) are displayed in Table 6.3.

Table 6.3 shows that group 12 (Oslo and Akershus) achieves the lowest character and word error rate for both decoding schemes, while group 7 (Sogn og Fjordane) in general achieves the highest error rates for both decoders. The only exception is for CER for beam search decoding, where group 5 (Trøndersk) achieves slightly worse. The other groups from the southeast and the north are

also performing well (i.e., achieving low error rates), while the groups from the middle and western parts of Norway have high error rates. There seems to be a clear correlation with similarity to Bokmål: the dialects closest to Bokmål are those achieving the lowest error rates, and thus the best results. The beam search decoder achieves slightly lower CERs than the greedy decoder (0.4 % lower average value), but considerably better WERs (11.8 % lower average). The standard deviation is approximately equal for both decoders (4.4 % for beam search vs. 4.6 % for greedy), which naturally gives a higher *relative standard deviation*[1] for the beam search decoder (9.8 % for beam search vs. 8.2 % for greedy). The relative *improvement* in terms of WER when going from greedy to beam search decoding shows that in general, the groups achieving the lowest error rates (and thus the best results) have the biggest improvement. The exception is group 11, which has a considerably smaller improvement compared to the other Bokmål-close groups. In terms of CER, the improvement seems to be random between the groups. Some of the Bokmål-close groups have a small improvement in terms of CER, and some have a relatively high improvement. The same holds for the groups most distinct from Bokmål.

Next, we will present the results from the word analysis from the model's output when tested on module 3. For clarification, we will present Bokmål forms in *italics* and dialectal variants with quotation marks, " ".

In the case of first personal singular pronouns (*jeg* in Bokmål form), there were several dialectal forms per dialect group. The most common ones were "æ", "e", "ei", "i", "eg", "je", and "jæ". From our analyses, we found that all these forms were quite poorly recognized for all groups. Often, the ASR model missed the word (i.e., did not transcribe it at all). The following examples are taken from group 1, where the speaker says "æ" instead of *jeg*:

true: og det gjorde *jeg*

greedy: g dei gjorde

beam search: og dei gjorde

true: og da tenker *jeg* at _ hvis *jeg* har flaks så klarer *jeg* å få med _ yngste dattera mi

greedy: da tenker at *visa* floks å klarar å få med yngstedatami

beamsearch: da tenker at *visa* flaks å klarer å få med yngstedame

In the last example, both decoders miss all three occurrences of *jeg*. For the second *jeg*, it seems like both models merge the spoken form "æ" with the former

[1] Relative standard deviation is a dimensionless variable computed by dividing the standard deviation by the average value times 100, i.e., $\frac{\sigma}{\mu} \cdot 100\%$.

word *hvis*, to "visa" (which is a Bokmål word that means *the folk song* in English). In other cases, the model outputs similar short Bokmål words, like *de*, *et*, and *en* instead of *jeg*. Another common mistake was to merge the dialectal form with an adjacent word, for example "har æ" (dialectal form) was transcribed as "hara" and "det vil e" was transcribed as "det ville". Examples of these two error types are given below.

Jeg pronounced as "e":

true: hvis *jeg* skulle _ tatt meg arbeid der

greedy: hvis *de* skulle tatt meg arbeid der

beam search: hvis *de* skulle tatt med arbeid der

Jeg pronounced as "æ":

true: der jobber *jeg* da i klubbstyret

greedy: der jobbe *er* da i klubstr

beam search: der jobber *er* da i klubbstyr

Jeg pronounced as "jæ":

true: *hvis jeg* drar til vestlandet så

greedy: *vise* drar til vestlandet så

beam search: *vise* drar til vestlandet så

Sometimes the decoders output the Nynorsk form "eg" instead of *jeg*. Overall, it seems like the model is better at recognizing the forms "je" and "jæ", and to a smaller degree "eg" and "ei" than "e", "i", and "æ".

Next follows the results from analyzing the verbs. The word *være* (to be) is usually pronounced as either "være" or "vær" (apocope form) in different dialects. It had a relatively low error rate for all groups, and we did not find a clear difference in recognition rate between the dialects. If recognized wrongly, the model often outputs *vær* (weather) or *var* (was), like in this example, where the speaker says "vær":

true: for å *være* klar til å gå om søndagen

greedy: for det *vær* klart i gå om søndag

beam search: for det *var* klart å gå om søndag

For the verb *se*, there is a clear difference in recognition rate between the dialect groups. When the speakers pronounce it like the Bokmål form, *se* is mostly

transcribed correctly, but it is sometimes missed or merged with an adjacent word. Most speakers in groups 4-7 (and some in 8 and 9) pronounce it like "sjå" or "sje". We found no examples where these variants were transcribed as *se*. They were often transcribed as they are pronounced, e.g., "shå" and "sjå". Using beam search decoding, it was sometimes confused with the word *så* (saw). Additionally, it was often merged with the neighboring word. Some examples of wrong predictions are given below.
Se pronounced like "sjå":

true: og det blir veldig spennede å *se*

greedy: og det blir veldig spennende å *shå*

beam search: og det blir veldig spennende å *så*

true: så vi *får se hva* det blir til

greedy: så vi *forsa* ka det bli til

beam search: så vi *forsaka* det bli til

The verb *kommer* had low error rates when pronounced like the Bokmål form. The few times it is predicted wrongly, it is transcribed as "kom" or "komm". Some speakers in groups 4-10 pronounce *kommer* as "kom" or "kjem"/"kjæm". "Kom" was as good as always transcribed as "kom" by the model, which is the preterite form of *kommer*. We did not find any example where the dialectal forms "kjem"/"kjæm" were transcribed correctly as the Bokmål form *kommer*. They were often confused with similar words like *kjemi*, *tjern*, *kjenne*, and *skjenne*, or they were transcribed similarly as they are pronounced, for example, as "kjeme" or "kjem". Some examples of the errors caused by the dialectal forms "kjeme"/"kjæm" are given below.

true: og det blir jo verre før vi *kommer* i gang

greedy: og det blir jo verre for i *kjemei* gang

beam search: og det blir jo verre for i *kjemi* gang

true: og når man _ *kommer* fra en såpass liten plass som fauske

greedy: om men *tjerntf*rønenen såpass liten plass som fauski

beam search: men *kjernesunn* såpass liten plass som fauske

true: der eldstedattera mi som bor i oslo *kommer* og _ med fly

greedy: det r elstedotre mele burd oslo *kjemme* og men fløy

beam search: det er eldste dore melbu oslo *hjemme* og med fløy

The verb *dra* was mostly transcribed correctly when pronounced similarly to the Bokmål form, but was sometimes confused with similar short words. Some speakers from groups 4 and 5 pronounced it as "fær". We found no examples where this dialectal form was transcribed as the Bokmål form "dra". It was either transcribed as "fær", or confused with similar words like *far*, *være* and *faren*, as in these examples:

true: eller så kan du *dra* en tur på fjellet

greedy: eller så kan du *faren* tu på fjellet

beam search: eller så kan du *faren* tur på fjellet

true: ting som gjør meg _ _ glad det er jeg får lov å *dra* ut på velfjorden

greedy: ting som gjør me glader i å få lov å *fær* ut på valfjorden

beam search: ting som gjør meg glad er i å få lov å *være* på velferden

The verb *holde* had a low error rate when pronounced as the Bokmål form. The apocope form "hold" had a quite low error rate but was often transcribed as "hold". The dialectal forms "held/helde" and "hald/halde" had high error rates. They were either transcribed as they are pronounced, or confused with similar words like *helle*, *heller* and *helt*. Examples:

Pronounced as the apocope form, "hold":

true: det er ikke bare lett å *holde* på dialekten sin når en er utflytter

greedy: det er ikkje bare lett å *hold* på det lektnosin når regne ut fløttar

beam search: det er ikke bare lett å *holde* på det lekne sin når regne ut flyttar

Pronounced as "held":

true: og jeg *holder* på med tredjeåret på bachelorgraden min

greedy: og *held* på med tredje året på battleggeråden men

beam search: og *held* på med tredje året på balteren menr

In the last example, we also see that *jeg* is missed by both models, as shown in previous examples.

In the case of the interrogative words *hva*, *hvordan*, and *hvor*, we found some clear patterns. The model managed almost always to recognize the Bokmål-close variants, i.e., "va", "vordan", and "vor". Thus, the error rate was very low for groups 11 and 12. Group 10 had some dialectal forms of *hvor*, which resulted

in bad recognition. Otherwise, the model performed well for group 10. For the other groups, the error rate was high. The model is mostly wrong when the words are pronounced with a k-sound or kv-sound instead of a v-sound, which was the case for the remaining groups, with some exceptions. The model performed surprisingly well on the word *hvor* for groups 8 and 9. Listening to those samples, we found that speakers from Bergen pronounce *hvor* with a v-sound instead of k, like people from the southeast. Some examples of wrong predictions of the interrogative words are given below.

Hva pronounced as "ka":

true: kanskje ikke alle som *vet hva* informatikk er

greedy: kanskje kaller som *veka* infomatikk e

beam search: kanskje kaller som *vedta* informatikk e

true: få høre om _ *hva* som foregikk på _ finnmarksvidda i påska

greedy: for å høre om *kva* som fordegikk på filnmaksvidda i påska

beam search: for å høre om *hva* som foregikk på finnmarks ida i påska

In the second example, we see that the beam search decoder transcribes "ka" as *hva*, while the greedy decoder transcribes it as "kva".

Hvordan pronounced as "kossjn":

true: og det blir veldig spennede å se *hvordan* _ det skal bli

greedy: og det blir veldig spennende å shå *korson* det skal bli

beam search: og det blir veldig spennende å så *korson* det skal bli

true: jeg tenkte jeg skulle fortelle litt om *hvordan jeg* fant ut at jeg skulle bli lærer

greedy: eg tenktes u fortelle litt om *kursenei* fan nutta det skulle bli lærar

beam search: jeg tenktes u fortelle litt om *kursene* fant det skulle bli lærar

Hvordan pronounced as "korleis":

true: ja da skal jeg snakke om et eller anna sånn at dere får høre *hvordan jeg* egentlig snakker

greedy: ja da skal e snakke om etla arnesn hafo kuf høyre *korlasi* eigentlig snakka

beam search: ja da skal jeg snakke om et la arnes han føre *korleis* egentlig snakka

When *hvor* was pronounced as "kor", it was often transcribed as "kor" or confused with similar words like *kan* and *går*. For information, "kor" is the Nynorsk form of *hvor*, but means *choir* in Bokmål. Another common mistake was to merge "kor" with an adjacent word to form a similar Bokmål word, e.g., *kommer*. Some examples:

true: *hvor* vi la grillmaten oppi

greedy: *kor* vi lagrelen mot n oppe

beam search: *korvi* lagdel mot oppe

true: og før så hadde jeg en sommerjobb *hvor man* bare klippa gress i _ mange dager

greedy: og før så hadde en somme jobb *kommar* bare klibagress i mange dager

beam search: og før så hadde den samme jobb *kommer* bare klimagass i mange dager

true: *hvor man* _ _ _ hjelper dem i _ hverdagen deres

greedy: *korman* e hjelper dem i hvardagen næmes

beam search: *forman* er hjelper dem i hverdagen nemes

6.2.2 Discussion

Native vs. non-native speech

From the analysis of the non-native speakers in module 2, we got some expected and some surprising results. The average WER and CER from the native speakers in module 1 were lower than the WER and CER for the total test set, while the average error rates for the non-native speakers in module 2 were higher. This tells us that it is more difficult to recognize non-native speakers than native speakers, as expected. Not surprisingly, the Scandinavian speakers achieved the best results among the non-native speakers. Interestingly, they perform better than the validation group in terms of WER. Among the other groups that in theory are closest to Norwegian, the English speakers (group 21) are also achieving low error rates. Group 19 performs surprisingly badly, having in mind that German also is a Germanic language. Group 17 (Kurdish and Persian) performed unexpectedly well. After listening to the speakers, we heard that one of them spoke Norwegian very well, almost fluent, which explains why this group is performing that well. The speaker's young age (born in 1990) tells us that she might be a second-generation immigrant, which can explain her proper pronunciation. Speakers from Asia achieved the worst error rates, which is unsurprising since

these languages' phonetic inventory differ a lot from the Norwegian phonemes. We had expected worse results for the African speakers (group 23) since African languages differ a lot from Norwegian.

However, we should bear in mind that we have too few samples in each group to obtain a valid analysis. There are few speakers per group (approximately six per group for the native speakers and between two-four speakers per group for the non-native speakers), so the speaker variations may be as dominant as the language variations. Anyways, the average values of the error rates from the two modules confirms that the model recognizes native speakers more easily than non-native speakers, and overall, the groups from the Germanic language group achieve lower error rates than the remaining non-native groups.

Dialect analysis

From the dialect analysis of module 3, we see that the results for the different dialect groups are consistent between the two decoders: the same groups achieve relatively low or high error rates for both decoders. Table 6.3 shows that the southeastern dialects achieved the lowest error rates, as expected. Group 12 (Oslo and Akershus) performed considerably best, followed by groups 11 and 10. As already introduced, these are the dialects closest to Bokmål. As we suggested, the dialects most far from Bokmål lexically, the ones from the middle and western parts of Norway, are those achieving the highest error rates and thus the poorest results. The two northernmost dialect groups (groups 1 and 2) performed considerably better than those from the middle and the west. As introduced previously, the lexical inventory of these dialects is close to Bokmål. Our results are consistent with the findings from Solberg and Ortiz [12], which also got the best results for the southeastern dialects, followed by the northernmost dialects, and the middle and western dialects achieved the worst results. These findings indicate the information encoded in the model is affected by dialectal variations.

Table 6.3 shows that the relative standard deviation is bigger in terms of WER for beam search decoding than for greedy decoding, which indicates that introducing an language model (LM) for the decoding task enhances the difference between the dialects. This is confirmed by the relative improvement in terms of WER when going from greedy to beam search decoding, which shows that the dialects closer to Bokmål get a bigger improvement when including an LM, they benefit more from including an LM for the decoding task. This is reasonable since the LM is mostly trained on Bokmål data. Noticeably, group 11 had a relatively low improvement compared to the other Bokmål-like groups. We find this surprising since it is the group achieving second best for both decoders. One possible explanation for this is that group 11 is very big geographically, covering almost all of eastern Norway except for the Oslo area (group 12), as can be seen in Figure 5.3. As introduced in Section 3.3, there are no strict borders between

dialects. Thus, the dialects in the areas in group 11 close to the borders of group 12 are close to the Oslo dialect and thus close to Bokmål. Going north and west in group 11, we see that it shares borders with groups 5, 6 and 7. Thus, in these areas, the dialects may be similar to these groups, and can thus be quite far from Bokmål. There are many municipalities inside group 11 that has Nynorsk as their official written standard, like Lom, Seljord, and Nissedal[2]. This is reflected in their dialects - they are closer to Nynorsk than Bokmål. This can explain why group 11 is not benefiting as much from including a language model trained on Bokmål text, compared to the other Bokmål-close dialects. Based on this, we ask ourselves whether group 11 is too big and should be divided into several subgroups since the variety between the dialects is that big. We suggest using the subgroups of *Austlandsk* (Østlandsk) given in Figure 3.2. As far as we know, there is less variance between the dialects in for example group 1 (Finnmark): more or less all its dialects are quite close to Bokmål lexically. This can explain why the northernmost dialects seem to have a stronger benefit from a language model.

Table 6.3 shows that the relative improvement when including a language model in terms of CER is random between the groups: there is not a bigger improvement for the Bokmål-close dialects. Though there is an improvement for all dialects, this improvement is considerably smaller compared to the improvement in terms of WER. This confirms that the model benefits from including an LM for decoding in terms of word recognition, but not for recognizing individual graphemes separately.

The results from the analysis of the model's output emphasize that the performance difference between the groups is caused (at least partly) by dialect variations. The analysis showed that the model mostly recognized the Bokmål-close forms correctly, but had problems with recognizing the dialectal variants that deviated considerably from Bokmål. The word *jeg* is an exception, which had a high error rate regardless of whether the dialectal form was close to Bokmål or not. This is not surprising since these forms are very short and often pronounced quickly. Since these variants mainly consist of vowels, we wonder if the model might confuse them with vocal hesitation (e.g., *eeh*), which explains why it often is missing the word completely (i.e., not transcribing it at all). We also find it reasonable that the model confuses these variants to be the ending of the preceding word (e.g., if "det vil e" is transcribed as "det ville") since it is difficult to distinguish these versions without a full understanding of the context. In general, the model had problems with recognizing short words. They were often missed by the model or merged with a neighboring word. This happened for example

[2]See the total overview of language decisions for all of Norway's municipalities (2020-01-01) here: `https://lovdata.no/dokument/SF/forskrift/2019-12-20-2114/KAPITTEL_1-4#KAPITTEL_1-4`

with the words *se* and *hvor*.

In the case of verbs, we see that apocope is not a big challenge for the model. It often managed to output the Bokmål form of the apocope forms "vær" (from *være*) and "hold" (*holde*) correctly. These apocope forms are close to their respective Bokmål form. When these words are spoken by people from the east, they are usually pronounced with the schwa sound, @[3]. For example, *være* is phonetically transcribed as /v""{4@/. Here, the *e* is substituted with the schwa, which is a less distinct sound. Thus, the difference in realization of the pronunciation with apocope or with the schwa sound is not that big.

When the vowel in the root of the verb changes, e.g., from *holde* to the dialectal form "halde/hald", the model had problems with transcribing the words correctly. In other cases, where the dialectal form differed a lot from the Bokmål version, like "sjå" for *se* and "fær" for *dra*, the model failed in transcribing the Bokmål form. In the case of the interrogative words, the model achieved a relatively low recognition rate when the pronunciation was close to the Bokmål form but struggled when the v sound is substituted with a k or kv sound, as for "kor" instead of *hvor*. The model sometimes managed to transcribe the dialectal forms "ka" or "kva" to *hva*, especially when using a beam search decoder. Expectedly, this was not the case for greedy decoding since it opts to output exactly what it hears, and both "ka" and "kva' are quite distinct from *hva*. Additionally, the different versions of *hvordan* are challenging for the model. "Kordan" is the closest version, which sometimes is transcribed correctly. "koss", "kossn" and "korleis" deviate considerably from the Bokmål version. It might be that the speakers use the words "ka"/"kva" and "kordan" both when reading Bokmål aloud and for spontaneous speech, so that these words are present in the training data, but only use the more distinctive forms "koss", "kossn" and "korleis" when they talk freely.

For both the verbs and the functional words, we found that the model often outputs the *dialectal* form correctly, i.e., the output was close to the spoken utterance, especially when using greedy decoding. For example, it transcribed "sjå" as "sjå" and "fær" as "fær", while the true, normalized transcription is annotated with the Bokmål form. In general, we found that there were mostly speakers in groups 4-9 that spoke with dialectal forms with a clear distinction from the Bokmål form, which seems to be reflected in the resulting error rates for the dialect groups. This raises the question of whether we should emphasize to output grammatically correct output, or if we should opt to output a transcription as close to the spoken utterance as possible. This will probably depend on the application. Beam search decoding clearly gives lower error rates, especially in terms of WER. For applications where the output transcription should be formal text, for example when using speech-to-text systems for writing emails

[3] You can read more about the schwa sound at `https://snl.no/schwa`

or medical records, achieving a low WER is desired. Thus, we would probably use a beam search decoder for such applications. In less formal settings, for example, when sending personal messages, many people write dialectal forms outside the dictionary. For such applications, we could accept a higher WER to get a more dialect-close output. Thus, Greedy decoding might be preferable for such applications.

When training an ASR system, it might be problematic to favor the written form instead of the dialectal form in the ground truth transcriptions, especially when those two forms differ substantially. E.g., the Bokmål form *likevel* can take the dialectal form "læll". If the model learns that utterances like "læll" are transcribed as *likevel*, it might be confused by the phoneme-to-grapheme mapping which can give wrong predictions for other words. Since the training does not force the model to learn phonetic representation, we cannot say for sure that this will confuse the model. But, as our results from Section 6.1.1 suggests, the ASR model seems to implicitly learn some kind of phonetic information. Many of the dialectal forms are rare and occur only in some dialects. Thus, these words will occur infrequently in the training data. Since neural networks need a great amount of training data, it can be difficult to learn the model all seldom, dialectal forms of a word. We find this issue interesting and important for improving ASR systems adapted to dialects and encourage future studies to investigate this further.

In our word analysis, the dialectal word is often a Nynorsk grammatical word. This is for example the case for "kor", "kjem", "sjå", and "halde". For the beam search decoding, using a language model which is trained on a more balanced amount of Nynorsk vs. Bokmål data would probably have given better results. Since we did not have such an LM available (which was trained with a sufficient amount of data), we did not have the opportunity to test this but encourage future studies to look at it. Another possibility is to let the ASR model rely on two different language models, one trained on Nynorsk data and the other one trained on Bokmål. By implementing a dialect classifier, the system can use the most suitable LM based on the classification result. If the dialect is classified from the middle or western area of Norway, a Nynorsk LM should be used, while if it is classified from the southeast or the north, a Bokmål LM should probably be used. This method relies clearly on the classifier - a wrong classification could lead to the wrong choice of LM and thus less optimal conditions for good recognition.

In general, we observe that the greedy decoder is better at recognizing dialect-specific words than the beam search decoder. This was expected since the greedy decoder tries to output exactly what it hears, while the beam search decoder combines the most probable labels with an LM to find the most probable prediction, as discussed in Section 3.8.4. Thus, if the dialectal word is not a lexical word, it is more likely that the greedy decoder transcribes it correctly than the

beam search decoder.

To summarize, our results indicate that the information encoded in the ASR model is dialect-dependent. Out of twelve dialect groups, the groups closest to Bokmål are the ones achieving the lowest error rates. Group 12 (Oslo and Akershus) is the group closest to Bokmål and stands out with the lowest error rates. It seems to be a clear correspondence between this, and the data used for training the ASR model and the data used to train the LM the decoder relies on. To develop ASR systems that can perform well on the other dialect groups, data sets containing a wide variety of dialects and spontaneous speech are needed. The decoder should rely on a language model trained on both Bokmål and Nynorsk data, or two LMs trained on each of the written languages, since many dialects are considerably closer to Nynorsk than Bokmål. Based on our results, we can discuss whether greedy decoding or beam search decoding is the most optimal choice when transcribing Norwegian dialects. We suggest that it depends on the application, as discussed previously.

Chapter 7

Conclusion

In this report, we have modeled the internal representations of an end-to-end ASR model adapted to Norwegian speech. This Chapter gives a summary of our most important findings and answers the research questions introduced in Section 2.1. Section 7.1 gives a summary of our suggestions for focus areas for future work.

Research question 1: To what extent does the model capture phonetic and orthographic information?

We trained a supervised classifier for each recurrent layer of the ASR and performed sequence classification to gain knowledge about the orthographic representation of each layer. We found that the error rates decreased gradually for the higher layers, with the last layer performing considerably better than the others. In our previous work [1], we found that the phonetic representation improved for the first layers but decreased for the upper layers. This indicates that the last layers are more geared towards graphemes than phonemes, which is reasonable since the model is trained to output graphemes. Our results indicate that the model learns phonetic representation implicitly, even though there is nothing with the training method that forces it to learn phonetic representation.

Research question 2: Is the information encoded in the model dialect-dependent?

In terms of dialect analysis, we tested the ASR model on twelve different dialect groups. The groups from the southwestern parts of Norway achieved the lowest error rates, followed by the groups from Troms and Finnmark. These are the dialect groups being closest to Bokmål. The groups from the western and middle parts of Norway, which are most distinct from Bokmål, achieved the highest error rates. We find these results reasonable since the model is fine-tuned on data from speakers reading Bokmål text aloud. The dialect differences were enhanced by introducing a language model trained on Bokmål text for the decoding task. Thus, the information encoded in the model seems to be dialect-dependent.

To verify that the model is affected by dialect variations, we investigated the model's performance on some characteristic words that can vary considerably between different dialects. The analysis showed that the model struggled with dialectal forms that deviate from the Bokmål form but achieved low error rates for Bokmål-close variants. The only exception was the word *jeg* (I), where the model struggled with all dialectal variants, also the Bokmål-close forms *je* and *jæ*. Dialectal forms that deviate from the Bokmål standard are most often present in the dialects from the middle and western parts of Norway, which are the dialects achieving the highest error rates. Thus, dialectal variance seems to affect the model.

Often, the model outputs a transcription that is equal to the pronounced word instead of the Bokmål form, especially when using Greedy decoding. This raises the question of whether we are interested in outputting transcriptions that are closer to an official written language or the spoken dialect. We think this depends on the application the ASR system is intended for. Many dialects are closer to Nynorsk than Bokmål. For these dialects, it would make more sense to annotate the true transcriptions in Nynorsk for training. Additionally, the model's output should be transcribed in Nynorsk instead of Bokmål for these dialects.

7.1 Future work

Based on the findings and discussions from our experiments, we propose several focus areas for future work for gaining knowledge about the interpretability of end-to-end ASR models and for improving such systems adapted to Norwegian speech.

7.1.1 End-to-end ASR models

For further investigation of end-to-end ASR models, we recommend performing sequence classification with a model consisting of only a few recurrent layers, to investigate whether this would affect the performance of the last layer. I.e., will the final output give the same results in terms of CER and WER as for the model with nine recurrent layers? This can be used to investigate whether the middle layers are trying to generalize their output, to facilitate the best possible final output. Additionally, we propose conducting a similar investigation of an end-to-end model's phonetic and orthographic representations as we did but using the same ASR model and classification scheme (e.g., either sequence or framewise classification) for both analyses. This would give an even better comparison of the model's phonetic and orthographic representations than we obtained.

7.1.2 ASR for Norwegian speech

Focusing on the dialect analysis, we find it important to include more spontaneous speech in the training set, so that the ASR model will see dialectal forms during training. Additionally, our results show that it is crucial to train the language model on a balanced amount of Nynorsk and Bokmål data, to achieve better recognition of the dialects that are closer to Nynorsk than Bokmål. As discussed in Section 6.2, it would be interesting to investigate the possibility of combining two LMs, one trained with Bokmål data and the other trained with Nynorsk data, combined with a dialect classifier.

Lastly, we would recommend looking more into the effects of dialectal forms. Does the model get confused when the transcription differs considerably from the acoustic input during training? And how to handle that such dialectal forms occur infrequently in the training data?

Bibliography

[1] S. R. Lunde *et al.*, *Analyzing the Hidden Layers of an End-To-End Model Adapted to Norwegian Dialects*, Dec. 2021.

[2] D. Amodei *et al.*, "Deep Speech 2: End-to-End Speech Recognition in English and Mandarin," Dec. 2015.

[3] A. Graves *et al.*, "Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks," in *Proceedings of the 23rd International Conference on Machine Learning*, Association for Computing Machinery, 2006, pp. 369–376.

[4] O. Husby and T. Høyte, *An introduction to Norwegian dialects*. Tapir academic press, 2008, p. 112.

[5] Y. Belinkov and J. Glass, "Analyzing Hidden Representations in End-to-End Automatic Speech Recognition Systems," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2017, pp. 2438–2448.

[6] Y. Belinkov *et al.*, *Analyzing Phonetic and Graphemic Representations in End-to-End Automatic Speech Recognition*, 2020.

[7] J. Garofolo *et al.*, "TIMIT Acoustic-phonetic Continuous Speech Corpus," *Linguistic Data Consortium*, Nov. 1992.

[8] A. Prasad and P. Jyothi, "How Accents Confound: Probing for Accent Information in End-to-End Speech Recognition Systems," Jan. 2020, pp. 3739–3753.

[9] J. Zhu *et al.*, *Phone-to-audio alignment without text: A Semi-supervised Approach*, 2021.

[10] A. Baevski *et al.*, *wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations*, 2020.

[11] The National Library of Norway, *Norwegian Parliamentary Speech Corpus*, The Language Bank: oai:nb.no:sbr-58, 2021.

[12] P. Solberg and P. Ortiz, *The Norwegian Parliamentary Speech Corpus*, Jan. 2022.

[13] J. P. Williams, "Phonemic Analysis and How It Relates to Reading," *Journal of Learning Disabilities*, vol. 17, no. 4, pp. 240–245, 1984.

[14] M. Skjekkeland, *Dialektlandet*. Portal, 2010.

[15] Statistics Norway, *Table 03743: Pupils in primary and lower secondary school, by official form of norwegian (c) 2002 - 2020*, Accessed: 2022-04-27, 2020. [Online]. Available: `https://www.ssb.no/en/statbank/table/03743`.

[16] M. Skjekkeland, *Dei norske dialektane*. Høyskoleforlaget, 1997, p. 283.

[17] H. Christiansen, *Norske dialekter*. Tanum-Norli, 1976.

[18] A. Adami, "Automatic Speech Recognition: From the Beginning to the Portuguese Language," Apr. 2022.

[19] X. Huang *et al.*, "Spoken Language Processing: A Guide to Theory, Algorithm, and System Development," Jan. 2001.

[20] I. J. Goodfellow *et al.*, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.

[21] S. F. Chen and J. Goodman, "An empirical study of smoothing techniques for language modeling," *Computer Speech & Language*, vol. 13, no. 4, pp. 359–394, Oct. 1999.

[22] A. Graves, *Generating Sequences With Recurrent Neural Networks*, 2014. arXiv: `1308.0850 [cs.NE]`.

[23] F. Informatik *et al.*, "Gradient Flow in Recurrent Nets: the Difficulty of Learning Long-Term Dependencies," *A Field Guide to Dynamical Recurrent Neural Networks*, Mar. 2003.

[24] A. Graves *et al.*, "Unconstrained On-line Handwriting Recognition with Recurrent Neural Networks.," vol. 20, Jan. 2007.

[25] A. Graves *et al.*, "A Novel Connectionist System for Unconstrained Handwriting Recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 5, pp. 855–868, 2009.

[26] A. Graves and J. Schmidhuber, "Framewise Phoneme Classification with Bidirectional LSTM Networks," in *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, vol. 4, 2005, pp. 2047–2052.

[27] P. Baldi *et al.*, "Exploiting the Past and the Future in Protein Secondary Structure Prediction," *Bioinformatics (Oxford, England)*, vol. 15, pp. 937–46, Dec. 1999.

[28] K. Cho *et al.*, "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation," Jun. 2014.

[29] A. Hidaka and T. Kurita, "Consecutive Dimensionality Reduction by Canonical Correlation Analysis for Visualization of Convolutional Neural Networks," vol. 2017, Dec. 2017, pp. 160–167.

[30] M. García-Ordás *et al.*, "Detecting Respiratory Pathologies Using Convolutional Neural Networks and Variational Autoencoders for Unbalancing Data," *Sensors*, vol. 20, Feb. 2020.

[31] G. Bailey, "Automatic Detection of Sociolinguistic Variation Using Forced Alignment," Dec. 2016.

[32] Lingit, NB Tale – Speech Database for Norwegian. The Language Bank: oai:nb.no:sbr-31, 2015.

Appendix A

Maps with dialectal phenomena

The following figures show maps with the distribution of some of the most important dialectal phenomena.

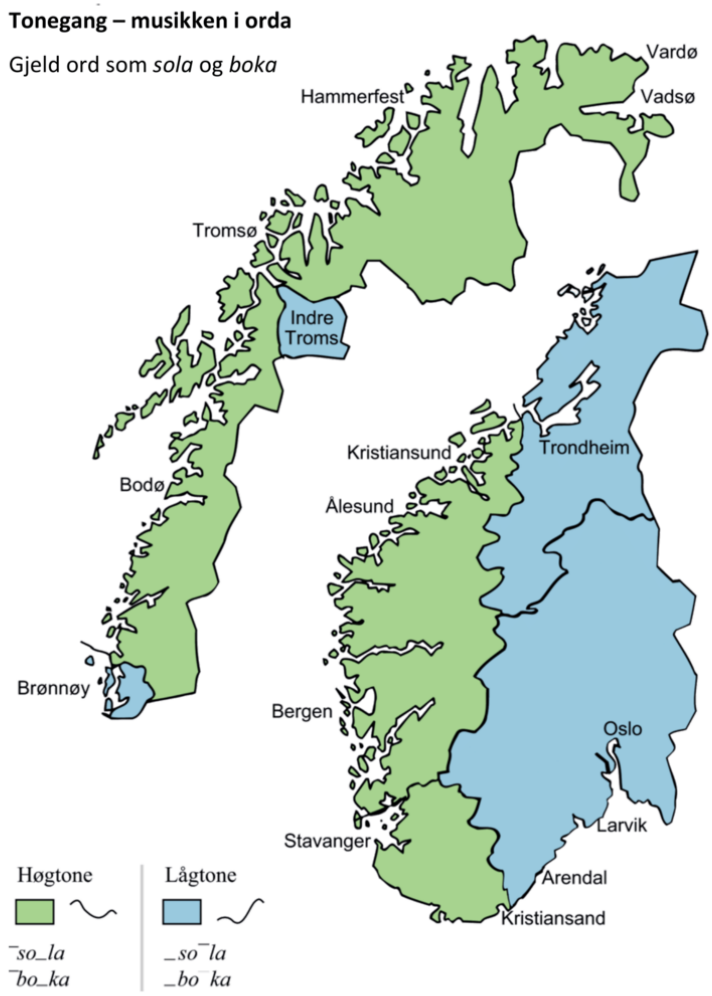


Figure A.1: High-pitch and low-pitch dialects [14].

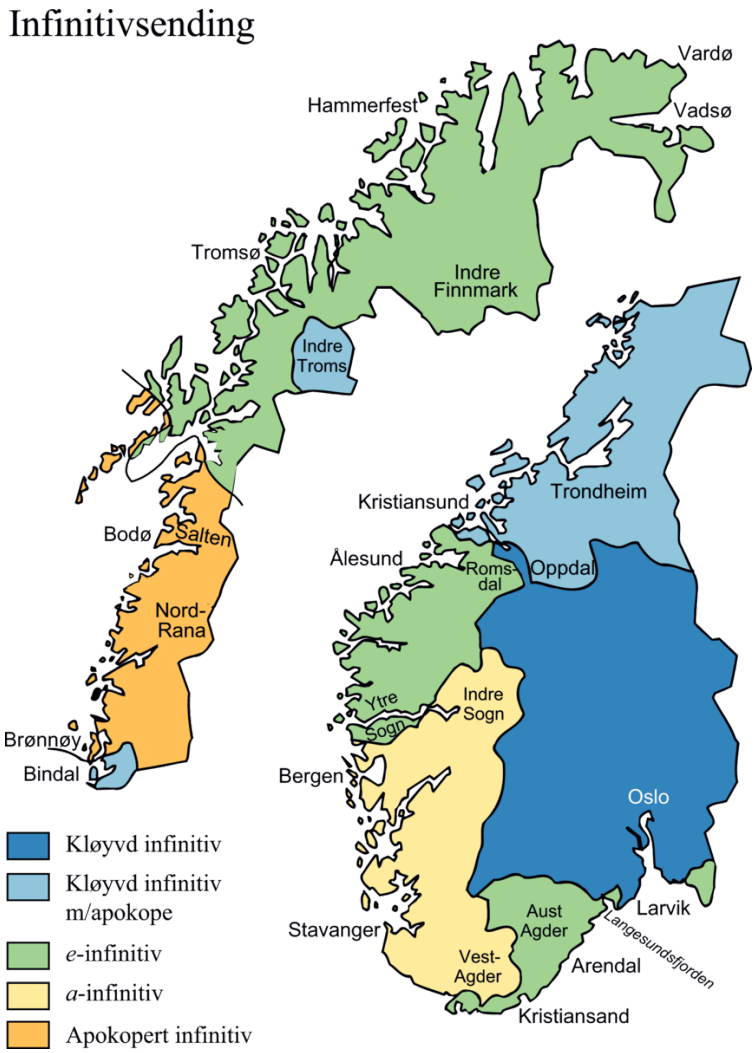


Figure A.2: Conjugation of infinitive verbs [14].

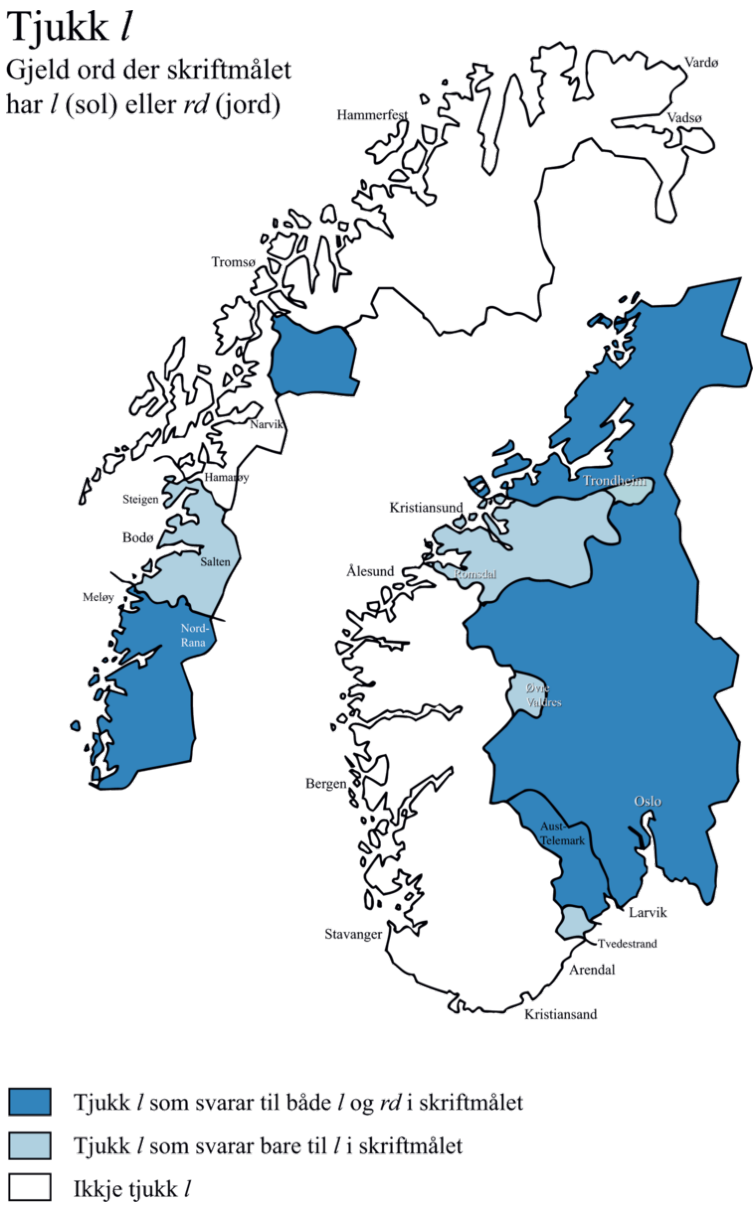


Figure A.3: Thick L [16].

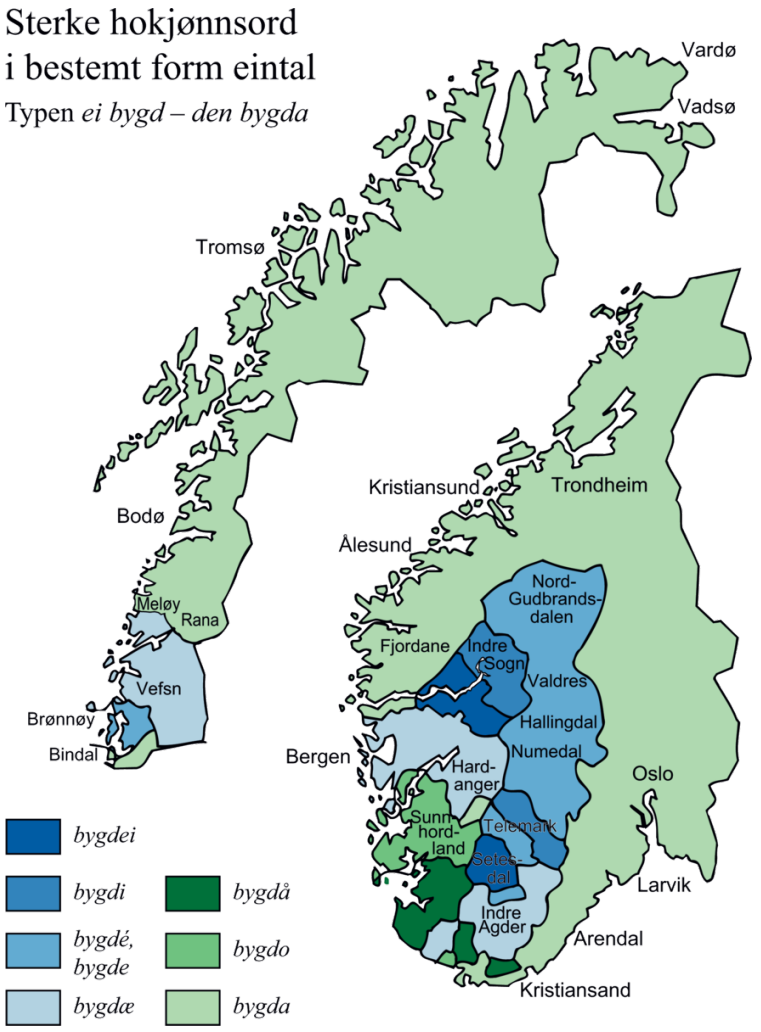


Figure A.4: Conjugation of “strong” feminine nouns [14].

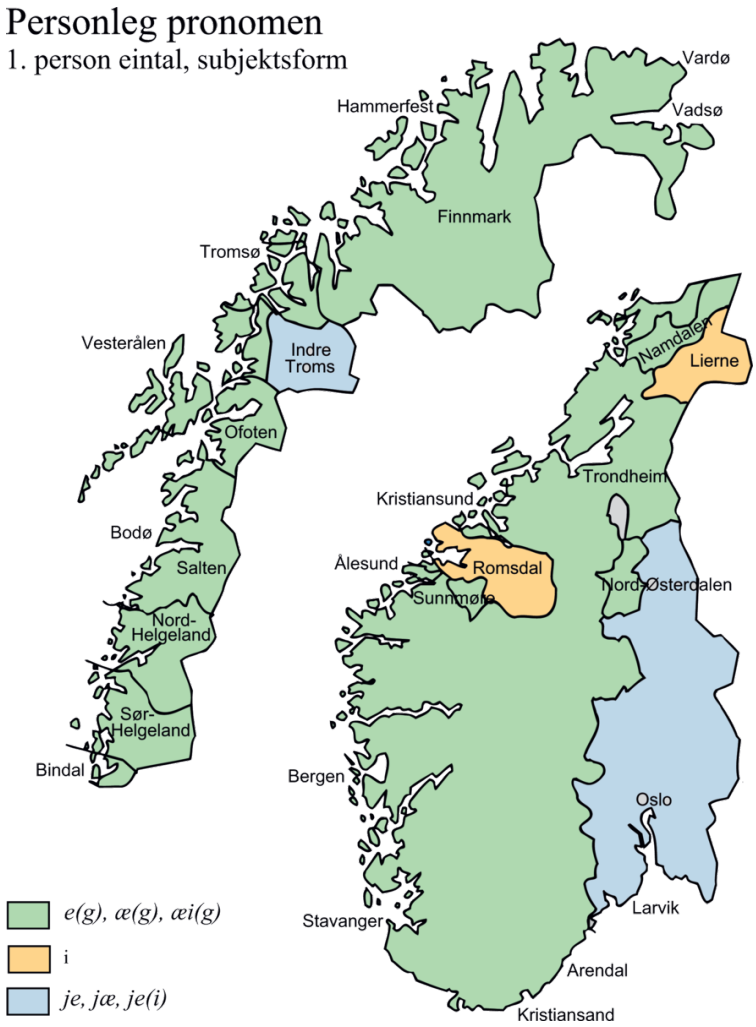


Figure A.5: Personal pronouns singular [14].

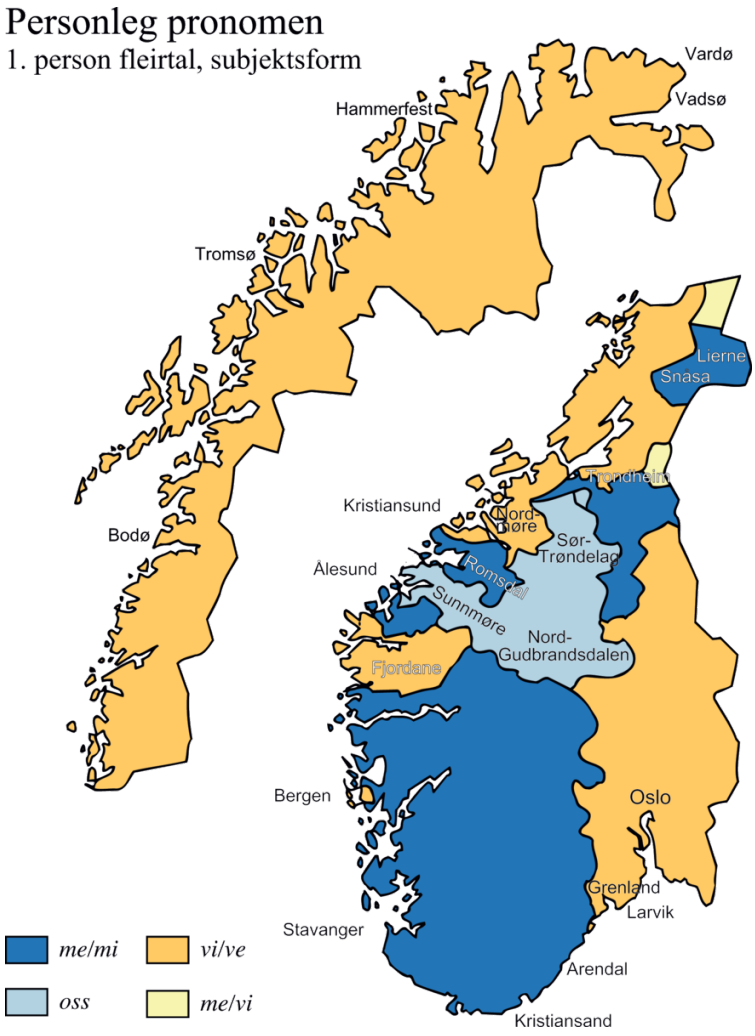


Figure A.6: Personal pronouns plural [14].

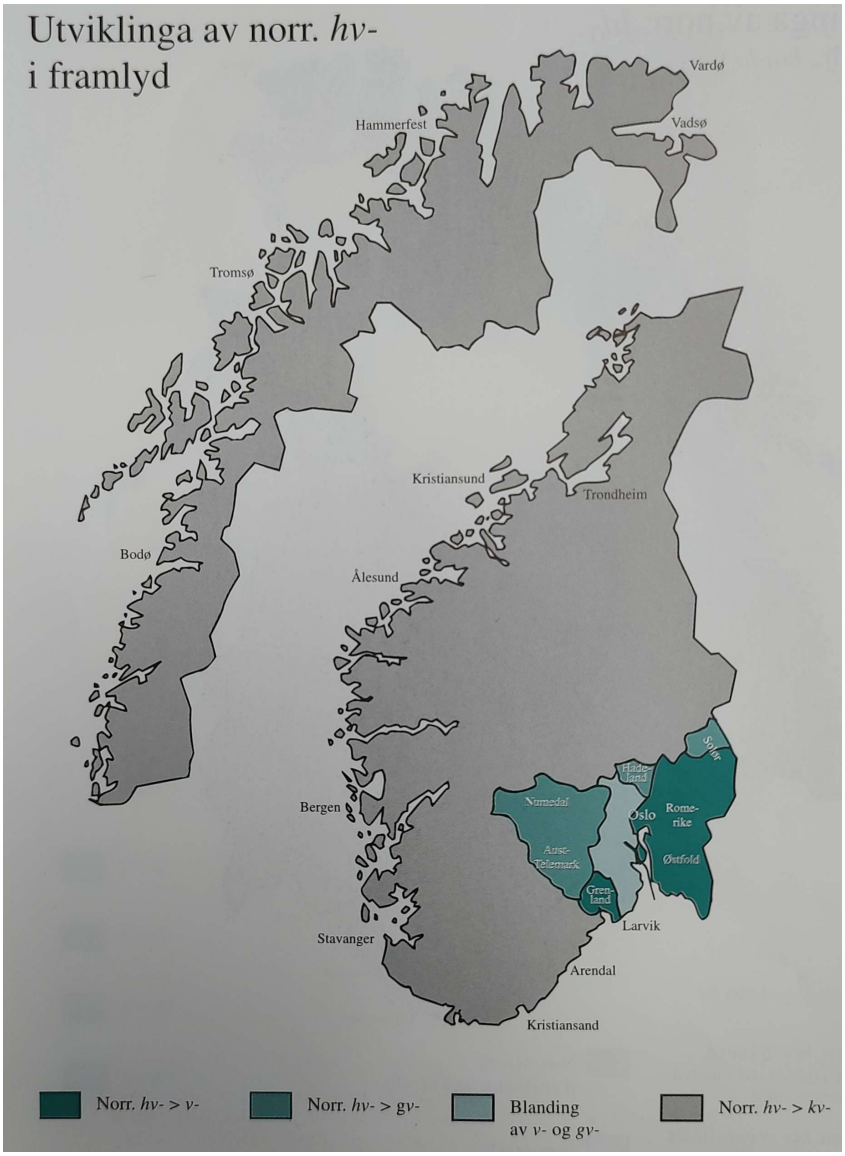


Figure A.7: Evolution of the hv-sound [14].

Appendix B

Results from forced alignment

An example of the resulting text-to-audio alignment using forced alignment is given in Figure B.1. The upper transcription in the figure shows the resulting alignment. The bottom transcription shows the manually created phonetic alignment, and the middle one shows the orthographic alignment based on the phonetic alignment. Figure B.1 shows that resulting alignment is poor. The model outputs big spaces between successive words, and the duration of each grapheme is usually shorter than the true duration.

We have some suggestions for why the alignment is poor. As introduced in Section 3.8.4 and Section 3.8.5, CTC disregards repetition of graphemes and thus frame-level alignment, which can be problematic when applying forced alignment. Based on our results, it seems like our CTC model does not care about the alignment of the graphemes to the audio, it only cares about outputting the correct transcription. Since the CTC is trained to reduce all predictions to only one character, we assume that this makes the frame-level alignment a difficult task. As mentioned in Section 3.8.5, forced alignment does not yet rival manual alignment, so we could not expect a perfect alignment. Anyways, our results are too bad to conduct a valid grapheme classification. Since the alignment of the graphemes often are wrong, using these alignments as training data would not give a consistent relationship between graphemes and phonemes. Thus, it would not make sense to use this data for training a supervised classifier to output graphemes based on phonetic input. We concluded that the method was insufficient for our experiments. Anyways, we find it interesting to work with improving forced alignment for CTC models, and encourage future work to look more into this.

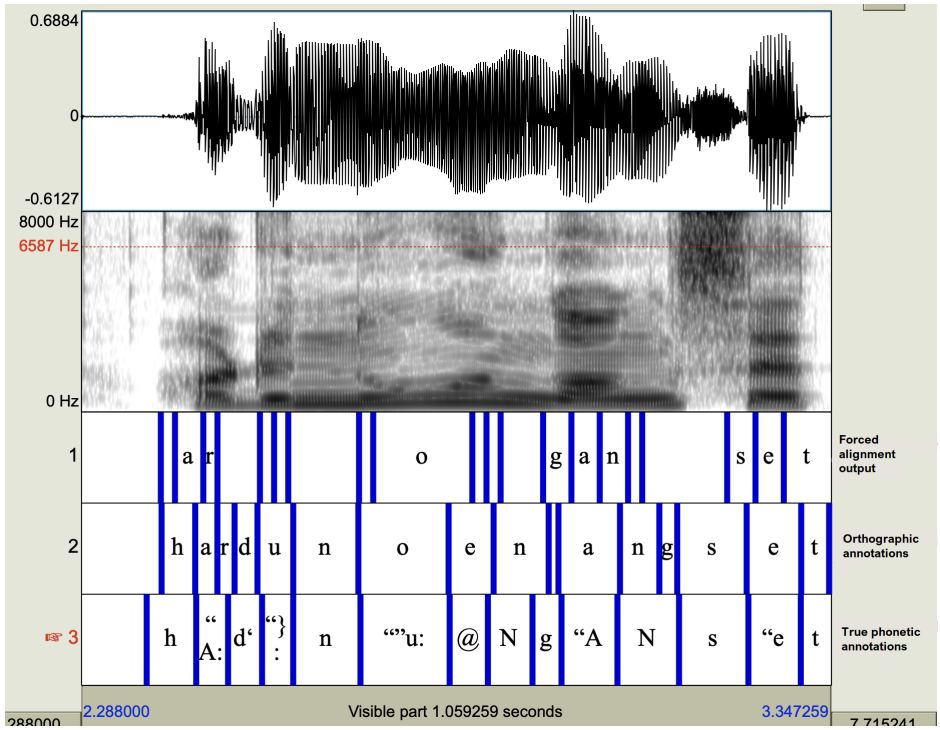
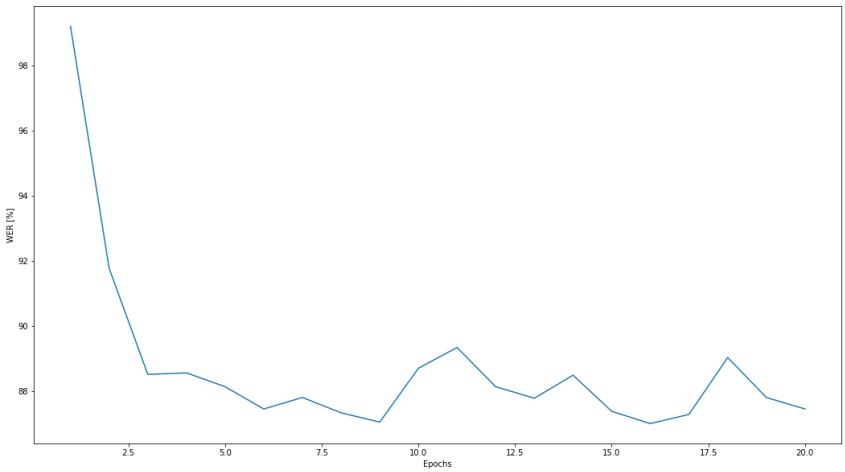


Figure B.1: An example of the resulting alignment from forced alignment. The upper transcription shows the predicted alignment. The bottom transcription shows the manually created phonetic alignment, and the middle one shows the orthographic alignment based on the phonetic alignment.

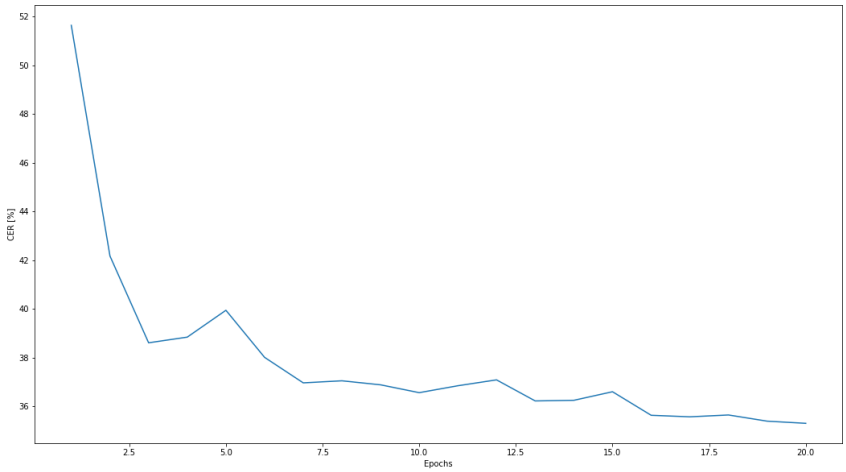
Appendix C

Graphs from the grapheme recognition task

The resulting WER and CER from testing each RNN layer on NB Tale module 1 & 2 after each training epoch are given in the following figures.

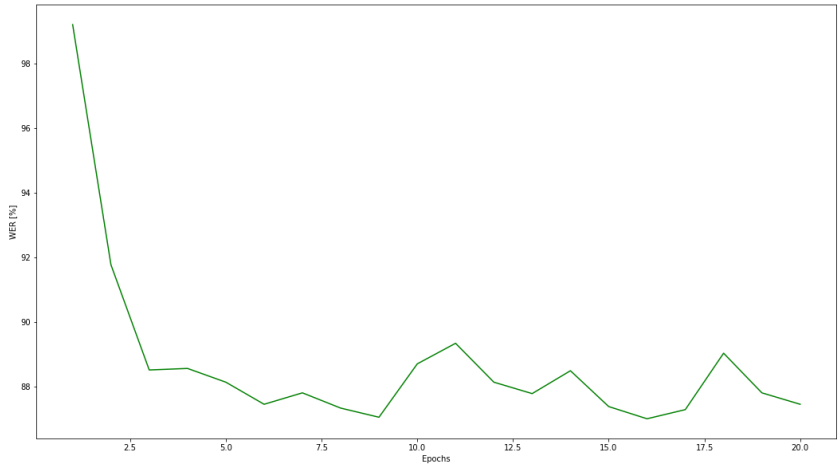


(a) WER.

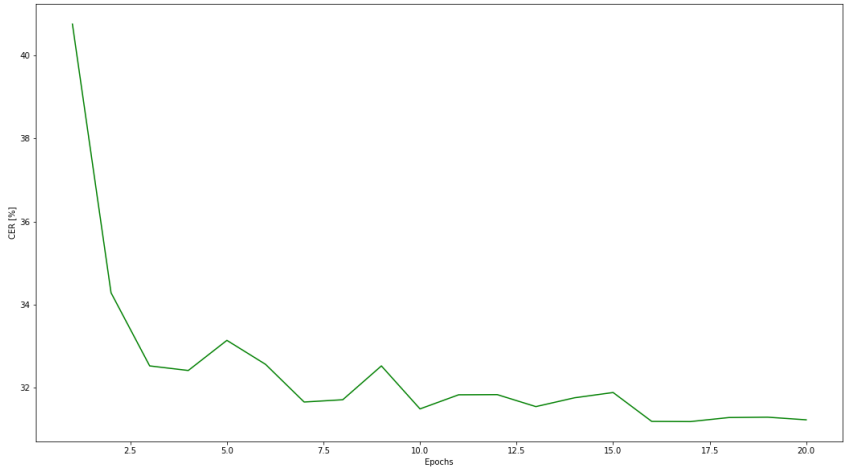


(b) CER.

Figure C.1: Graphs of test WER and CER for RNN layer 1.

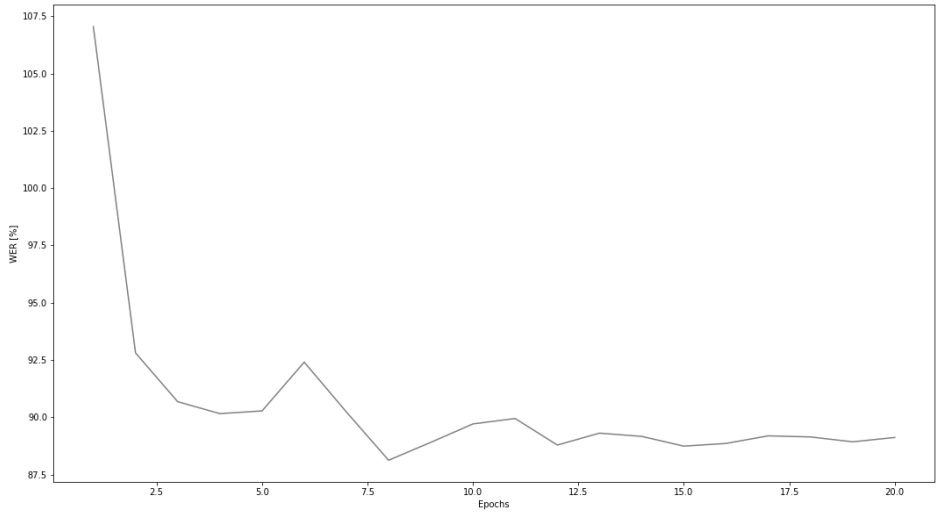


(a) WER.

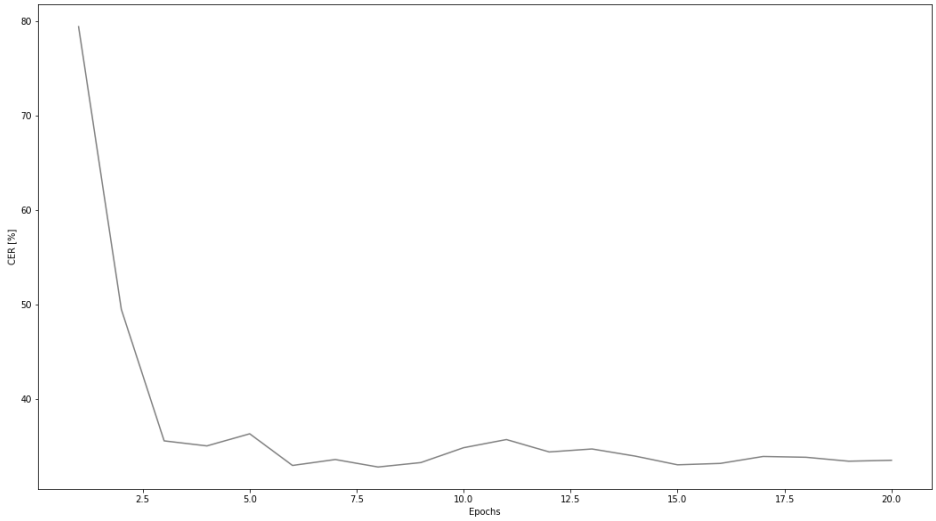


(b) CER.

Figure C.2: Graphs of test WER and CER for RNN layer 2.

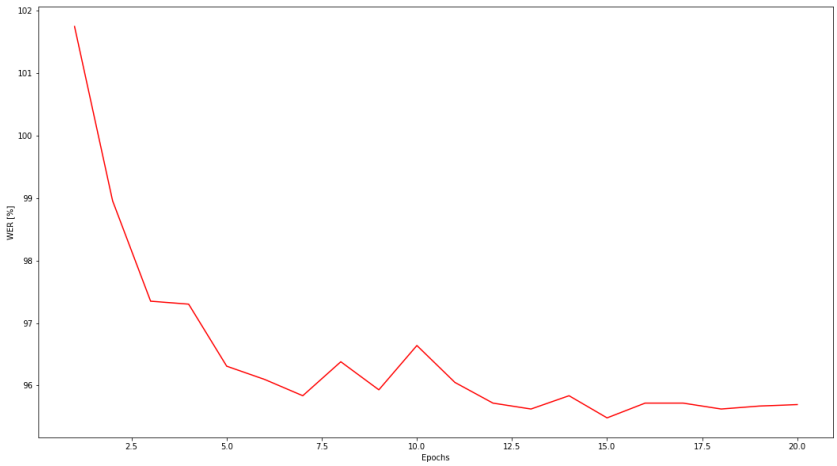


(a) WER.

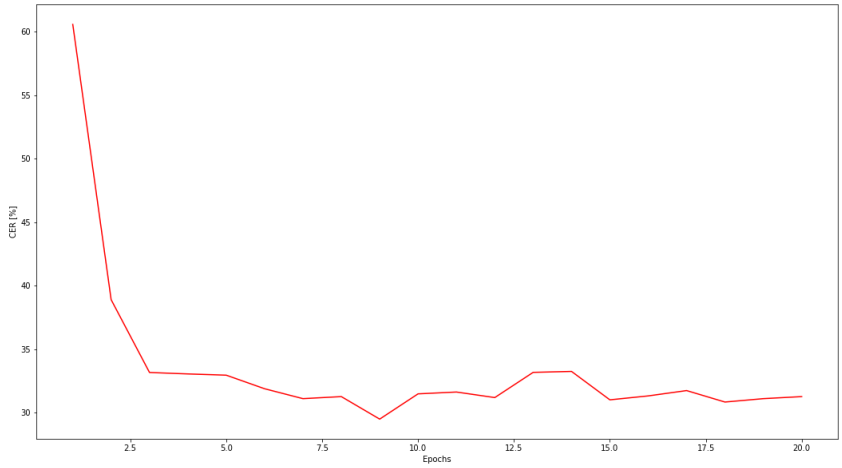


(b) CER.

Figure C.3: Graphs of test WER and CER for RNN layer 3.

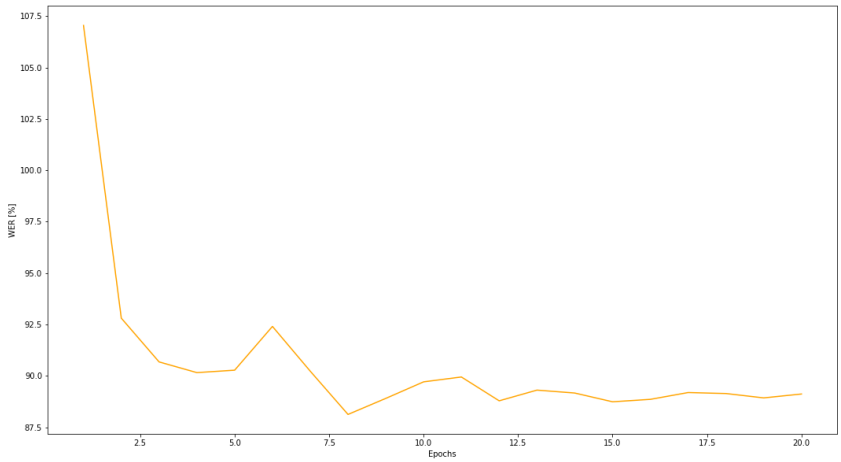


(a) WER.

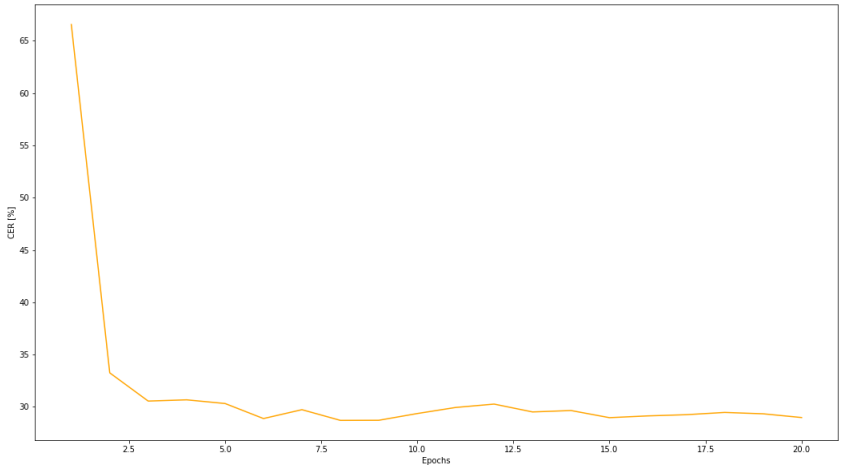


(b) CER.

Figure C.4: Graphs of test WER and CER for RNN layer 4.

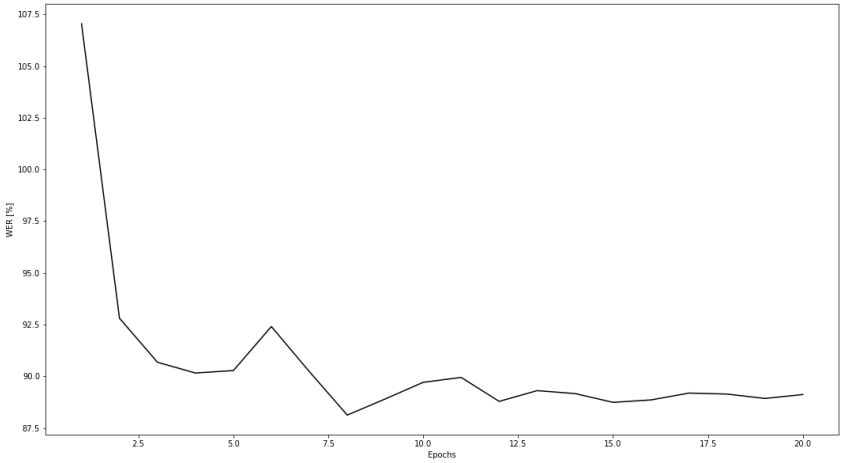


(a) WER.

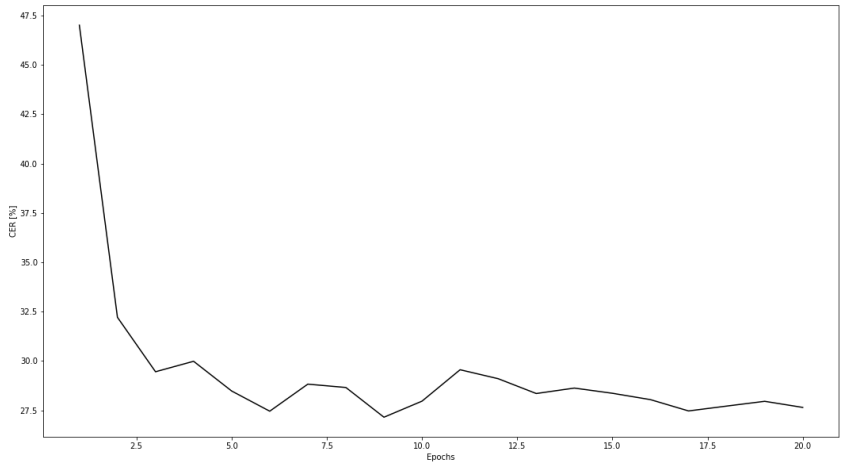


(b) CER.

Figure C.5: Graphs of test WER and CER for RNN layer 5.

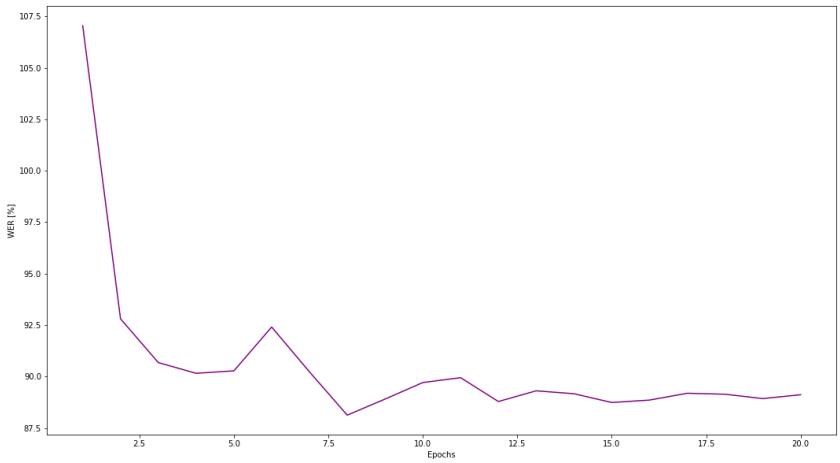


(a) WER.

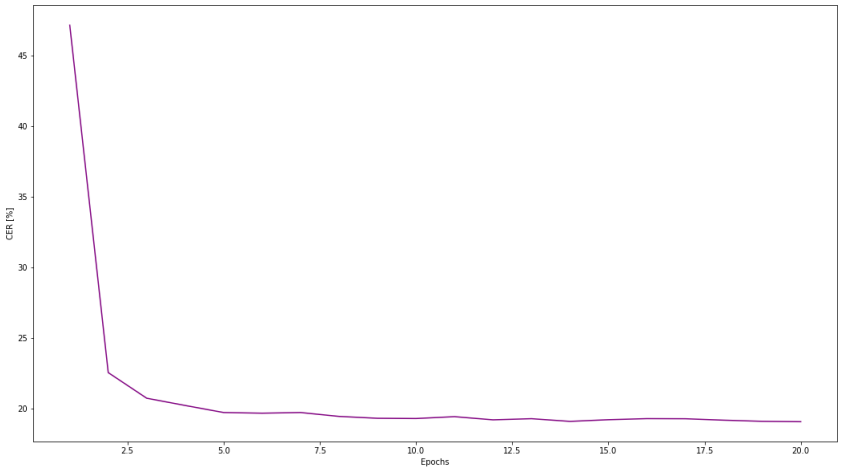


(b) CER.

Figure C.6: Graphs of test WER and CER for RNN layer 6.

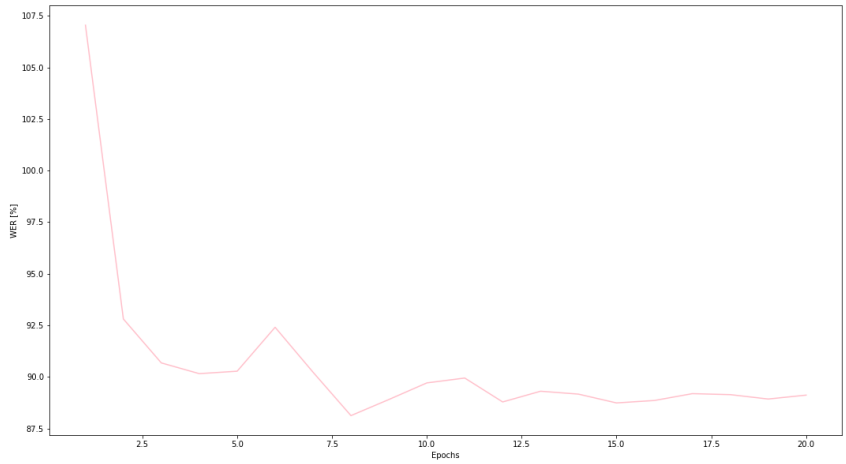


(a) WER.

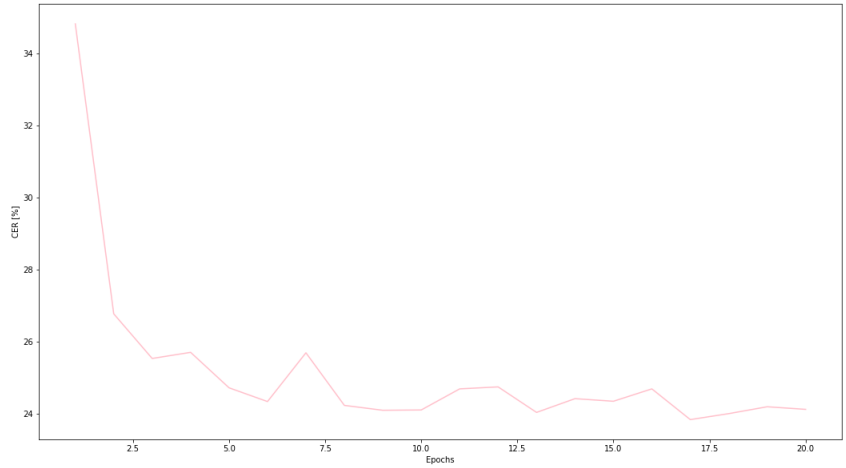


(b) CER.

Figure C.7: Graphs of test WER and CER for RNN layer 7.

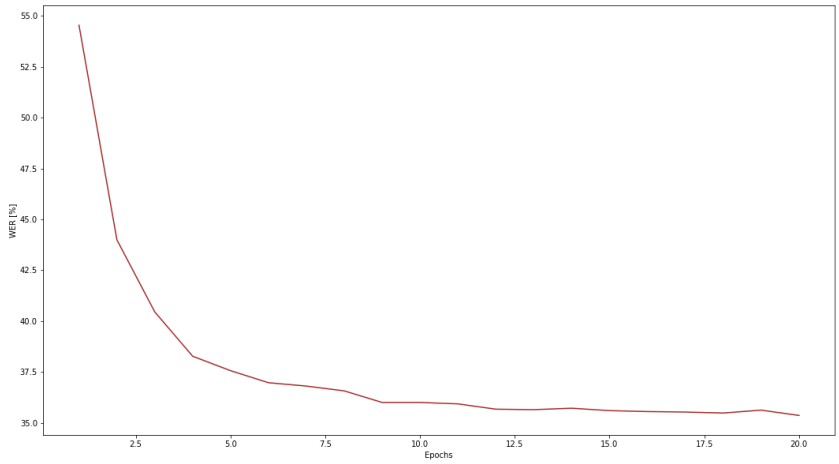


(a) WER.

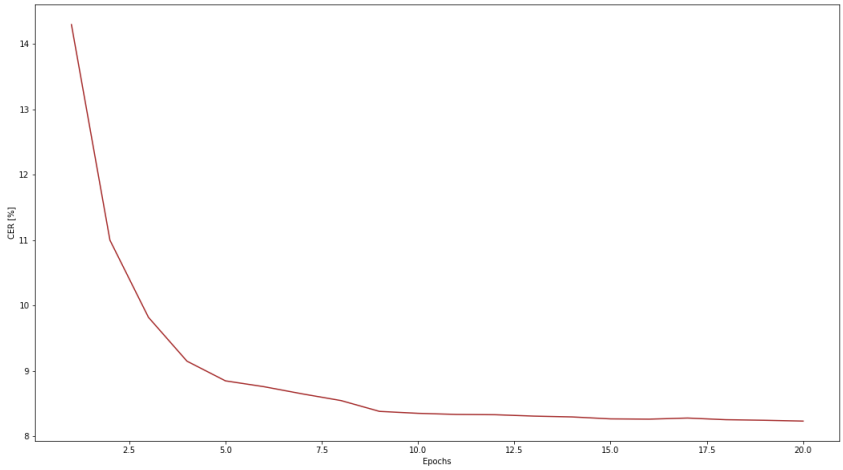


(b) CER.

Figure C.8: Graphs of test WER and CER for RNN layer 8.



(a) WER.



(b) CER.

Figure C.9: Graphs of test WER and CER for RNN layer 9.

Solveig Reppen Lunde

Modeling the Interpretability of an E2E ASR System

NTNU
Norwegian University of
Science and Technology

telenor