Master's thesis

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

Edvard Birkeland

Synchronisation of Speech and Text for the Norwegian Book Industry

Master's thesis in Electronic Systems Design
Supervisor: Torbjørn Karl Svendsen
June 2022

NTNU
Norwegian University of
Science and Technology

Edvard Birkeland

Synchronisation of Speech and Text for the Norwegian Book Industry

Master's thesis in Electronic Systems Design
Supervisor: Torbjørn Karl Svendsen
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

Abstract

A link between audiobooks and digital e-books makes numerous navigation options available, including navigation by searching in audiobooks and switching media platforms between audio and text seamlessly without needing to navigate manually. It will in addition enable solutions where audio and text are presented simultaneously.

The specialization project "Automatic Synchronization of Text and Speech in Audiobooks" concluded that available algorithms, such as the Montreal Forced Aligner, can provide good quality time-alignments of Norwegian Bokmål speech and text. This is as long as the speech and text segments are exact matches and within a length limit. This issue can be solved by dividing audiobooks into suitable segments. To automatically find fitting segments and make the entire synchronization process automatic, precise anchor points can be found by searching the audio for unique phrases. This technique is called keyword search.

The keyword search system is realised by extracting i-vector features from the audio, which are used to construct a lattice describing the most probable word sequences, in classical speech recognition fashion. Phrases are then spotted if they appear in the lattice with a probability above a set threshold. The decoding graph is constructed by combining knowledge of pronunciation, grammar, and acoustics of Norwegian Bokmål. This thesis investigates the proposed solution to discover if it is a viable strategy. The strategy is promising, but the results are unclear as the system is not managed to be assembled correctly.

Sammendrag

En kobling mellom lydbøker og digitale e-bøker vil gjøre flere navigeringsmuligheter mulig, blant annet navigasjon i e-bøker ved tekstsøk og muligheten til sømløs bytting mellom bokformatene hvor tjenesten automatisk husker siste leseposisjon uavhengig av mediet. Det vil dessuten være mulig å tilby løsninger hvor brukeren leser og lytter på en bok samtidig.

Spesialiseringsprosjektet «Automatisk synkronisering av tale og tekst i lydbøker» konkluderte med at tilgjengelige verktøy, som Montral Forced Aligner, kan generere tilstrekkelig nøyaktige koblinger mellom lydbøker og e-bøker for norsk bokmål. Dette fungerer så lenge segmentene som kobles sammen stemmer nøyaktig overens og ikke er for lange. Bøker kan deles inn i passende deler ved å bruke nøyaktige ankerpunkt. Ankerpunktene kan automatisk bestemmes ved å søke etter unike fraser i lydfilen, ved bruk av nøkkelordgjenkjenning.

Nøkkelordgjenkjenning er realisert ved å ekstrahere «i-vector» egenskaper fra lydfilene, som blir brukt til å konstruere «lattice»-strukturer som inneholder de mest sannsynlige ordsekvensene i klippet, på samme måte som ved vanlig talegjenkjenning. Nøkkelfraser er deretter gjenkjent dersom de blir funnet med en tilstrekkelig sannsynlighet blant alternativene. Dekodingsgrafen er konstruert ved å kombinere kunnskap om uttalelse, grammatikk og akustikk ved bokmål. Denne avhandlingen undersøker om en slik strategi er fordelaktig. Strategien er lovende, men resultatene er utydelige siden systemet ikke er satt sammen feilfritt.

Acknowledgements

This thesis marks the end of the two-year program Electronic Systems Design. I would like to thank everyone who have supported and helped me through this journey: friends, family, my girlfriend, and professors.

I would especially like to thank my theses supervisor, professor Torbjørn Svendsen, for frequent discussions and valuable guidance throughout the last semesters.

I will like to thank Lars-Erik and Tone from Bokbasen AS for attending weekly meetings and offering their advice and encouragement.

A special thanks goes to Pieter Uys from Saigen (Pty) Ltd for being available and helpful through mail, and for giving me insight into his keyword spotting algorithm and South African models.

Table of Contents

Acronyms

ASR Automatic Speech Recognition. 3, 8, 13, 15, 18, 19, 21, 22, 23, 24, 25, 30, 31, 32, 34

CMVN Cepstral Mean and Variance Normalisation. 5, 14

DFT Descrete Fourier Transform. 4

FFT Fast Fourier Transform. 4

FST Finite-State Transducer. 9, 10, 11, 12, 13, 14, 15, 16, 19, 30, 31, 34

HMM Hidden Markov Model. 6, 8, 9, 14, 15, 16, 21, 22, 23

JFA Joint Factor Analysis. 7

KWS Keyword Spotting. 2, 3, 13, 15, 18, 19, 20, 21, 24, 26, 27, 28, 30, 31, 32, 33, 34

MFA Montreal Forced Aligner. 1, 18, 34

MFCC Mel-frequency Cepstrum Coefficients. 3, 5, 6, 7, 13, 14, 16, 17, 21, 22, 30, 31, 32

RAM Random Access Memory. 30

TDNN Time Delayed Neural Network. 21, 23

VAD Voice Activity Detection. 14, 31

1 Introduction

Passing on knowledge through books and writings has been a defining factor of human development. Books have provided people with joy and understanding of advanced topics for decades, and have only become more and more available. Recently, with the digitisation of society, wast amounts of books have become accessible from anywhere in the world through smartphones and tablets. Audiobooks and e-books are popular formats that make combining daily chores with reading convenient.

Navigation within audiobooks is not always a straightforward exercise. Audiobooks are often provided as one long audio file lacking navigation tools to bring the listener to desired chapters, sections, or other points of interest. Creating a link between audiobooks and ordinary books can enable numerous navigation possibilities, as text is much easier to search than audio recordings. An accurate link will, in addition to make navigation within audiobooks easier, enable navigation between audio and text media. This allows seamless navigation between audiobooks and e-books, perfect for those who like to alternate between listening and reading. Live highlighting of text spoken, in "karaoke style", will be achievable and can give those having trouble reading audio assistance. It will also be able to assist people with listening issues by displaying belonging text to the speech uttered.

Bokbasen AS is an organisation that gathers metadata and digital books from over 1500 publishers and producers, and organises this data in order to distribute it from one place. Bokbasen provides both audio- and e-books in their online library service, Allbok. Bokbasen AS desires to create a link between audio- and e-books to provide innovative navigation options and reading experiences for their customers, and is thus the project description provider of this thesis.

The specialisation project "Automatic Synchronization of Text and Speech in Audiobooks" investigated the reliability of the open-source tool Montreal Forced Aligner (MFA) for alignments of Norwegian speech and text. The results are overall very good. Figure 1 displays a screenshot of a demonstration of live text highlighting in speech using results produced in the project. Some issues occurred when attempting to align long sequences. The main issue is that whole books cannot be aligned in one go because of the vast amount of available computer memory needed. The segments can not exceed a limit of about 40 minutes of speech and must consist of exact speech and text matches. Something to keep in mind when finding these segments is that the text and speech may differ. Books can, for example, contain images not described directly in the text that audiobook readers choose to describe.

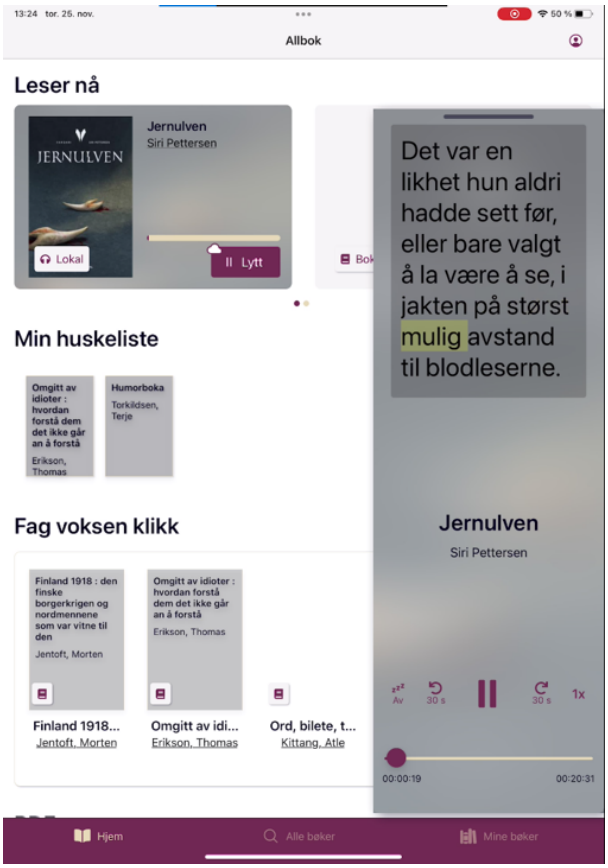


Figure 1: Demo displaying a joint listening and reading experience in the Allbok application. Taken from the specialisation project.

In order to solve these alignment issues automatically, precise anchor points have to be found throughout books. This thesis will research the possibility of utilising Keyword Spotting (KWS) to find these anchor points by searching and locating unique phrases in audio recordings. The spoken term detection system developed by Pieter Uys [1] is adapted to Norwegian Bokmål to develop this system.

This thesis starts by introducing relevant theory for the system development and evaluation of the KWS system in Section 2. The system used is then presented in Section 3, and the implementation procedure of this system is revealed in Section 4. Results are presented Section 5. The system choice, implementation process and results are discussed in Section 6, where a conclusion is found in Section 7. Work left for the future is discussed in Section 8.

2 Theory

This section explains the fundamental concepts of a keyword spotting system.

2.1 Keyword Spotting

Keyword Spotting (KWS) shares similarities with Automatic Speech Recognition (ASR) but has a different goal. KWS is applied to find preselected words or short phrases in continuous speech, whereas ASR aims to interpret an entire speech segment. KWS is commonly used in surveillance systems in order to spot statements of interest.

One approach to realise a KWS system is to make acoustic models of the words attempted to spot, to then compare them to a universal model of all other words [2]. This method is well suited for spotting a few keywords. Another approach is to do traditional ASR in order to search the output for chosen keywords.

A KWS system consists in any case of two main components: feature extraction and decoding. A decoding graph, which essentially is a model of the language, is used to determine spoken words in a recording by applying extracted features from the audio signal. A word string likelihood given array features, $P(\mathbf{W}|\mathbf{X})$, is modeled as

$$P(\mathbf{W}|\mathbf{X}) = P(\mathbf{W})P(\mathbf{X}|\mathbf{W}) \tag{1}$$

where $\mathbf{W}$ is a word string, $\mathbf{X}$ is a feature array, $P(\mathbf{W})$ is the language model, and $P(\mathbf{X}|\mathbf{W})$ the acoustic model, see Section 2.3 and Section 2.4. The most likely string is accepted as recognised words in ASR, while KWS-systems can search within rejected strings as well if desired.

When using a universal word model, the keyword estimation is compared to the sum of all other word estimations, see Equation 2. In this equation $\sum_{\mathbf{W}} P(\mathbf{W})P(\mathbf{X}|\mathbf{W})$ is substituted in place of P($\mathbf{X}$), which gives a ratio used to estimate confidence of keywords. Keywords within a confident threshold are accepted. The sum of all string probabilities is not used in practice, as it is sufficient to use a finite number of the highest likelihood estimations. [3, Chapter 9.7]

$$P(\mathbf{W}|\mathbf{X}) = \frac{P(\mathbf{W})P(\mathbf{X}|\mathbf{W})}{P(\mathbf{X})} = \frac{P(\mathbf{W})P(\mathbf{X}|\mathbf{W})}{\sum\limits_{\mathbf{W}} P(\mathbf{W})P(\mathbf{X}|\mathbf{W})} \tag{2}$$

2.2 Acoustic Features

Features are extracted to get hold of the relevant information within the sampled speech signal. There are numerous ways to extract useful information, where frequency-domain features generally provide much more accurate results than time-domain based features in speech recognition systems.

2.2.1 MFCC

Mel-frequency Cepstrum Coefficients (MFCC) have been the most common features for a long time. They provide useful information for speech analysis without being correlated, making them easier to model statistically. MFCCs are calculated following the steps shown in Figure 2.

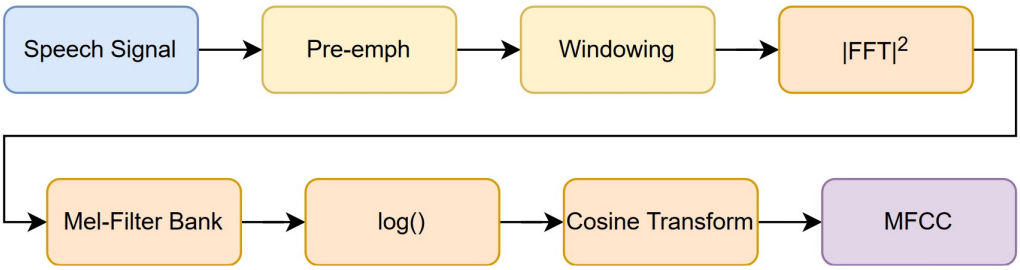


Figure 2: Overview of the MFCC extraction process.

Pre-emph

A common first step is to pass the sampled audio signal through a pre-emphasis filter. This filter amplifies high frequencies, as they usually have smaller magnitudes than low frequencies. This effect can also be achieved by mean normalisation of the raw cepstra at a later stage, which makes this step optional. [4]

Windowing

The sampled audio signal is split into overlapping, short frames in the windowing step. Typical values for these are 25 ms frames that are shifted 10 ms at the time. The reason for splitting the signal into shorter frames is that the frequencies through an entire audio signal will vary and thus not give an accurate frequency profile from a Fourier transform. We assume that the frequency is stationary within the short frames, meaning that there are no time-varying frequency components. This division of the signal assures that information about the varying frequency context throughout the audio signal is kept. The short audio frames are windowed by Hamming windows in order to reduce altering of the frequency domain information extracted from the Fourier transform step. This is because the Fourier transform assumes infinite signal length, introducing new, unwanted frequency components from the sharp edges of a rectangular window.

The Hamming window is a raised cosine window defined as

$$h(n) = \alpha + (1 - \alpha)\, cos\left[\frac{2\pi}{N} n\right] \tag{3}$$

where N is the filter length, and α is defined as 25/46. [5]

Fourier Transform

Following the windowing step, a Descrete Fourier Transform (DFT) of the signal is done. This is often done by using the Fast Fourier Transform (FFT) algorithm, which is an efficient implementation of DFT. The power spectrum is calculated by taking the square of the magnitude of the DFT for each frame, written out in Equation 4 and Equation 5.

$$X_i[k] = \sum_{n=0}^{N-1} x_i[k] e^{-j2\pi \frac{n}{N} k} \tag{4}$$

$$P_i[k] = \frac{1}{N} |X_i[k]|^2 \tag{5}$$

x_i is a frame of the windowed signal x, N is the length of the frame, X_i is the DFT of x_i, and P_i is the power of X_i. [3, Chapter 6]

Mel-Filter Bank

The next step is to calculate a Mel-Filter Bank, which is a uniform filter bank on the mel scale. The mel scale, short for melody, is a scale that describes our perception of sound, being less sensitive

to higher frequencies. The mel scale can be formulated as

$$mel(f) = 1127ln\left(1 + \frac{f}{700}\right)$$

Filter Banks are computed by applying overlapping triangular filters on the power spectrum. These triangular filters are placed out following the mel-scale, where the filters have their zeroes at their neighbours' filter banks peaks, shown in Figure 3. These triangular filters, defined as $H'(h)$, are calculated in Equation 6, where m are the filters, M is the number of filters ($m = 1, 2, .., M$), and $f(m)$ is the mel-filter centers in frequency scale. [3, Chapter 6]

$$H'_m(k) = \begin{cases} 0 & k < f(m-1) \\ \frac{k-f(m-1)}{f(m)-f(m-1)} & f(m-1) \leq k \leq f(m) \\ \frac{f(m+1)-k}{f(m+1)-f(m)} & f(m) \leq k \leq f(+1) \\ 0 & k > f(m+1) \end{cases} \tag{6}$$

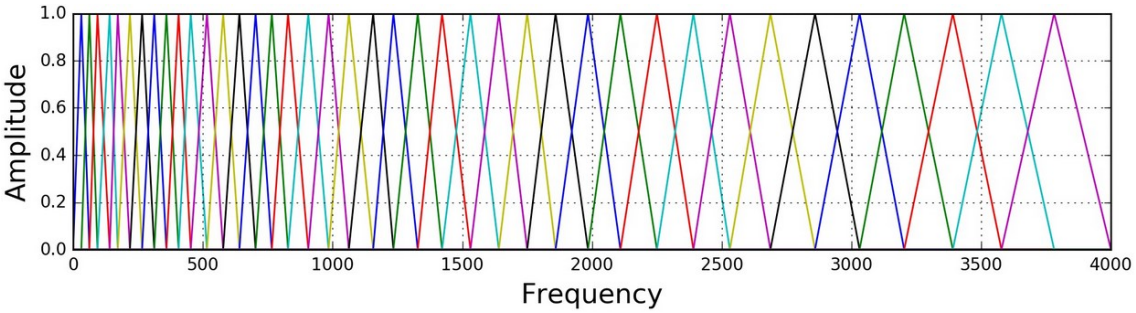


Figure 3: An example of a size 40 filter bank. [4]

Log - Cosine Transform

The log energy for each filter bank is then computed as described in Equation 7.

$$S[m] = ln\left[\sum_{k=0}^{N-1} |X_i[k]|^2 H'_m[k]\right], \qquad 0 \leq m < M \tag{7}$$

The mel-frequency cepstrums are then computed by a discrete cosine transform, see Equation 8. This step decorrelates the MFCCs, as opposed to filter bank coefficients. [3, Chapter 6]

$$C[n] = \sum_{m=0}^{M-1} S[m]cos\left(\frac{\pi n(m+1/2)}{M}\right), \qquad 0 \leq n < M \tag{8}$$

Cepstral Mean and Variance Normalisation (CMVN) is an additional step often done with the raw cepstrums. Normalised cepstrums, $C'[n]$, are computed as shown in Equation 9. CMVN is particularly helpful for making recognition more reliable in noisy environments, but it also appears to improve results in clean environments. [6]

$$C'[n] = \frac{C[n] - E\{C[n]\}}{var\{C[n]\}} \tag{9}$$

2.2.2 MFCC dynamic features

In order to capture temporal information with MFCCs, MFCC can be expanded to include delta-coefficients, which are approximations of the temporal derivative. These delta features work well

in systems built on Hidden Markov Models, as HMMs assume that frames are independent of one another.

The delta differences, Δc_n, are generally defined as

$$\Delta c_n = \frac{\sum_{k=1}^{N} k\,(c_{n+k} - c_{n-k})}{\sum_{k=1}^{N} k^2} \tag{10}$$

where N is the number of adjacent frames to each side used to calculate the deltas, and c_n is the features used to calculate differences. [7]

The second-order delta features are calculated the same way, using delta features instead of MFCC features in Equation 10. All these features are normally combined into one feature vector x_n, see Equation 11. The total number of neighbouring frames used to calculate one feature vector is then $4N + 1$; see Figure 4 for a visualisation of this when two neighbouring frames are used.

$$x_n = \begin{pmatrix} c_n \\ \Delta c_n \\ \Delta\Delta c_n \end{pmatrix} \tag{11}$$

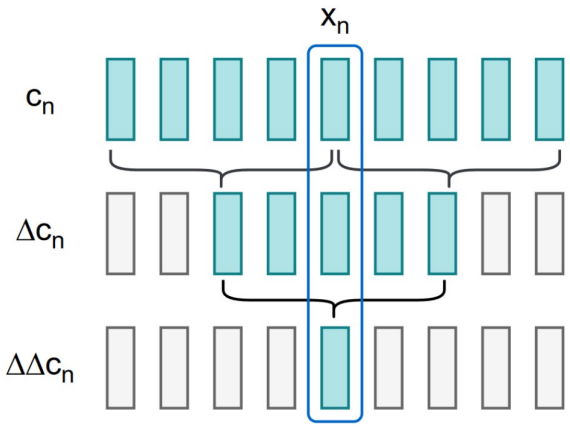


Figure 4: $\Delta\Delta$-feature extracted from $N = 2$ adjacent frames.

It is not common to use delta features of a higher order than two since they do not improve speech recognition after second order. [3, Chapter 9.3]

2.2.3 The Pitch

A significant amount of languages use pitch to distinguish between words. About 60 to 70 per cent of languages are tonal languages, meaning that they actively use pitch to disambiguate between words. Mainly African, East Asian and Southeast Asian languages are, to a large extent, highly tonal. However, most languages use pitch to some degree even though they are not classified as tonal. [8, Chapter 1]

Including pitch as a feature can significantly improve the speech recognition performance for tonal languages. However, even non-tonal languages can use the pitch to improve their performance to some degree. Some pitch-based features that can be used for speech recognition are the pitch and the delta-log-pitch. The delta feature can be calculated from the un-normalised log pitch, using N neighbouring previous and future frames. These pitch based features can be added to the MFCC feature vector to improve the performance. [9]

2.2.4 I-Vectors

The i-vector approach, or total variability approach, is a relatively newly suggested approach initially developed to negate the environmental disturbances for speaker recognition systems. The i-vector model is built on the Joint Factor Analysis (JFA) model. Compared to JFA, i-vectors are significantly reduced in dimension, usually having a dimension in the hundreds.

JFA decomposes an ideal speaker supervector s into components consisting of speaker-independent m, speaker-dependent Vy, channel dependent Ux and residuals speaker dependent Dz elements, see Equation 12. V, U, and D are eigenvoice matrices, while y, x, and z are their corresponding eigenvoice factors. The matrices are trained one at a time so that the factors can be computed from them.

$$s = m + Vy + Ux + Dz \tag{12}$$

Contrary to JFA, i-vectors model all relevant variability into one low-dimension subspace. For i-vectors, the environment and speaker-dependent supervector is modelled as

$$s = m + Tw \tag{13}$$

where m is the universal background model supervector. T is a low-dimensional total variability matrix where relevant variability is modelled, and w is the i-vectors, which are random vectors that describe the variability not portrayed by T. [10]

The total variability matrix, T, is often referred to as the i-vector extractor. The extractor is trained similarly to the V, the speaker eigenvoice matrix for the JFA case. The procedure is described in [11]. The difference is that when training T, all utterances from the same speaker are labelled as having different speakers. [12]

The supervectors s are essentially obtained by adapting features, such as MFCCs, to a universal background model using maximum-a-posteriori adaptation. From this, the i-vectors can be found. [13]

2.3 Language Modeling

Language models describe the grammar of languages, which fundamentally is a description of the order words typically occur. This can be interpreted as a probability distribution $P(\mathbf{W})$ where $\mathbf{W}$ is a sequence of words. For example, in a Norwegian model, some probabilities could be P("og") = 0.02 and P("tårnet avis Barcelona klump") = 0. This would mean that the word "og" is said on average once every fifty sentences and that the odd word string is practically never spoken.

The probability distribution can be expressed as

$$\begin{aligned} P(\mathbf{W}) &= P(w_1, w_2, ..., w_N) \\ &= P(w_1)P(w_2|w_1)...P(w_n|w_1, w_2, ..., w_{N-1}) \\ &= \prod_{i=1}^{N} P(w_i|w_1, w_2, w_{N-1}) \end{aligned} \tag{14}$$

where N is the number of words presented. This means that w_N is dependent on every previous word in the sequence presented.

In practice, this is impossible to do as when the sequences, often a sentence, get long, the histories of the sequences get unique or very rare. Instead, only a small number of previous words are typically used. A method for this is the n-gram model. In this model, n represents the number of words used to calculate the probability distributions, often referred to as the order of the model.

This leads to the distribution $P(w_i|w_{i-N+1}, ..., w_{i-1})$ for any n. A tri-gram model will for instance depend on two previous words $P(w_i|w_{i-2}, w_{i-1})$, while a bi-gram only one $P(w_i|w_{i-1})$, and a uni-gram only the word itself $P(w_i)$.

When generating an n-gram model, a corpus containing millions of words is usually used. The probabilities are determined by simply observing the frequency of unique word combinations. [3, Chapter 11.2]

A weakness of the n-gram model is that many combinations of words are given an extremely low probability even with relatively large corpora. For English n-gram models trained on corpora with a size of several million words, more than half of the detected tri-grams only occur once. If an utterance spoken is not included in the training data, the correct option will not be considered even if the acoustic model draws the correct conclusion. N-gram smoothing can be used to work around this issue. Smoothing makes it that any time $P(\mathbf{W}) = 0$, a low, non-zero probability is given instead. This assures that no sequences are regarded as impossible at the cost of losing some information from the training. [3, Chapter 11.4]

2.4 Acoustic Modeling

The most probable sequence of words $\hat{W}$ can, by using Bayes theorem, be estimated as

$$\hat{W} = \arg\max_W P(\mathbf{W}|\mathbf{X}) = \arg\max_W \frac{P(\mathbf{W})P(\mathbf{X}|\mathbf{W})}{P(\mathbf{X})} \tag{15}$$

where $\mathbf{X}$ is the observation or features, and $\mathbf{W}$ is a sequence of words. $P(\mathbf{W})$ is in this context the language model, and $P(\mathbf{X}|\mathbf{W})$ is the acoustic model. $P(\mathbf{X})$ can be disregarded, as it does not influence the estimation. This gives

$$\hat{W} = \arg\max_W P(\mathbf{W})P(\mathbf{X}|\mathbf{W}) \tag{16}$$

The most successful method for modelling acoustic models is the Hidden Markov Model (HMM). A HMM is briefly explained as a powerful statistical method that enables modelling probabilities of a series of unknown, not observable, events by observing a series of related data. The data observed are features in the speech recognition case.

An acoustic HMM consists of states describing possible phonetic units and transition probabilities between them. The acoustic HMM weights are determined by observing the frequency transitions between states occurring in a training set. In order to make an acoustic model that works for larger vocabularies, the words are decomposed into smaller units, see Section 2.6. [3, Chapter 8, 9]

2.5 Pronunciation Lexicon

Pronunciation lexicons are lexicons that include phonetic representations of each word in a language. Lexicons vary in size and should ideally include all words encountered in a recognition task.

Lexicons used in ASR-systems include extra words that represent silence and other non-word noises. These lexicons can include multiple pronunciation alternatives for words with corresponding probabilities. These pronunciation likelihoods sum up to one for every word. Speech recognition lexicon are often presented in the following format:

```
word    phone1 phone2 ... phoneN    probability
```

Pronunciation lexicons define phonetic combinations accepted in a model. This ensures that the recogniser only looks for existing words and not made-up ones. Pronunciation lexicons enable

decomposition of languages into corresponding phonemes, and in extension, other phonetic units. This makes it possible to make more general and less computationally demanding models, see Section 2.6.

2.6 Phonetic Modeling

There are a couple of ways to model the units used in speech recognition. The most precise unit that typically gives the best results is words. The advantage of using words over smaller units, such as phones, is that they capture the acoustic context within words that smaller units cannot. However, using whole words as units only works for very small vocabularies, as the size of storage and training data needed would be far too large. This strategy would fail for unknown words.

Phones are more commonly used. They are far fewer, with about 50 phones in English and Germanic languages, which makes them more trainable and general, and more suited for continuous speech recognition. The drawback with phones as units is that they tend to overgeneralise. These units do not consider that phones are usually pronounced differently in separate contexts.

In order to capture some context, units such as triphones and syllables can be used. Triphones are in this setting phonemes with added context from their neighbouring phonemes. These units are compromises between words and phones in that they capture more context than phones while being more general than words. Their numbers are still larger than phones by a fair margin; there are, for reference, over 30 000 different syllables in the English language. The number of these units makes systems built on them too complex for most continuous speech use cases. Another strategy to capture context is distinguishing phones by separating them by their position in words. Valuable context can be captured by reshaping each phone into four new phones depending on whether the phone occurs as a one-phone word or at the beginning, middle, or end of a word. The number of these context-dependent phones is more manageable than words and syllables. [3, Chapter 9.4]

2.7 Finite-State Transducers

Finite-State Transducers (FSTs) can provide a convenient representation of central components of speech recognition systems, and enables combination and optimisation of these. These main components include n-gram language models, pronunciation dictionaries, and acoustic HMMs. [14]

Finite-state transducers are finite automata that have labelled input and output state transitions, in addition to transition weights. An automaton is a system where input information is automatically transformed without outside interference. Automata with finite memory change state depending on inputs, where outputs depend on the current state. Transducers with the same input and output labels are called acceptors. [15, Chapter 3] See Figure 5 for a visualisation of an FST.

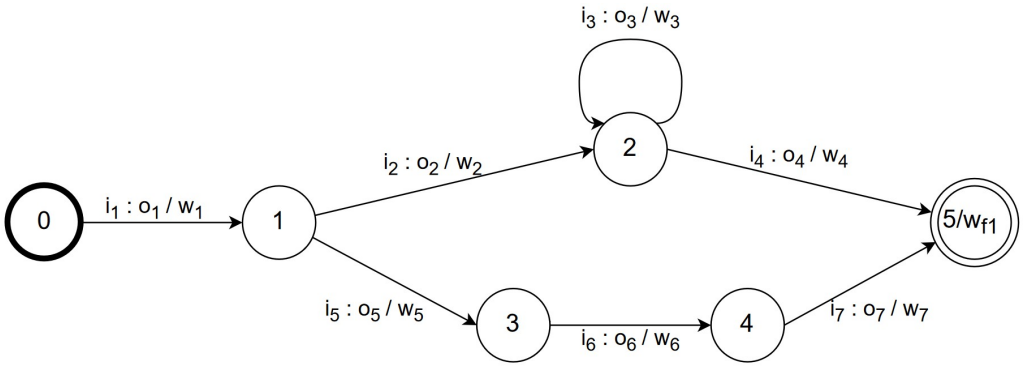


Figure 5: A finite-state transducer example. Each state is marked with a unique number. The initial state is represented as a bold cirkle and the final states are drawn as double circles. The transitions are marked with $i : o \ / \ w$, where i are input labels, o are output labels, and w are transition weights. Final states f are marked with their final weight w_f

FSTs are combined by composition, described in Section 2.7.1. FSTs can be optimised to reduce its complexity. This is useful as probabilistic models in speech recognition have to be complex to be accurate. FSTs can be optimized by determinisation, see Section 2.7.2, and minimization, see Section 2.7.2.

2.7.1 Composition

Composition is used to combine FSTs. The composition of two transducers, $a : b \ / \ w_1$ and $b : c \ / \ w_2$, is $a : c \ / \ w_1 \otimes w_2$. Weights are often presented as logarithmic values, making weight compositions the sum. See Figure 6 for an example of FST composition. The state numeration is rearranged, as it is important that states remain unique. Composition between transducers T_1 and T_2 is denoted as $T_1 \circ T_2$.

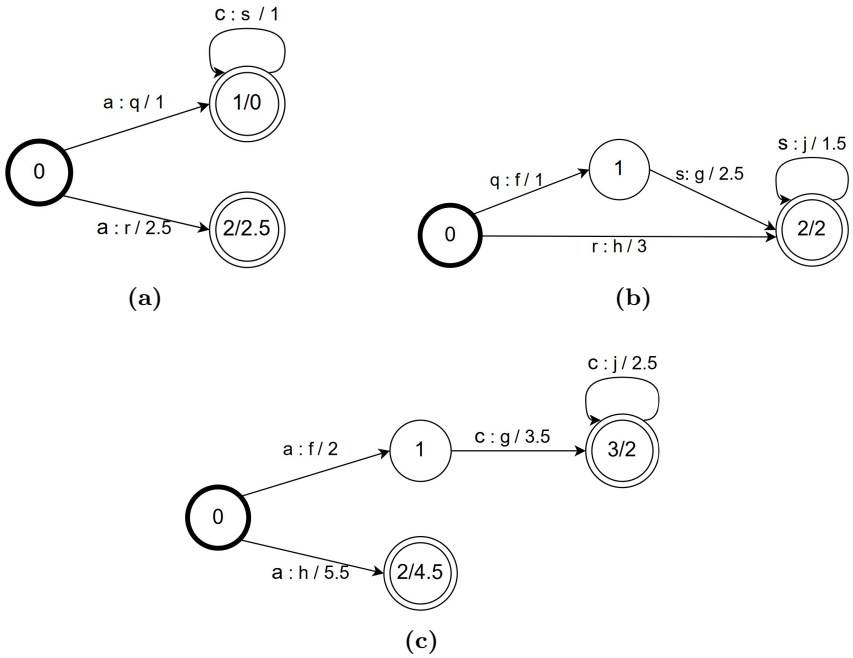


Figure 6: Example of FST composition. Composition of **a** and **b** results in **c**. [16]

2.7.2 Determinisation

Determinisation reduces the memory required and the duration it takes to process a string by combining redundant transitions. Determinisation ensures that every state has only one transition option for a given input label. Determinisation results in an equivalent FST, which means that the FST produces the same output string given identical inputs, with the same final weight. The structure of paths, distribution of output labels, and distribution of transition weights within the FST can however differ.

See Figure 7 for a visualisation of the determinisation process, where ϵ denote that input labels are not consumed during transitions, or that output labels produce no outputs.

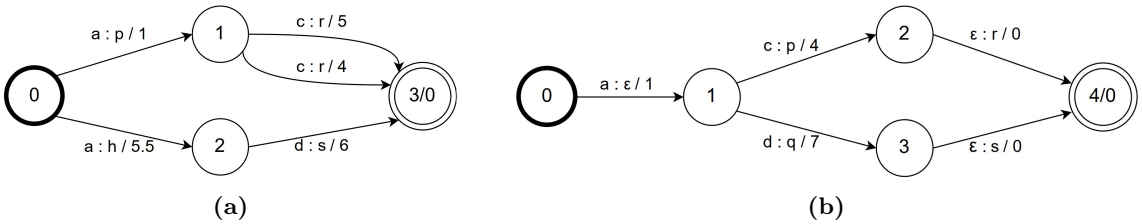


Figure 7: Example of FST determinisation. Determinisation of **a** results in **b**. [17]

2.7.3 Minimisation

Minimisation reduces the number of states and transitions in FSTs to the minimum numbers realisable while preserving deterministic properties. The resulting FST is equivalent to the original one. The minimality is achieved by enabling the option to have strings as output labels. See Figure 8 for an illustration of the minimization effect. This optimisation method further reduces time and required memory and is often combined with determinisation. [14]

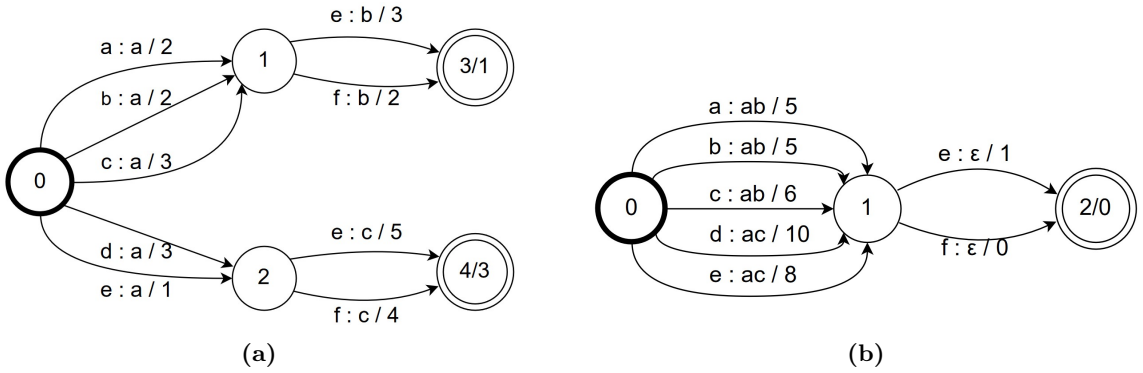


Figure 8: Example of FST minimisation. Minimisation of **a** results in **b**. The number of states are reduced from five to three, while the number of transitions are reduced from nine to seven. [18]

2.8 Lattices

Lattices are an efficient approach to presenting different decoding hypotheses. Lattices portray similarly likely strings in branches; see example in Figure 9. This is much more memory efficient than storing multiple full-length strings. [3, Chapter 13]

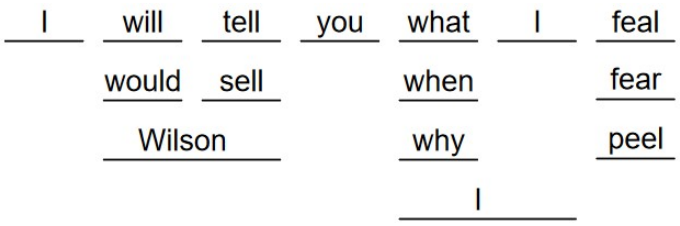


Figure 9: An example of a word lattice displaying likely utterance options. [3, Chapter 13]

Lattices can be constructed from decoding graphs represented as Finite-State Transducers. Scores from decoding are used to prune unlikely options compared to the most probable within a margin α. [19]

3 System Description

In this thesis, a spoken term detection framework developed by Pieter Uys, Saigen (Pty) Ltd, for South African languages is used as a basis for Norwegian detection, see [1]. The Kaldi Spoken Term Detection Wrapper is distributed under the Creative Commons Attribution 4.0 License, making it available for sharing and adaption, even for commercial purposes. This framework is chosen as it can spot phrases of multiple words, making it able to find unique sequences in a longer audio clip. It is constructed similarly to an ASR system in that it searches for keywords or phrases in a generated lattice. An overview of the central components of the spoken term detection system is presented in Figure 10.

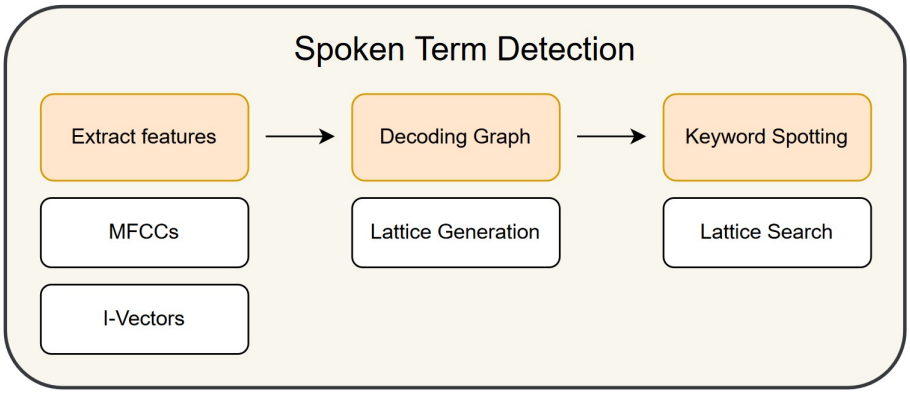


Figure 10: Overview of the spoken term detection process.

3.1 User Interface

The spoken term detector produces a keyword search result of the users' specified keywords and audio files, given that precise models of the desired language are provided. See Figure 11 for an outline of the interface.

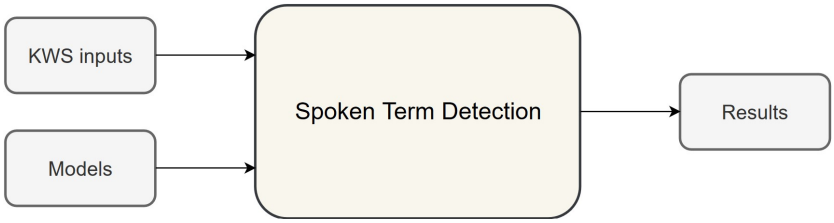


Figure 11: Overview of the spoken term detection interface.

The KWS inputs consist of a list of audio files and a list of keywords to be searched. These keywords can be either individual words or phrases. Optionally, correct transcripts of the speech utterances can be inputted. This is used to score the performance of the KWS-system. These transcripts can be provided with or without word timestamps, where transcripts with timestamps result in more accurate scoring.

The model inputs include a decoding graph, an i-vector extractor, and information about the phonetics of the language. The decoding graph is constructed as a FST which combines all information used to do KWS. The input model folder also includes configuration files that specify the MFCC extractions.

The produced results are a list of detected keywords and phrases with corresponding time labels, as well as an ASR result, which is the recognised transcript of the whole speech utterance. The spoken keyword detector also scores its performance if the user provides correct transcripts or alignments. The KWS performance is scored by the F4DE detection evaluation toolkit from NIST

[20]. Detected keywords are the desired output for this system; the other results are mainly for debugging. The user is also responsible for providing a working directory where all temporary files are created.

3.2 Feature Extraction

The first step of the process is extracting features. View Figure 12 for an overview of the process of extracting the final i-vector features used in the system.

At the start, thirteen-dimensional MFCCs are extracted from the audio with frameshifts of 0.01 seconds. This extraction step downsamples the audio to 8 kHz prior to extraction and does not use the energy, f0, as a feature. These MFCC features are used for Voice Activity Detection (VAD) in order to segment long audio clips into appropriate lengths.

Following, high-resolution MFCC features, with dimensions of 40, are extracted. Frame shifts of 0.01 seconds are used for these as well. These features are computed according to user-specified configurations, including sample and cut-off frequencies. CMVN statistics are computed and applied to all feature vectors. The high-resolution MFCCs are extracted in order to compute i-vectors using a pre-trained extractor. Three pitch features are extracted and can be added to the MFCC feature vectors if desired, see Section 2.2.3.

I-vectors are then computed using a pre-trained extractor. The extractor has to be compatible with the dimension and contents of the high-resolution MFCC features, meaning that it is trained on the same features types used in this step. The technique used is called online extraction, meaning that i-vectors are extracted every couple of frames instead of at the end of an utterance. I-vectors are extracted every ten frames in this specific process.

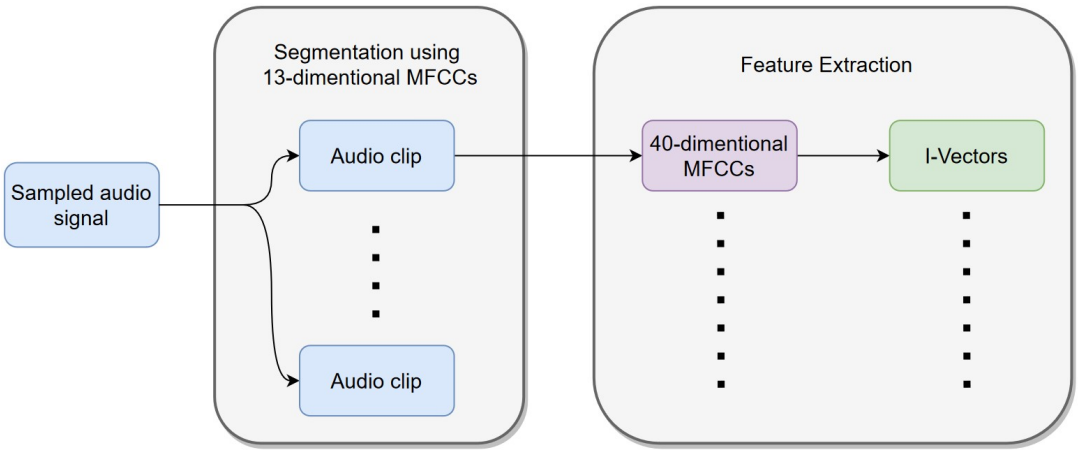


Figure 12: Overview of the feature extraction step.

3.3 Decoding and Spotting

Decoded lattices are constructed using i-vector features in combination with the decoding graph. The decoding graph is modelled as a FST. The decoding algorithm uses the acoustic HMM to scale the importance of the acoustic part of the FST. The acoustic scaling is set to 1.0, using the FST as it is. The lattice beam value is set to 8.5 and determines the lattices' depth. Increasing this value makes the lattices include less likely options and increases their size.

The lattices are then searched for selected keyword phrases. Any input key-phrase length work, but longer phrases are more error-prone and less likely to get recognised. Spotted keywords and phrases are provided alongside their time slot and confidence score.

4 Implementation

The system is realised by creating a decoding graph, which is provided alongside an i-vector extractor according to the description of the system. This completes a ASR system that is used as the basis for KWS.

4.1 Decoding Graph Creation

The decoding graph is made by combining all information available for decoding into Finite-State Transducers (FSTs), see Section 2.7. The information includes a pronunciation lexicon, language, and acoustic model. The complete graph is a FST named **HCLG**, as it is constructed of an acoustic **HMM**, the phonetic **Context**, a pronunciation **Lexicon**, and a **Grammar** part.

4.1.1 Pronunciation Lexicon

The pronunciation lexicon used contains nearly 90 000 words with adhering phones. This is a relatively compact lexicon with only one pronunciation option for each word. The pronunciation lexicon is listed in the following format, where the word consists of N phonemes:

```
word phone1 phone2 ... phoneN
```

This lexicon is used to generate a lexicon FST, L, which transduces phones to words. Each word gets assigned a unique numeric value that is used as a reference in the FST. The same goes for phonemes. Index zero is reserved for ϵ for both the word and phoneme list. New phonemes, called disambiguation symbols, are introduced in order to solve non-determinism. These disambiguation symbols are added as an extra phoneme at the end of words with similar pronunciations to differentiate between them. The phoneme list used contains 51 phonemes frequently used in the Norwegian language, where six extra disambiguation symbols are added. The original phonemes used are:

```
@ A A: Ai C N O O: Oy S Xl Xn ae ae: b d e e: ei f g h i i: j k l m n
oe oe: oei oev ou ou: p r rd rl rn rs rt s t u u: v xl xn y y:
```

A couple of silent words, such as pausing, hesitation, and coughing, are included in the lexicon. These words have no phonetic representation and have a chance to be reached in between regular words. The lexicon FST includes the option of entering silent states, linked to these silent words, at the end of words and start of sentences. The probability of entering silence is set to 0.5 in this implementation.

Self-loops are added to all states since words and phones usually are pronounced over an extended period. The FST is also sorted after output labels in order to make further processing easier. [21]

4.1.2 Language Model

The language model, or grammar, describes the probability of words and word sequences appearing in the language. The n-gram model describing the grammar is generated from a database consisting of mainly news texts, with roughly 355 million words. In this project, a tri-gram model is generated, which means that patterns of up to three consecutive words are taken into account, see Section 2.3. The tri-gram model only models combinations of words where all words are present in the vocabulary. The language model includes bi-grams and uni-grams in addition to the tri-grams.

In order to reduce the size of the grammar model and, in extension, the final FST, the grammar model can be pruned. Pruning means that less probable branches in the model are removed. The pruning value is the probability threshold determining which words and sequences are kept. Keeping all branches will generally give the most accurate model, as pruning will counteract the

effect of smoothing. Pruning is therefore a trade-off which is tuned in experiments. Probability thresholds of 10^{-8} and 10^{-7} are values used in testing.

The n-gram model is made using executables from Srilm, a language modelling toolkit that can be installed in Kaldi [22]. The n-gram model is then converted to FST format and named G for grammar. The grammar FST is mostly an acceptor. It accepts input symbols and applies weights according to the language model.

The grammar G and the pronunciation lexicon L are composed into a combined FST, LG. LG is then determinised and minimised in order to become less computationally demanding. The determinisation algorithm removes ϵ's besides its standard operation. The minimising algorithm used in this implementation differs from ordinary algorithms in that it does not push weights. Not pushing weights conserve stochasticity, meaning that weights throughout transitions, including final weights, sum up to one. The resulting FST can be expressed as $LG = min(det(L \circ G))$.

The combined FST is additionally weight pushed, ensuring that all weights sum up to the same value if they already are not. The FST is also sorted after input labels in order to speed up future compositions. The label sorting step is optional. [21]

4.1.3 Phonetic Context

Phonetic context is then introduced to the FST. See Section 2.6 for an explanation of why this is important. Every ordinary phoneme is transformed into one of four newly created phonemes depending on their position within words. The phoneme *ae* is for example substituted with *ae_B* if it is located at the start of a word, *ae_I* if it appears in the middle of a word, *ae_E* if it is ending a word, and *ae_S* if the phoneme stands alone.

The resulting FST after including context dependency is then $CLG = C \circ LG$. [21]

4.1.4 Acoustic Model

The acoustic HMM is transformed into an FST in order to be combined with the CLG. The H transducer is constructed as a FST that has the same state as its initial and final state, which makes the FST loop.

As in previous steps, the two FSTs H and CLG are combined and optimised. The FST is then $HCLG = min(det(H \circ CLG))$. All disambiguation symbols are then removed, as well as the ϵ's that are easily removable. At last self-loops are added to complete the FST. [21]

4.2 System Assembling

The necessary directories are created and provided to the spoken term detection system. This includes a folder containing audio clips and a keyword list, and a models folder. The models folder contain the decoding graph created in Section 4.1. A few versions of the decoding graph is made, where both pruned and full language model are used. This is done to make lightweight systems that can run on most computers. The model folder also contains an acoustic HMM, lists of indexed phonemes and words, and a pre-trained i-vector extractor. The HMM and i-vector extractor are both previously made, where the HMM is constructed using the same phone set as the one used in this implementation. The i-vector extractor is trained on high resolution MFCCs in the same format as the ones used in this system.

The system is originally developed for audio sampled at 8kHz. Audiobooks are generally sampled at much higher sampling frequencies, typically 44.1kHz. The system is adapted to work for 16kHz sampled audio, as the i-vector extractor is trained on high-resolution MFCCs extracted from a 16kHz sampled data set. The audio files are down-sampled to 16kHz prior to being organised in the input folder. The segmentation step of the system still uses 8kHz sample frequency. The audio signal used for segmentation is down-sampled from whatever frequency it is originally sampled at

within the system. Their respective configuration files set the sample frequencies for the thirteen and 40-dimensional MFCC extractions.

The original system built for South-African languages includes three pitch features in the 40-dimensional high-resolution feature vectors. Pitch is, however, excluded from the Norwegian implementation.

5 Results

Three KWS experiments are presented as results. The reference times used in experiments 1 and 2 are generated by force aligning the audio clips with their corresponding texts. The Montreal Forced Aligner (MFA) [23] is used to produce these time labeled alignments.

5.1 Experiment 1

Keyword Spotting is done on the following utterance, an excerpt from "Det Ellevte Manus" by Hanne Wilhelmsen.

> "Brilleglassene hennes var av en eller annen grunn gule og det gjorde det vanskelig å fange blikket hennes. Hun bar en ensfarget lysegrønn genser (...)"

The audio clip is nine seconds long and read by a woman. From this clip, the words "brilleglassene" and "ensfarget" are not included in the vocabulary. Tri-grams not included in the language model from this utterance are "annen grunn gule", "grunn gule og", and "å fange blikket", in addition to the ones containing unknown words. All possible bi-grams and uni-grams from the utterance are present in the grammar model.

5.1.1 Pruned Grammar Model

The following results are produced by a system where the n-gram model pruned with a cut off probability of 10^{-7}. The decoded ASR lattice is displayed in Figure 13, where the most likely utterance recognised is determined as:

> "drill gassen en sara hjemmel ram rund gullet og jo anke fang bytte hugg a m ens arg dis i en genser"

In this recognition, no tri-grams are present from the n-gram model. Seven bi-grams from the model are found in the utterance, they are: "gassen en", "gullet og", "og jo", "jo anke", "a m", "i en", and "en genser".

In this experiment, every single word of the audio clip is attempted to spot. The list of keywords attempted to spot is provided as follows:

```
brilleglassene
hennes
var
...
genser
```

The results of this KWS is presented in Table 1. In this table, reference time displays all correct word starts and endings, while system times are displayed only when the system recognises the keywords. The systems confidence scores for the keywords are also displayed, where 1.0 is the maximum score achievable.

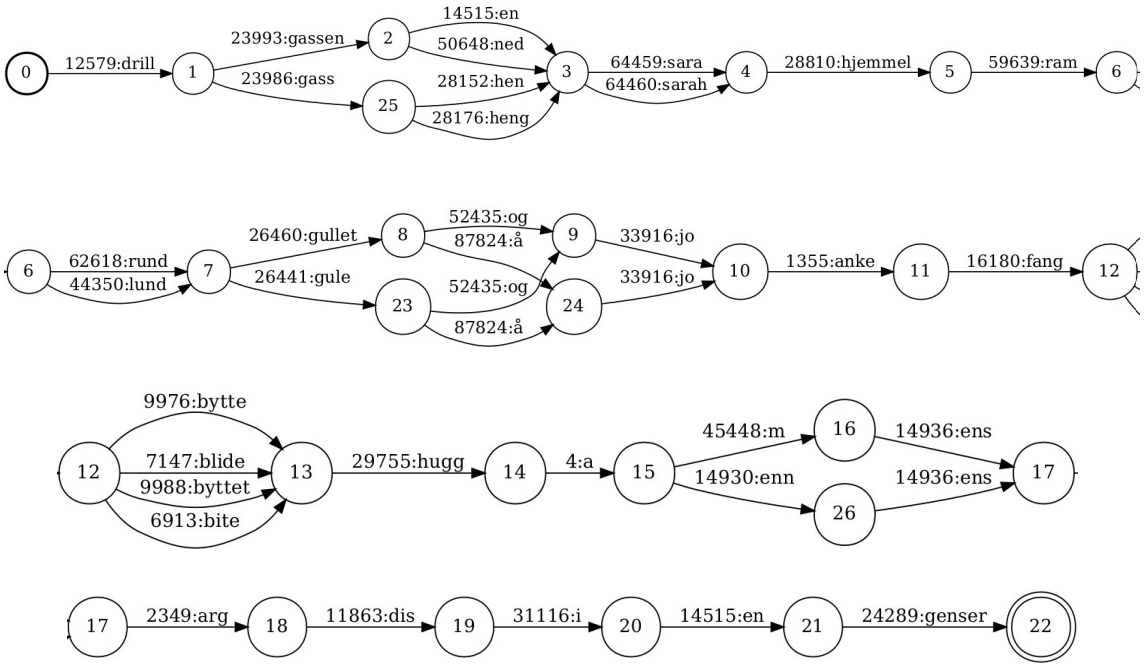


Figure 13: Lattice of the ASR result, used as basis for KWS. The result is not a FST, it is only presented as one. Weights, or "scores", are not included in this graphic. The numbers next to the words indicate their unique vocabulary indexes.

#	Word	Ref. start	Ref. end	Sys. start	Sys. end	Sys. score	Comment
1	brilleglassene	0.48	1.19				
2	hennes	1.19	1.50				
3	var	1.50	1.74				
4	av	1.74	1.91				
5	en	1.91	2.23	1.23	1.44	1.0	
6	eller	2.23	2.45				
7	annen	2.45	2.66				
8	grunn	2.66	2.96				
9	gule	2.96	3.41	2.97	3.48	0.004	”gullet” recognised as more likely
-	-						
10	og	3.79	3.87	3.69	3.93	1.0	
11	det	3.87	3.95				
12	gjorde	3.95	4.12				
13	det	4.12	4.15				
14	vanskelig	4.15	4.53				
15	å	4.53	4.56				
16	fange	4.56	4.84				
17	blikket	4.84	5.16				
18	hennes	5.16	5.52				
-	-						
19	hun	6.06	6.17				
20	bar	6.17	6.47				
21	en	6.47	6.67	7.77	7.95	1.0	
22	ensfarget	6.67	7.36				
23	lysegrønn	7.36	7.94				
24	genser	7.94	8.48	7.95	8.52	1.0	

Table 1: Word level KWS results. Timestamps are given in seconds.

5.1.2 Full Grammar Model

Full tri-gram recognition of the same utterance gives the results displayed in Table 2. Only recognised words are included in this table. The most probable word sequence from the lattice is:

> ”drill gassen en sara hjemmel ram rund gullet og jo anke fang bytte hugg a m ens arg dis i en genser”

#	Word	Ref. start	Ref. end	Sys. start	Sys. end	Sys. score	Comment
5	en	1.91	2.23	1.23	1.44	1	
9	gule	2.96	3.41	2.97	3.48	0.004	”gullet” recognised as more likely
10	og	3.79	3.87	3.69	3.93	1	
21	en	6.47	6.67	7.77	7.95	1	
24	genser	7.94	8.48	7.95	8.52	1	

Table 2: Recognised words from KWS. Timestamps are given in seconds.

5.1.3 Automatic Speech Recognition Results

Some simple ASR experiments are conducted using the same vocabulary and audio file. The words ”brilleglassene” and ”ensfarget” are still out of vocabulary. Table 3 and Table 4 display some ASR performances on the audio clip. The experiment in Table 3 is conducted with a plain HMM acoustic model, triphone phonetic units, and ordinary thirteen-dimensional MFCC features, while the test presented in Table 4 is done with hybrid HMM and TDNN model and i-vector features. The grammar model is a tri-gram model, where the pruning threshold is set to 10^{-8} in the experiments. The recognised words are labelled either Correct (C), Substitution (S), Insertion (I), or Deletion (D).

Referance		Pruned n-gram		Full grammar	
#	**Words**	**Recognised**	**Score**	**Recognised**	**Score**
		det	I	det	I
		lille	I	lille	I
1	brilleglassene	glasset	S	glasset	C
2	hennes	hennes	C	hennes	C
3	var	var	C	var	C
4	av	av	C	av	C
5	en	en	C	en	C
6	eller	eller	C	eller	C
7	annen	annen	C	annen	C
8	grunn	grunn	C	grunn	C
9	gule	gravide	S	gullet	S
10	og	og	C		D
11	det	det	C	all	S
12	gjorde	gjorde	C	jord	S
13	det	det	C	er	S
14	vanskelig	vansker	S	vanskelig	C
15	å		D	å	C
16	fange	fanget	S	fange	C
17	blikket	blikket	C	blikket	C
18	hennes	mens	S	hennes	C
19	hun	det	S	men	S
20	bar	bare	S	bare	S
21	en	en	C	en	C
		ens	I	ens	I
22	ensfarget	farget	S	farget	S
23	lysegrønn	lysegrønn	C	lysegrønn	C
24	genser	genser	C	genser	C
Total (C - S - I - D)		15 - 8 - 3 - 1		16 - 7 - 3 - 1	

Table 3: ASR results of a pruned and full grammar model, using acoustic HMM and ordinary MFCC features.

Referance		Pruned n-gram		Full grammar	
#	**Words**	**Recognised**	**Score**	**Recognised**	**Score**
		biller	I	briller	I
1	brilleglassene	glasset	S	glasset	S
2	hennes	hennes	C	hennes	C
3	var	var	C	var	C
4	av	av	C	av	C
5	en	en	C	en	C
6	eller	eller	C	eller	C
7	annen	annen	C	annen	C
8	grunn	grunn	C	grunn	C
9	gule	gullet	S	gullet	S
10	og	og	C	og	C
11	det	det	C	det	C
12	gjorde	gjorde	C	gjorde	C
13	det		D		D
14	vanskelig	vanskelig	C	vanskelig	C
15	å	å	C	å	C
16	fange	fange	C	fange	C
17	blikket	blikket	C	blikket	C
18	hennes	hans	S	hans	S
19	hun	hun	C	hun	C
20	bar	bare	S	bar	C
21	en	en	C	en	C
		ens	I	ens	I
22	ensfarget	farget	S	farget	S
		lise	I	lise	I
23	lysegrønn	grønn	S	grønn	S
24	genser	genser	C	genser	C
Total (C - S - I - D)		17 - 6 - 3 - 1		18 - 5 - 3 - 1	

Table 4: ASR results of a pruned and full grammar model, using hybrid HMM and TDNN model, and i-vector features.

5.2 Experiment 2

Keyword Spotting is done on a new utterance, another excerpt from "Det Ellevte Manus", with the same reader. The two words "forfattertype" and "skrøner" are out of vocabulary in this text. The audio clip is 15.3 seconds long and automatically segmented by the system into two clips covering 0 to 5.3 and 5.3 to 15.3 seconds. The phrase read in the audio clip is:

> "Hun lar seg på et vis ikke korrigere. Selv når folk skjønner at hun lyver. Katja er på mange måter den mest salgbare forfattertype som fins. Sjarmerende, selvopptatt, full av skrøner og uten særlig mange hemninger."

This audio clip contains three tri-grams not present in the language model, in addition to the combinations containing unknown words. These three unrecognised tri-grams are "vis ikke korrigere", "ikke korrigere ⟨s⟩", and "katja er på", where "⟨s⟩" denotes silence.

5.2.1 Pruned Grammar Model

The pruning threshold of the n-gram model is set to 10^{-7}. The decoded ASR lattice is displayed in Figure 14, where the most likely utterance recognised is:

> "unn a kjapp vis ikke korn gjerde selv og fag skjønn av tull iver katja a på mate dem mest salgbar fatt rips og fins skei mene selv å katt kulda skauen ordens alma henning er"

No tri-grams from the language model are present in this recognition.

Results of a word level KWS of all individual correct words are presented in Table 5 and Table 6. Two-word phrase search results in one correct hit: "vis ikke" in the period from 1.47 to 2.04 seconds into the clip.

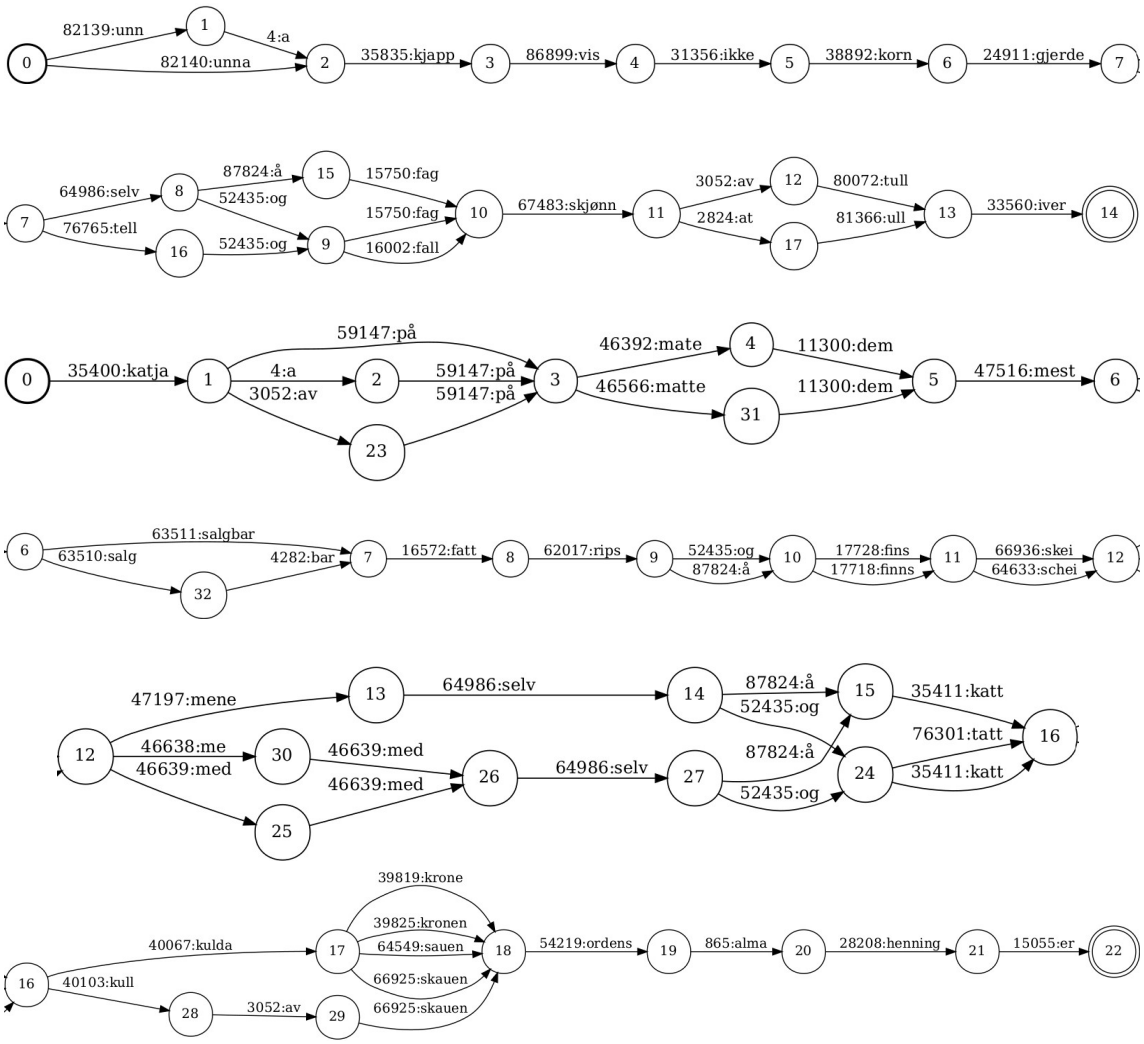


Figure 14: Lattice of ASR result. Result is decoded as two segments.

#	Word	Ref. start	Ref. end	Sys. start	Sys. end	Sys. score	Comment
1	hun	0.89	0.96				
2	lar	0.98	1.15				
3	seg	1.15	1.31				
4	på	1.31	1.42				
5	et	1.42	1.51				
6	vis	1.51	1.74	1.47	1.77	1.0	
7	ikke	1.74	2.02	1.77	2.04	1.0	
8	korrigere	2.02	2.63				
-	-						
9	selv	3.21	3.42	3.21	2.48	1.0	
10	når	3.42	3.53				
11	folk	3.53	3.84				
12	skjønner	3.84	4.18				
13	at	4.18	4.32	4.17	4.35	0.008	Decision: NO
14	hun	4.32	4.43				
15	lyver	4.43	4.80				
-	-						
16	katja	5.75	6.28	5.73	6.39	1.0	
17	er	6.28	6.39				
18	på	6.39	6.54	6.39	6.57	1.0	Detected twice
19	mange	6.54	6.77				
20	måter	6.77	7.22				
21	den	7.22	7.69				
22	mest	7.74	8.20	7.77	8.22	1.0	
23	salgbare	8.20	8.98				
24	forfattertype	9.20	9.57				
25	som	9.67	9.85				
26	fins	9.85	10.35	9.87	10.47	1.0	
-	-						
27	sjarmerende	10.77	11.45				
28	selvopptatt	11.55	12.33	11.58	11.85	1.0	Only ”selv” detected
29	full	12.61	12.85				
30	av	12.85	12.92	12.87	12.96	$5 \cdot 10^{-17}$	Decision: NO
31	skrøner	12.92	13.50				
32	og	13.50	13.58				
33	uten	13.58	13.82				
34	særlig	13.82	14.05				
35	mange	14.05	14.29				
36	hemninger	14.29	14.99				

Table 5: Individual words recognised from KWS. Timestamps are given in seconds.

#FA	Word	Sys. start	Sys. end	Correct word
1	og	3.48	3.57	#10 når, or #11 folk
2	av	4.14	4.26	#13 at
3	og	9.78	9.87	#25 som
4	og	11.85	11.94	#28 selvopptatt - middle
5	er	14.73	14.91	#36 hemninger - ending

Table 6: KWS false alarms with scores of 1.0. Timestamps are given in seconds.

5.2.2 Full Grammar

A full grammar system recognises the words displayed in Table 7, and falsely recognises the words in Table 8. The most likely path through the word lattice is:

> "unn a kjapp vis ikke korgen av selv og faxen at ulver katja a på mate dem mest salgbar fatter på pins skei mene selv å tatt kulda skauen ordens alma henning er"

#	Word	Ref. start	Ref. end	Sys. start	Sys. end	Sys. score	Comment
6	vis	1.51	1.74	1.47	1.77	1.0	
7	ikke	1.74	2.02	1.77	2.04	1.0	
9	selv	3.21	3.42	3.21	2.48	1.0	
13	at	4.18	4.32	4.14	4.35	1.0	
16	katja	5.75	6.28	5.73	6.39	1.0	
18	på	6.39	6.54	6.39	6.57	1.0	
21	den	7.22	7.69	7.23	7.68	0.008	
22	mest	7.74	8.20	7.77	8.22	1.0	
28	selvopptatt	11.55	12.33	11.58	11.85	1.0	Only "selv" detected
30	av	12.85	12.92	12.87	12.96	$5 \cdot 10^{-17}$	Decision: NO

Table 7: Correctly recognised words. Timestamps are given in seconds.

#FA	Word	Sys. start	Sys. end	Correct word
1	og	3.48	3.57	#10 når, or #11 folk
2	på	9.60	9.84	
3	er	14.73	14.91	#36 hemninger - ending

Table 8: KWS false alarms with scores of 1.0. Timestamps are given in seconds.

5.3 Parameter Tuning

The lattice generation step allows configuration of the acoustic scale and lattice beam. This experiment uses pruned language model on the same audio clip as Section 5.2, where the spoken utterance is:

> "Hun lar seg på et vis ikke korrigere. Selv når folk skjønner at hun lyver. Katja er på mange måter den mest salgbare forfattertype som fins. Sjarmerende, selvopptatt, full av skrøner og uten særlig mange hemninger."

5.3.1 Acoustic Scale

The acoustic scale in the lattice generating step is set to new values in this test. The word sequence recognised as most likely by the KWS system is presented for each acoustic scale value.

Acoustic scale = 0.2:

> "la kjapp vis ikke kurere selv og fag skjønn at river katja på ung måte den mest salgbar fatter på pins skei mene selv å katt kulda skrå nordens alma henning er"

Acoustic scale = 0.5:

> "lars av ris ikke korene selv og fag skjønn av tull iver katja a på matte den mest salgbar fatter på pins skei mene selv å katt kulda grå nordens alma henning er"

Acoustic scale = 2:

> "unn a kjapp vis ikke korn gjerde selv og fag skjønn av tull iver katja a på matte den mest salgbar fått rips om inn sved skei mene selv å katt kulda skauen ordens alma henning er"

Acoustic scale = 5:

> "unn a kjapp vis ikke kår g det selv og fag skjønn av tull iver katja a på matte den mest salgbar fått rips om inn sved skei med med selv å katt kulda skauen ordens alma henning er"

The utterance recognised with an acoustic scale of 0.2 included one tri-gram from the language model, "måte den mest". The rest of the acoustic scale experiment results contain no modelled tri-grams at all.

5.3.2 Lattice Beam

The lattice beam value is raised to 20 from 8.5 in this test. Figure 15 displays a snippet of the word lattice generated with this value.

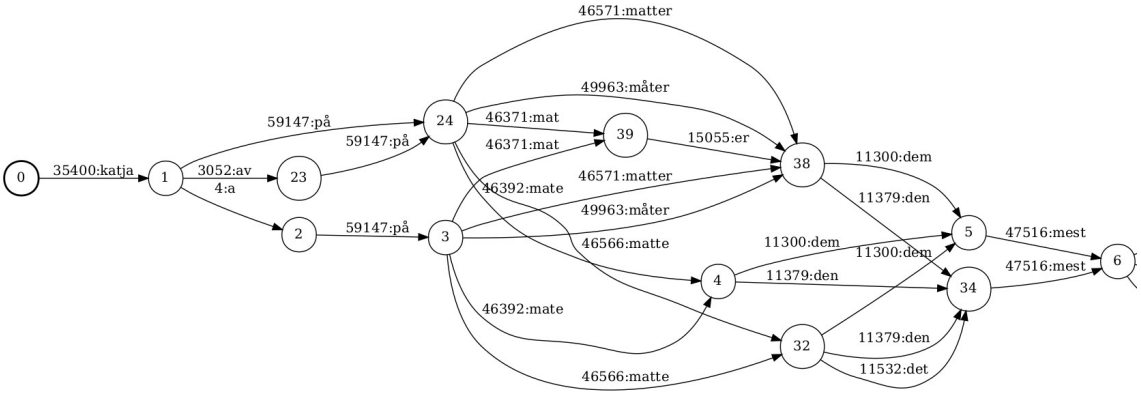


Figure 15: Snippet of word lattice when the lattice beam value is raised to 20. Correct utterance for this part is "katja er på mange måter den mest".

6 Discussion

The traditional Automatic Speech Recognition (ASR) approach for Keyword Spotting (KWS) is chosen as it is flexible when searching for longer keyword phrases. The other approach discussed in Section 2.1, regarding comparing models of keywords with a universal background model, will have to have readily available models of every keyword phrase searchable. This is not practical as the keyword phrases used to find exact anchor points are supposed to be unique, meaning that a new model has to be used for every KWS attempt.

The proposed solution will be more general and suitable for this task. Searching a word lattice containing several likely sequences of words allow for a robust KWS system that spots phrases that do not appear as the most probable phrase in the direct ASR result. This is long as they are recognised with a similar likelihood.

Either way, the recognition is as good as the coverage of the dictionary. Out-of-dictionary words will not appear in any of these KWS systems without a particular way to handle them. The simplest solution to this issue is to only search for known words. This solution can be realised in this application, as audiobook segmentation can be done wherever as long as the splitting is precise. Unique phrases of known words can surly be found throughout audiobooks within required steps. These phrases do not have to get particularly long on average as they quickly become unrepeated in most cases. This can nonetheless be assured by simple text searches of selected phrases.

A complete system for aligning an entire audiobook with its transcript will first merge all audio recordings and text clips of the book into one long audio file and a single text. The KWS system will then find anchor points used to divide the text and speech into corresponding segments of suitable lengths for the alignments. These anchor points can be placed next to figures in the text to work around potential text notes inside these, or other illustrations possibly described by the audiobook reader. Forced alignment can then be done with corresponding audio and text segments.

6.1 Implementation

The original system built for South-African languages is designed to include three pitch features in its high-resolution MFCCs. This system does however not include pitch feature vectors as Norwegian is not a very tonal language. Similar words that are differentiated by pitch can generally be distinguished by their grammatical context. Including pitch features will increase the system's complexity without improving performance much, if at all, and is therefore left out.

The pronunciation lexicon chosen for this implementation is relatively short and contains no alternative pronunciation options. This is used instead of larger vocabularies because the FST decoding graph gets too large otherwise. The Norwegian pronunciation lexicon used contains nearly 90 000 words. The lexicons of the original implementations of Afrikaans and Sesotho contain nearly 30 000 words, while the isiZulu lexicon contains almost 20 000 words, for reference.

A much larger, automatically generated lexicon with multiple pronunciation options for each word, totalling over 600 000 words, is at first attempted used to construct a decoding graph. The FST construction fails in this case at an early stage because of RAM shortage, using a computer with 64GB available RAM. A sizeable vocabulary increases the proportions of the decoding graph because the grammar model gets larger. This system built on the large lexicon is too computationally expensive as the lattice decoding process is even more RAM demanding than the FST creation algorithms and is therefore not used.

Even a full tri-gram grammar model of the shorter vocabulary makes the decoding graph too large for lattice generation utilising 64GB RAM. The size of the FST is reduced by pruning the grammar model. This is not ideal as unusual word sequences not encountered in the training set will not be considered in the decoding process, as explained in Section 2.3. Books are often written differently than news texts, which this n-gram model is trained on, making the pruned model more likely to come across word combinations not included in the model. It is therefore better to increase the computational power and use the full grammar model. 128GB RAM is sufficient for this system

using the shorter lexicon.

The memory limitation will not be a problem later on as the system segments the audio prior to KWS, ensuring that the computational requirements do not scale with the length of audio files. A KWS system built on ASR will by default have the same memory issues as a forced alignment system. Segmentation using Voice Activity Detection (VAD) negates this issue. The VAD used for segmentation uses ordinary thirteen-dimensional MFCC features which are extracted from 8kHz sampled audio. The sampling frequency for this extraction step is not changed from the original system since the VAD works well as it is.

6.2 Results

This KWS system realisation is clearly not usable for its purpose, as visible from the results in Section 5. The decoded lattices are far off the truth, making the KWS attempts fail.

The grammatical model looks at first glance guilty for the poor results, as the recognised utterance possibilities visualised by the lattices in Figure 13, 14, and 15 do not make sense grammatically. The fact that the recognised words do not look utterly random from an acoustic point of view supports the hypothesis that the language model holds the blame. Examples of this are "brilleglassene" recognised as "drill gassen" in Section 5.1.1 and 5.1.2, and "Sjarmerende, selvopptatt, full av skrøner (...)" to "skei mene, selv å katt, kulda skauen (...)" in Section 5.2.1. However, looking at the results of acoustic scale tuning in Section 5.3.1, it is apparent that neither the acoustic or grammatical model is by themselves accountable for the poor performance, as the results do not get better when one of the models is weighted significantly more than the other.

Comparing the pruned grammar experiment results in Section 5.1.1 and 5.2.1 to the full models results in Section 5.1.2 and 5.2.2 show no notable impact to the quality of the recognition. N-gram model pruning should make the results worse to some degree, but it is not visible in these tests as both provide unrecognisable results. Pruning should especially affect sequences that do not appear in the training data and are only included in the n-gram model as a smoothing effect, see Section 2.3.

The lack of grammar modelling of unknown word sequences is likely not the sole reason for the performance issues either, as the first segment of the utterance in Section 5.2 only contains known words. This segment is recognised independently from the later part of the same utterance but is not recognised any better than the other utterances presented in the results section. It can be speculated that out-of-vocabulary words introduce follow-up errors because the language model is led in the wrong direction trying to find likely following words to the system's false prediction of unknown words. This hypothesis can not be verified from the observations as none of the results gives logical results; the segment including exclusively known words is recognised as poorly as the others.

From all the utterances recognised it is evident that the language model does not influence the recognition at all. Some of the recognitions are done with pruned models. One would expect that the system only predicts tri-grams included in the grammatical model in these cases, as the smoothing effect that models unseen word combinations is removed. The fact that the most likely recognised utterances in all experiments include close to none modelled tri-grams suggests that the decoding graph is assembled falsely. It looks more specifically like the language model is not added to the decoding FST correctly, as the grammar appears to be non-existent while the acoustics make some sense. The system does e.g. recognise the word "Katja" correctly several times with different combinations of neighbouring words. Continuity between recognitions suggests a systematic error, such as a defect decoding graph.

The i-vector extractor used for feature extraction in this keyword spotter is used to successfully conduct ASR experiments, see Table 4. The i-vector representation is therefore most likely not defective. Nonetheless, the feature extraction step should not confuse the results in this manner if it is faulty. The experiment recognitions should result in mostly modelled tri-grams even if the input features are incorrect, especially when the language model is heavily favoured. This supports the hypothesis that the system error is located at the decoding step.

The KWS system made multiple false alarms. Examples of these are listed in Table 6. The false alarms are primarily short, common words that are recognised in place of other phonetically resembling words. It makes sense for an ASR system to conclude that common words are spoken often, as they by default are given higher likelihoods by the language model to appear in varied sentence combinations. The false alarms observed in the results are, as discussed, not correctly impacted by the grammar model in this case. False alarms should either way not be an issue for this application since false detections of unique keyword phrases are fairly unlikely for a functioning system.

Results of simple ASR tests presented in Section 5.1.3 show that decent performance is feasible when recognising utterances involving words absent from the vocabulary. These results suggest, as expected, that unpruned grammar models tend to perform better than pruned ones. The results displayed in Table 4 confirms that using i-vector features can be a good idea. The ASR system built on i-vector feature representation performs better than the more elementary thirteen-dimensional MFCC system. This can be because the system realisation is more complex overall, but using the more advanced portrayal of acoustic properties can nevertheless be beneficial.

Increasing the lattice beam allows deeper searching of keyword phrases. Section 5.3.2 displays an example of an expanded lattice. Increasing the lattice beam is not a solution to system errors, as the system will draw the same conclusion as earlier and not get any more confident. It can however help tune a working system to spot less likely keyword phrases, or alternatively, increase the decoding speed by lowering the value if the ASR is exceptionally precise.

6.3 System Issues

Several compromises are made when assembling the system. A small dictionary and pronunciation lexicon is used to compress the final decoding graph, keeping the memory requirements within practical limits. Including multiple pronunciation variations for words is beneficial, as it will make the system more robust when handling alternative pronunciations. It is experimented with pruned language models to compress the decoding graph. N-gram pruning is not advisable as it can hinder spotting unique phrases, which the system is meant to handle. It is not ideal that the language model is trained on a database consisting of mainly news texts when the application is all kinds of books. The model should be trained on the same type of data that the system is meant to be applied on.

Audiobooks are originally sampled at a significantly higher frequency than what this KWS system utilises. According to the Nyquist theorem, downsampled audio recordings lose information of high-frequency components. The Nyquist theorem states that the sampling frequency must be at least double of a signal's highest frequency component to reconstruct the signal perfectly. 16kHz sampled audio will only be able to successfully portray frequencies up to 8kHz fully, while 44.1kHz is able to retain frequencies up to 22.05kHz. Keeping original recordings can only make performance better as more information is available. This would demand more computational power and lower the decoding speed. Higher sampling frequencies should regardless be used when it is realisable, as it should improve performance.

None of the compromises mentioned above is responsible for all of the significant performance issues of the complete system, individually or altogether. The issues are more likely caused by incorrect assembly of the system, as discussed earlier. The results suggest that the problems occur at the decoding phase in a way that does not include the language model as intended because "illegal" results are achieved.

The main structural differences to the original South African spoken term detector are the change of sampling frequency of the input audio and the absence of pitch features. None of these differences influences the system negatively, as discussed. The original system provides a test for validating the setup process. The Afrikaans utterance "hierdie jy moet elkeen van ons laat met 'n gevoel van se realistiese" is recognised perfectly, making any KWS attempt within the recording succeed. The original system does likely not always have perfect performance, but the test shows that the system has good potential. This indicates that a functional Norwegian adaptation of the system

should be achievable if correct models are supplied.

6.4 Alternative Solutions

The KWS solution proposed in this thesis is promising in theory, and is worth attempting fixed. Some alternatives do however exist.

The need for phrase search can be removed altogether if audiobooks are produced according to some guidelines. Audiobook readers can manually segment the recording into appropriate lengths when producing them while marking the divisions in the text. The narrator can additionally note down information about illustrations and other deviations they might make from the script. This assures that corresponding audio recordings and texts are accessible for time alignment. This is a good solution for future audiobook productions, but will not work on existing audiobooks.

An alternative to pre-segmentation before the alignment is to develop another forced alignment strategy that handles longer text than audio clips. Using such a method will allow alignments of small steps at the time. This will result in a procedure that resembles online speech recognition in the sense that alignment happens continuously throughout the audio recording without a reference of endpoints.

7 Conclusion

In conclusion, a Keyword Spotting (KWS) system suited for finding unique phrases in audio recording for Norwegian Bokmål is constructed. The KWS system is built on ordinary Automatic Speech Recognition (ASR) technology that produces recognition options of audio recordings. This technology enables precise segmentation of audio and text into suitable lengths by identifying unique phrases in the speech. These segments are used as inputs in forced alignment systems, producing time-aligned audiobook transcripts.

A decoding graph combining Norwegian acoustics, phonetics, phonetic context, and grammar is constructed in a Finite-State Transducer (FST) format, which is used to decode i-vector feature inputs into lattices containing recognition results. A decoding graph using pruned language model is additionally made to have a lightweight system that can run on ordinary computers.

The performance of both these systems is however not as expected. Results of experiments suggest that the decoding graph is assembled incorrectly. Grammatical rules are not followed in the recognition attempts, meaning that the language model is not included in the equation. Several compromises are made, including downsampling of audio, and using a short dictionary and pronunciation lexicon. The compromises influence the quality of keyword spotting, but the extent of this is not visible as the system has more critical issues.

This KWS approach is reasonable for the application, so fixing the system will be worthwhile. However, alternatives exist. The simplest solution is to provide clear guidelines for future audiobook production, which can eliminate the need for automatic segmentation using KWS altogether.

8 Further Work

The KWS strategy proposed in this thesis is interesting even though the results are lacking, as the strategy is well suited for the application. The system should therefore preferably be fixed. A working system will solve most of the challenges hindering the Montreal Forced Aligner from being automatically applied on entire books. The first step to fix the keyword spotter should be to remake the decoding graph as it appears to by faulty, and make sure that all parts are put together correctly.

The system should be adjusted to use higher frequency sampled audio inputs in order to improve its performance, as discussed in Section 6.3. A new i-vector extractor must be trained to realise this enhancement.

Other techniques can alternatively be used to work around the issues, for example the strategies proposed in Section 6.4.

A Norwegian Nynorsk keyword recognition system should additionally be made to handle a larger diversity of books. New language models, acoustic models, and i-vector extractors have to be made to realise this system.

References

[1] Pieter Uys. Kaldi Keyword Spotting. https://github.com/pieter129/KaldiSpokenTermDetectionWrapper. 2020.

[2] J.R. Rohlicek et al. Continuous hidden Markov modeling for speaker-independent word spotting. In: *International Conference on Acoustics, Speech, and Signal Processing,* 1989, 627–630 vol.1. DOI: 10.1109/ICASSP.1989.266505.

[3] X. Huang et al. Spoken Language Processing: A Guide to Theory, Algorithm, and System Development. Prentice Hall PTR, 2001. ISBN: 9780130226167.

[4] Haytham M. Fayek. Speech Processing for Machine Learning: Filter banks, Mel-Frequency Cepstral Coefficients (MFCCs) and What's In-Between. 2016. URL: https://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html.

[5] Frederic J. Harris. Chapter 3 - Multirate FIR Filters for Interpolating and Desampling. In: *Handbook of Digital Signal Processing.* Ed. by Douglas F. Elliott. San Diego: Academic Press, 1987, pp. 173–287. ISBN: 978-0-08-050780-4. DOI: https://doi.org/10.1016/B978-0-08-050780-4.50008-4.

[6] Alex Acero and Xuedong Huang. Augmented Cepstral Normalization for Robust Speech Recognition. In: *Proc. of the IEEE Workshop on Automatic Speech Recognition.* Dec. 1995.

[7] Giampiero Salvi. A Note on Delta Features. Aug. 2018.

[8] M. Yip. Tone. Cambridge Textbooks in Linguistics. Cambridge University Press, 2002. ISBN: 9780521774451.

[9] Pegah Ghahremani et al. A pitch extraction algorithm tuned for automatic speech recognition. In: *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).* 2014, pp. 2494–2498. DOI: 10.1109/ICASSP.2014.6854049.

[10] Najim Dehak et al. Language Recognition via I-Vectors and Dimensionality Reduction. In: *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH.* Aug. 2011, pp. 857–860. DOI: 10.21437/Interspeech.2011-328.

[11] Patrick Kenny, Gilles Boulianne and Pierre Dumouchel. Eigenvoice modeling with sparse training data. In: *Speech and Audio Processing, IEEE Transactions on* 13 (June 2005), pp. 345–354. DOI: 10.1109/TSA.2004.840940.

[12] Najim Dehak et al. Front-End Factor Analysis for Speaker Verification. In: *IEEE Transactions on Audio, Speech, and Language Processing* 19.4 (2011), pp. 788–798. DOI: 10.1109/TASL.2010.2064307.

[13] Srikanth R. Madikeri et al. Implementation of the Standard I-vector System for the Kaldi Speech Recognition Toolkit. In: 2016.

[14] Mehryar Mohri, Fernando Pereira and Michael Riley. Speech Recognition with Weighted Finite-State Transducers. In: *Springer Handbook of Speech Processing.* Ed. by Jacob Benesty, M. Mohan Sondhi and Yiteng Arden Huang. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 559–584. ISBN: 978-3-540-49127-9. DOI: 10.1007/978-3-540-49127-9_28. URL: https://doi.org/10.1007/978-3-540-49127-9_28.

[15] K. L. P. Mishra and N. Chandrasekaran. Theory of Computer Science: Automata, Languages and Computation. In: 2006.

[16] Michael Riley. Compose FST. Accessed on 04.06.2022. Apr. 2018. URL: https://www.openfst.org/twiki/bin/view/FST/ComposeDoc.

[17] Michael Riley. Determinize FST. Accessed on 05.06.2022. Feb. 2019. URL: https://www.openfst.org/twiki/bin/view/FST/DeterminizeDoc.

[18] Michael Riley. Minimize FST. Accessed on 05.06.2022. Feb. 2019. URL: https://www.openfst.org/twiki/bin/view/FST/MinimizeDoc.

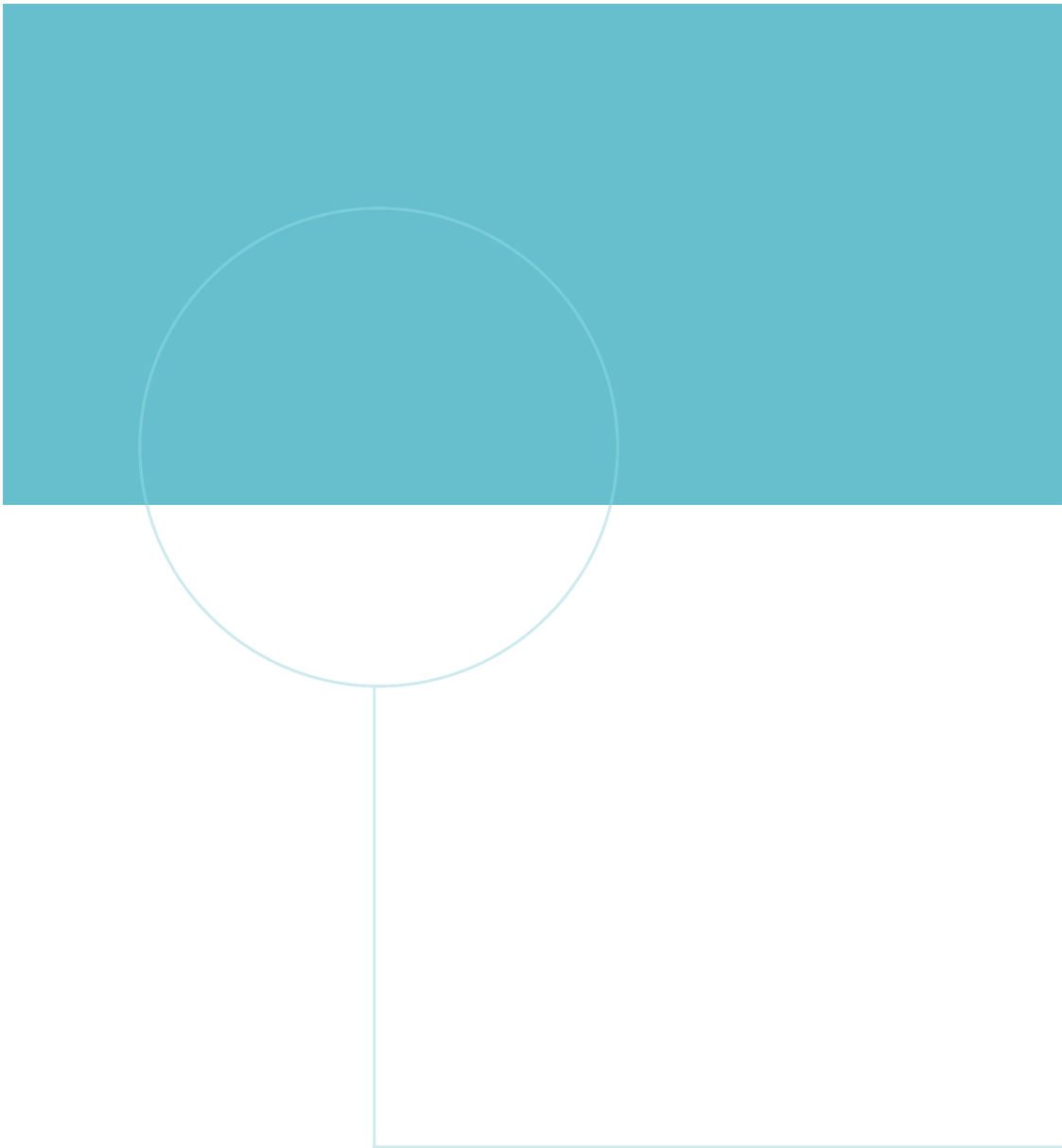
[19] Daniel Povey et al. Generating exact lattices in the WFST framework. In: *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).* 2012, pp. 4213–4216. DOI: 10.1109/ICASSP.2012.6288848.

[20] Martial Michel et al. Framework for Detection Evaluation (F4DE). https://github.com/usnistgov/F4DE. 2017.

[21] Kaldi: Decoding-graph creation recipe. Accessed on 21.05.2022. URL: https://kaldi-asr.org/doc/graph_recipe_test.html.

[22] Andreas Stolcke. Srilm – An Extensible Language Modeling Toolkit. In: *Proc. Intl. Conf. on Spoken Language Processing* 2 (2002), pp. 901–904.

[23] Montreal Forced Aligner documentation. June 2022. URL: https://montreal-forced-aligner.readthedocs.io/en/latest/.



NTNU

Norwegian University of
Science and Technology