FINNISH LANGUAGE SPEECH RECOGNITION FOR DENTAL HEALTH CARE

SUOMENKIELINEN PUHEENTUNNISTUS HAMMASHUOLLON SOVELLUKSISSA

A THESIS SUBMITTED TO THE DEPARTMENT OF
INFORMATION AND COMPUTER SCIENCE OF
AALTO UNIVERSITY SCHOOL OF SCIENCE
IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE OF
LICENTIATE OF SCIENCE IN TECHNOLOGY

Seppo Enarvi
March 2012

Aalto University

AALTO UNIVERSITY SCHOOLS OF TECHNOLOGY PO Box 11000, FI-00076 AALTO http://www.aalto.fi	ABSTRACT OF THE LICENTIATE THESIS

Author: Seppo Enarvi

Title: Finnish Language Speech Recognition for Dental Health Care

School: School of Science

Department: Department of Information and Computer Science

Professorship: Computer and Information Science	Code: T101Z

Supervisor: Erkki Oja

Instructor(s): Mikko Kurimo, Janne Pylkkönen

Abstract:

A significant portion of the work time of dentists and nursing staff goes to writing reports and notes. This thesis studies how automatic speech recognition could ease the work load.

The primary objective was to develop and evaluate an automatic speech recognition system for dental health care that records the status of patient's dentition, as dictated by a dentist. The system accepts a restricted set of spoken commands that identify a tooth or teeth and describe their condition. The status of the teeth is stored in a database.

In addition to dentition status dictation, it was surveyed how well automatic speech recognition would be suited for dictating patient treatment reports. Instead of typing reports with a keyboard, a dentist could dictate them to speech recognition software that automatically transcribes them into text. The vocabulary and grammar in such a system is, in principle, unlimited. This makes it significantly harder to obtain an accurate transcription.

The status commands and the report dictation language model are Finnish. Aalto University has developed an unlimited vocabulary speech recognizer that is particularly well suited for Finnish free speech recognition, but it has previously been used mainly for research purposes. In this project we experimented with adapting the recognizer to grammar-based dictation, and real end user environments.

Nearly perfect recognition accuracy was obtained for dentition status dictation. Letter error rates for the report transcription task varied between 1.3 % and 17 % depending on the speaker, with no obvious explanation for so radical inter-speaker variability. Language model for report transcription was estimated using a collection of dental reports. Including a corpus of literary Finnish did not improve the results.

Date: 30.3.2012	Language: English	Number of pages: 70

Keywords: Automatic speech recognition, language modeling

Aalto-yliopisto

AALTO-YLIOPISTO TEKNIIKAN KORKEAKOULUT PL 11000, 00076 AALTO http://www.aalto.fi	LISENSIAATINTUTKIMUKSEN TIIVISTELMÄ

Tekijä: Seppo Enarvi

Työn nimi: Suomenkielinen puheentunnistus hammashuollon sovelluksissa

Korkeakoulu: Perustieteiden korkeakoulu

Laitos: Tietojenkäsittelytieteen laitos

Professuuri: Informaatiotekniikka | Koodi: T101Z

Työn valvoja: Erkki Oja

Työn ohjaaja(t): Mikko Kurimo, Janne Pylkkönen

Tiivistelmä:

Hammaslääkärien ja hoitohenkilökunnan työajasta huomattava osa kuluu raportointiin ja muistiinpanojen tekemiseen. Tämä lisensiaatintyö tutkii kuinka automaattinen puheentunnistus voisi helpottaa tätä työtaakkaa.

Ensisijaisena tavoitteena oli kehittää automaattinen puheentunnistusjärjestelmä hammashuollon tarpeisiin, joka tallentaa potilaan hampaiston tilan hammaslääkärin sanelemana, ja arvioida järjestelmän toimivuutta. Järjestelmä hyväksyy rajoitetun joukon puhuttuja komentoja, jotka identifioivat hampaan tai hampaat ja kuvaavat niiden tilaa. Hampaiden tila tallennetaan tietokantaan.

Hampaiston tilan sanelun lisäksi tutkittiin kuinka hyvin automaattinen puheentunnistus sopisi potilaiden hoitokertomusten saneluun. Näppäimistöllä kirjoittamisen sijaan hammaslääkäri voisi sanella hoitokertomukset puheentunnistusohjelmistolle, joka automaattisesti purkaisi puheen tekstimuotoon. Tämän kaltaisessa järjestelmässä sanasto ja kielioppi ovat periaatteessa rajoittamattomat, minkä takia tekstiä on huomattavasti vaikeampaa tunnistaa tarkasti.

Status-komennot ja hoitokertomusten kielimalli ovat suomenkielisiä. Aalto-yliopisto on kehittänyt rajoittamattoman sanaston puheentunnistimen, joka soveltuu erityisen hyvin suomenkielisen vapaamuotoisen puheen tunnistamiseen, mutta sitä on aikaisemmin käytetty lähinnä tutkimustarkoituksiin. Tässä projektissa tutkimme tunnistimen sovittamista kielioppipohjaiseen tunnistukseen ja todellisiin käyttöympäristöihin.

Hampaiston tilan sanelussa saavutettiin lähes täydellinen tunnistustarkkuus. Kirjainvirheiden osuus hoitokertomusten sanelussa vaihteli 1,3 ja 17 prosentin välillä puhujasta riippuen, ilman selvää syytä näin jyrkälle puhujien väliselle vaihtelulle. Kielimalli hoitokertomusten sanelulle laskettiin kokoelmasta hammaslääkärien kirjoittamia raportteja. Kirjakielisen aineiston sisällyttäminen ei parantanut tunnistustulosta.

Päivämäärä: 30.3.2012	Kieli: Englanti	Sivumäärä: 70

Avainsanat: Automaattinen puheentunnistus, kielen mallintaminen

Acknowledgments

This thesis has been carried out in the Speech Group at the Department of Information and Computer Science (ICS) in Aalto University School of Science. The work has been funded by In Net Oy and lasted from January 2011 to January 2012.

The project was managed by Esko Ristkari from In Net Oy, and my instructors Professor Mikko Kurimo and Janne Pylkkönen from ICS. In particular, credit is due to Janne Pylkkönen for his continuous support. Equally deep gratitude goes to Mikko Kurimo for providing me this opportunity and carefully reading and commenting my thesis.

Contents

1

Introduction

1.1 Motivation

The priority of dentists and nursing staff is interaction with the patient and concentration on the treatment. A significant portion of their work time goes to writing reports and notes, and interacting with a computer. Much of this work load could be automated, and controlled by the dentist using his or her voice. Typing on a keyboard may also be out of the question in a medical environment for hygienic reasons.

Recent advances in automatic speech recognition have made it possible to transcribe free speech dictation into text and reports in real time. A good recognition accuracy is achieved, because the dentists rather try to speak clearly than type the reports on a keyboard.

Even better recognition accuracy is achieved by restricting the language according to a strict grammar. Such a grammar could include commands for controlling the dentist's instruments while the dentist could concentrate on the treatment.

It would be possible to record minutes from a patient's visit and store the audio files into a database, but without speech recognition the dentist could not search or read the text afterwards, only play back the recording.

1.2 Background

This thesis investigates the use of automatic speech recognition to assist dental professionals in their work. Foundations for the thesis are in the speech recognition research that Aalto University School of Science, former Helsinki University of Technology, has been carrying out

since the 1980s. This long research has resulted in the development of an unlimited vocabulary continuous speech recognizer. The recognizer is language-independent, but it is particularly well suited, if not the best of its kind, for languages such as Finnish, Estonian, and Turkish, in which words tend to consist of several morphemes. So far it has mostly been used in research projects.

This thesis project has been performed in collaboration with In Net Oy, a Finnish software company that has specialized in developing software for dental care. The idea is to couple dental software with speech recognition functionality. Automatic speech recognition can be used in applications including the dictation of dentition status and notes concerning e.g. X-ray images. The Aalto University speech recognizer is tailored to suit such applications, and evaluated in selected cases. The client has gained earlier experience in commercialization of automatic speech recognition in dental applications, and collected suitable text corpora and speech material from dentists for training the necessary statistical models.

The National Archive of Health Information, KanTa, for its part, has guided the type of information that the system is designed to store. KanTa is a collective name for information systems that are gradually being introduced in Finland. These include a centralized archive of patient records and electronic prescriptions. Eventually all dental clinics in Finland will have to implement these services.

1.3 Project Goals

Four concrete goals were agreed for the project. First, a simple grammar is defined for commands that can be used to dictate dentition status, and a free speech language model is adapted for the vocabulary specific to dentistry. An explicit grammar for the dictation of dentition status should enable high, near perfect recognition accuracy. Free speech dictation has secondary priority, and it is acknowledged that the task is error-prone. Creating a free speech language model specifically for documentation of patient's medical records, using the corpus provided by the client, should improve recognition accuracy in such context.

The client has developed an integrated information system for dental care, and desktop software that dentists use to access the information system. The second goal is to integrate the recognizer into the user interface software. The recognizer is running on a server computer, or as a background process in the dentist's desktop computer. The development in the processing power of personal computers, and advances in speech recognition methodologies, have made automatic speech recognition possible in real time even on a desktop computer.

A communication protocol is specified for interaction between the user interface application and the server process. A server process is implemented that listens to a network interface for recognition requests from the user interface software, and performs the recognition using the Aalto University recognizer.

It is the responsibility of the client to add the necessary functionality to the user interface software for connecting to the recognition server. The client also provides the necessary hardware, including the server computer, microphone and headphones. This makes a clear division between the responsibilities of the client and Aalto University.

The third goal that was set to the project is evaluation of speaker-dependent adaptation of the acoustic models. There are two strategies for the adaptation. The adaptation is called supervised, if it uses training data of which the true transcription is known. Supervised adaptation is possible e.g. by providing the speaker training sentences to pronounce. In unsupervised strategy the recognizer adapts to the speaker's voice gradually while the speaker is using the application. The goal was to evaluate these strategies, but lack of time forced to leave speaker adaptation for future work. The concept is still presented in this thesis.

The fourth goal that was set is to collect feedback from the users and evaluate the usability of the recognizer. Feedback from the users is an important factor, when evaluating a product that is used by dentists, in addition to the recognition accuracy and technical performance. Currently the client is starting to perform user tests and collect feedback from dental professionals.

There are multiple reasons why Aalto University was interested in participating this project:

1. To study how well a grammar-based dictation system can be implemented on a modern large vocabulary continuous speech recognizer.
2. To study how well a general-purpose speech recognizer can be adapted to dictating patient information using jargon that is very specific to the dental domain.
3. To see how good results can be achieved when there is exceptionally plenty of training data available.
4. To see what kind of issues emerge when the recognizer is used in practical environments.
5. To find relevant topics for further research for improving speech recognition.

1.4 Thesis Overview

The central contribution of this thesis is the integration of speech recognition into the information system developed by the client and used by dentists in a medical environment. More specifically, the work includes

- specification of a network protocol for communication between the recognition server and the user interface software,
- development of the server software that performs the recognition using the Aalto University recognizer,

- building a command grammar for dentition status dictation,
- constructing a large vocabulary language model suited to the language and speaking style used by dentists in their work,
- studies on voice activity detection for activating command input,
- studies to improve recognition accuracy at certain problematic areas, and
- studies on optimizing recognition performance by adjusting decoding parameters.

The rest of this thesis is organized into chapters in the following way.

- Chapter 2 defines some basic concepts that are necessary for analyzing speech and natural languages, and presents issues involved in recognizing casual speech.
- Chapter 3 develops theory for automatic speech recognition, measuring recognition accuracy and assessing confidence on recognition result.
- Chapter 4 introduces our speech recognizer and the dental information system, presents the algorithm that was used for voice activity detection, and fine-tunes the recognition parameters.
- Chapter 5 describes the models and algorithms that were developed for dentition status dictation, and presents experimental results.
- Chapter 6 describes the process of building a language model for report dictation, and evaluates the language model.
- Chapter 7 summarizes lessons learned and points to future directions.
- Appendix A contains a detailed specification of the network protocol used for communication between the information system and the recognizer.

2

Properties of Spoken Language

2.1 Graphemes and Phonemes

Analyzing speech requires a distinction to be made between *graphemes*, i.e. letters, digits, and other symbols of a writing system, and the sounds of a spoken language. The smallest units of speech sound are called *phones*. Every word is considered to be pronounced as a sequence of phones.

The phones of any language can be grouped into *phonemes*. The phones that are members of the same phoneme are called *allophones*. Each allophone has a slightly different sound, but all the allophones of a phoneme are considered semantically equivalent in the language in question. The distinction between a phone and a phoneme is rather small and these two terms are sometimes used interchangeably. For example, the *t* in words *top* and *stop* is pronounced slightly differently in each of the words, although most people might not even notice the difference. These different sounds are still members of the same phoneme, since substituting one with the other will not lead to a different word, only to a bit odd-sounding one. [8]

The distinction between graphemes and phonemes is bigger in some languages than in others. Finnish has a good, although not exact, correspondence between graphemes and phonemes. English is particularly notorious for having more than one way of spelling nearly every phoneme, and more than one way of pronouncing every letter.

A fundamental component of any speech recognition system is an *acoustic model* that contains a statistical representation of each phoneme. The model is used to classify audio samples into phonemes. A pronunciation dictionary, often called a *lexicon*, gives the pronunciation of each word. Using the lexicon, phonemes can be translated into graphemes.

A speech recognizer cannot perform well without any information of the language that is being recognized, even if it would be trivial to translate phonemes into graphemes. The grammar, or *language model*, models the probabilities of possible sequences of words, or other linguistic units. The choice of the unit affects the size of the language model, and the amount of required training data. A word might not be the best choice, if a practically unlimited number of words can be generated using prefixes, suffixes, inflections, and compound words.

2.2 Morphemes

Words can often be divided into a stem, and elements that vary the meaning of the base word. For example, the English word “unrealistic” has three meaningful elements: the stem “real” gives the word its base meaning, the suffix “istic” is added to mean something in accordance with “real”, and the prefix “un” inverts the meaning of the word.

In linguistics, such components that cannot be broken down into smaller parts without loosing all the semantic meaning, are called *morphs*. The concepts that morphs represent are called *morphemes*. In the above example, the morph “un” represents the negative morpheme, “not”. The same morpheme is represented in the word “impossible” by the morph “im”.

English language expresses some grammatical relationships morphologically, such as tense (look, looked), plurality (cat, cats), possession (you, your), but some grammatical relationships are only expressed by word order. For example, English noun phrase subjects, objects, and indirect objects are not inflected for case. Many other languages, including Finnish, express far more such grammatical relationships morphologically. Consider the following sentences. [15]

> Hän näki opiskelijan. [He saw the student.]
> Hän antoi opiskelijalle kirjan. [He gave a book to the student.]

In both cases the Finnish word “opiskelija” has been inflected by adding a suffix, but the English word “student” is not inflected. English language relies on word order and combining several words.

Languages that express grammatical relationships morphologically are called *synthetic languages*. As an opposite, in an *analytic language* words tend to consist of only few morphemes. Although English has some synthetic features, it is more of an analytic language. [15]

Finnish is a synthetic language, and words are commonly constructed from long morph sequences, whereas in analytic languages the same thing would be expressed using multiple words. Consequently, with regard to automatic speech recognition, modeling Finnish language as a sequence of words would be inefficient.

2.3 Colloquial Finnish

The Finnish language has a colloquial variant, used in informal speech. It is not anymore uncommon to use the colloquial variant also in written, especially electronic communications. The literary variant is only spoken in formal situations, such as newscasts. Comprehensive collections of written Finnish exist in electronic form that have commonly been used for training the language models for Finnish speech recognition, but text corpora of colloquial Finnish are rare.

The development of written Finnish is an exception in Europe, in that it is not based on the spoken dialect of some nationally important area, but a consciously created literary language. During the nineteenth century Helsinki became the capital of Finland, and Finnish became the official language, although the vast majority in Helsinki spoke Swedish. The written language was not based on any spoken dialect as such, so the written language was adopted also for spoken use in official situations. Furthermore, modern colloquial Finnish is not directly related to the written language, but evolved in Helsinki during the twentieth century from various linguistic tensions: [36]

- Proletarians who moved to town from various parts of the country, spoke their respective dialects.
- The speech of the educated classes was based on the written language, but any Swedish influences were deliberately eliminated due to nationalistic ideologies.

Helsinki has since then been the most important Finnish-speaking city, and its colloquial Finnish is combined with local dialects to form regional variants throughout the country. These differ from written Finnish in their vocabulary, grammar, phonology, and morphology. The set of phonemes is still practically identical in all the variants. With regard to a speech recognizer, the phonological and morphological differences can be seen as different vocabulary. This suggests that even though the same acoustic model can be used for both varieties, a language model trained using only written Finnish is inadequate for recognizing colloquial Finnish.

2.3.1 Colloquial Grammar

Some notable syntactical differences between the literary Finnish and the colloquial variant are given below. These changes are so natural to a Finnish speaker that they usually occur also when the colloquial variant is used in written [32].

- In modern colloquial Finnish, the first person plural verb form is replaced by the passive verb form.

  Me ostimme kirjan. [We bought a book.] → *Me ostettiin kirja.*

- The third person pronouns are very rarely used in spoken language, but the non-person equivalents are used instead.

Hän osti kirjan. [She bought a book.] → *Se osti kirjan.*

- The possessive suffix is usually omitted in spoken language, so that the personal pronoun cannot be omitted. The written language uses possessive suffixes, and the pronoun is often omitted.

  (Hänen) kirjansa ostettiin. [Her book was bought.] → *Sen kirja ostettiin.*

Because Finnish is an inflected language, word order can often be changed without changing the fundamental meaning of the clause. Different shades of meaning can still be communicated by different word orders. The most usual word order is subject—verb—object. Other word orders are more common in spoken than written language.

There are many differences in the structures used to form compound sentences from clauses, and some expressions that are used in literary Finnish do not sound natural when spoken. In practice, colloquial clauses are shorter. Long clauses that use dense expressions are broken into longer sentences consisting of simpler clauses when speaking. [43]

2.3.2 Colloquial Vocabulary

Even more important to speech recognition than grammatical differences is new vocabulary. In addition to completely new vocabulary such as slang words, there are many words whose colloquial pronunciation has become something that does not exist in the written language. Because each Finnish phone generally has its own grapheme, the colloquial pronunciations can be translated into written form. Some differences in vocabulary are given below.

- In everyday speech, verbs ending in *-ko* or *-kö* have the *-s* suffix added, and the *o* vowel deleted. In the example below, an unstressed diphthong also becomes a short vowel.

  Onko(s) teillä valkoista kirjaa? [Do you have a white book?] → *Onks teillä valkosta kirjaa?*

- Usually shorter equivalents are used for personal pronouns in colloquial Finnish. Some verbs have irregular forms.

  Minä tulen. [I will come.] → *Mä tuun.*

- Phonetic erosion has resulted in words that are not found in literary Finnish, but commonly used when speaking.

  Minkä lainen kirja se on? [What kind of a book is it?] → *Millanen kirja se on?*

Numerals from one to ten have shorter forms that are used in colloquial Finnish. The following six numerals are almost always shortened by removing the word-final /i/ in colloquial speech.

- *yksi* [one] → *yks*
- *kaksi* [two] → *kaks*
- *viisi* [five] → *viis*

- *kuusi* [six] → *kuus*

Numbers from one to nine also have their own names, different from the cardinal numbers used for counting. The last five of them have shorter variants used in colloquial speech. The last three are practically never used in the longer form.

- *ykkönen* [number one]
- *kakkonen* [number two]
- *kolmonen* [number three]
- *nelonen* [number four]
- *viitonen* [number five] → *vitonen*
- *kuutonen* [number six] → *kutonen*
- *seitsemäinen* [number seven] → *seiska*
- *kahdeksainen* [number eight] → *kasi*
- *yhdeksäinen* [number nine] → *ysi*

When it comes to dictating medical notes to a speech recognizer, the users try to speak clearly and pronounce correctly in standard Finnish. Still most speakers have tendency to pronounce some words, especially numerals, in a colloquial manner. Numerals are particularly problematic in the context of dental reporting, because they are used extensively and important to be recognized correctly.

When referring to teeth using the numbering system described in Section 5.1, number names are often used interchangeably with the cardinal numbers. Their spoken variants are so distant to the standard language that they will not be recognized correctly if only written language has been used to train the recognizer.

2.3.3 Casual Speech

The differences in grammar and vocabulary apply to colloquial Finnish in written as well. There are also aspects specific to speaking that make recognition of casual speech difficult. These issues become more and more important when speech recognition moves from controlled environments to real word applications.

- Strict sentence boundaries are not necessary in conversational speech. People try to avoid fragmented speech and combine clauses into long sentences using conjunctions [43].
- Spoken sentences may be grammatically incorrect, and contain only what is necessary to communicate the message.
- Disfluencies occur when the speaker hesitates or notices an error. Typically the speaker fills pauses with non-lexical utterances. For example, English speakers use *um*

and *uh* to announce a delay in speaking [12]. This may be followed by the speaker correcting the initial utterance.

- A phone may be altered by its neighboring phones at a morpheme or word boundary. For example, consider the English sentence *got you*. The word final /t/ sound causes the following /j/ sound to be translated into a /ʧ/. The phenomenon, called *sandhi*, is present in many languages, but particularly frequent in Finnish. It improves the flow of speech, which is why it can sometimes be heard in formal situations as well.

The first two properties are mostly related to conversational speech, but we found the lack of strict sentence boundaries and grammatical inconsistencies to be frequent also in medical reports. Lack of sentence structure clearly degrades speech recognition performance. The sandhi phenomenon is to some extent addressed by the triphone acoustic model described in Section 3.3. However, when it occurs at a word boundary, the change is sometimes transcribed in colloquial text, introducing new vocabulary:

sen lainen [of that kind] → *sellanen*

3

Automatic Speech Recognition

3.1 Speech Recognition as a Machine Learning Task

Speech recognition is a typical machine learning application. It can be formalized as a classification task, where we find the most probable word sequence $W=\{w_i\}$, given some features $X=\{\boldsymbol{x}_t\}$ that we observe from an audio signal, and a probabilistic model:

$$\arg\max_W P(W|X,\vartheta) \quad , \tag{1}$$

where ϑ represents the parameters of the probabilistic model. Since we cannot compute the *posterior* probability, $P(W|X,\vartheta)$, directly, we use the *likelihood* of the feature vector given the word sequence and the model parameters, $P(X|W,\vartheta)$, and the Bayes formula:

$$P(W|X,\vartheta)=\frac{P(W|\vartheta)P(X|W,\vartheta)}{P(X|\vartheta)} \tag{2}$$

The equation simplifies when we notice that the common denominator does not affect the order of the probabilities:

$$\begin{aligned} & \arg\max_W P(W|X,\vartheta) \\ = \; & \arg\max_W \frac{P(W|\vartheta)P(X|W,\vartheta)}{P(X,\vartheta)} \\ = \; & \arg\max_W P(W|\vartheta_L)P(X|W,\vartheta_A) \end{aligned} \tag{3}$$

First we have to decide how to extract a compact set of features, X, from an audio signal. The probability mass function $P(X|W,\vartheta_A)$ is called an acoustic model. Speech, whose textual transcription is known a priori, is used to create the acoustic model. The model parameters, ϑ_A, are estimated from features extracted from the speech signal.

The probability mass function that defines the prior probability of a word sequence, $P(W|\vartheta_L)$, is called a language model. Depending on the application, the language model can be specified as a grammar that strictly limits the language to specific sentences, or it can be a statistical model whose parameters, ϑ_L, are estimated from a large text corpus.

Thus we are left with the following issues.

1. How to extract X, the features, from an audio signal?
2. How to model the conditional probability distribution $P(X|W,\vartheta_A)$, the acoustic model? In practice we first need to convert the words W into phonemes using a pronunciation dictionary.
3. How to estimate the *prior* probability $P(W|\vartheta_L)$ of a sequence of words W, the language model?
4. How to solve the optimization problem, which is often called *decoding*?

The following sections discuss these problems in detail, and explain how the Aalto University recognizer has addressed them. A block diagram of a speech recognizer is depicted in Figure 3.

3.2 Feature Extraction

An observed audio signal has to be reduced into a time series of feature vectors, $X=\{\boldsymbol{x}_t\}$, that encapsulate enough information to discriminate between every two phonemes. Finding a suitable set of features that still is compact enough to be of practical interest, is arguably the most critical issue of the speech recognition problem.

Conventionally, acoustic features are computed from fixed-length audio segments, typically 25 ms long. Consecutive segments overlap with each other, positioned at 10 ms intervals. The segments are processed with the intention of extracting the phonetically essential information, and rejecting non-relevant information such as speaker characteristics and variations in recording condition. The result is typically a 39-element feature vector.

A number of different features have been proposed for speech recognition. The early solutions were often based on linear predictive analysis [39]. A linear predictor (LP) predicts the output of the system, s_n, as a linear combination of previous output values:

$$s_n = \sum_{k=1}^{p} a_k s_{n-k} \tag{4}$$

The predictor coefficients, a_k, are obtained by minimizing the mean-squared prediction error. The concept was originally used for speech coding, but the coefficients are directly usable features for speech recognition, since they characterize the spectral envelope of the signal. Their use have been reasoned with a simplistic model of speech production, where the coefficients specify a linear filter that corresponds to the human vocal tract [4].

While linear predictor coefficients characterize the frequency content of the signal, features based on *cepstral* representation have proven to work better in speech recognition. From spectral domain one can move on to cepstral domain by considering the logarithmic frequency spectrum as a signal, and performing another spectral transformation. Different authors have used slightly differing definitions of a *cepstrum*. In their original paper, Bogert et al. define it as the Fourier transform of the logarithmic power spectrum [7]:

$$C[f] = \mathscr{F}[\log S[f]] \quad , \tag{5}$$

where $S[f]$ is the transformation that gives the power spectrum of some function f, and $C[f]$ is the cepstrum of f. Cepstrum characterize the frequency content of the spectrum itself. In speech recognition systems, only a few coefficients are used as features. These correspond to a smooth estimate of the spectral envelope.

Oppenheim used the cascade Fourier transform → complex logarithm → inverse Fourier transform, which he calls the complex cepstrum, for deconvolution of speech [35]. Because of its usefulness in homomorphic filtering, many authors have taken this as the definition of cepstrum:

$$C[f] = \mathscr{F}^{-1}[\log \mathscr{F}[f]] \tag{6}$$

Using this definition, the cepstrum of a convolution of two signals is equivalent to the sum of the cepstra of the original signals. If we model speech as a convolution of components representing the excitation and the vocal tract, cepstrum maps the speech signal into a sum of these two components. Further assuming that the contributions of the excitation and the vocal tract vary rather differently with frequency, these components are separated in the speech cepstrum.

The cepstrum can also be derived from linear prediction coefficients using a simple recursion technique, without the need of an inverse Fourier transform [3]. Cepstral coefficients obtained this way are called linear prediction cepstrum coefficients (LPCCs), and have been frequently used as speech features.

A more advanced, and computationally more demanding, method for cepstral features is mel-frequency cepstrum coefficients (MFCCs) [14]. MFCCs have performed well in speech recognition tasks and perhaps been the most popular choice of features in today's state-of-the-

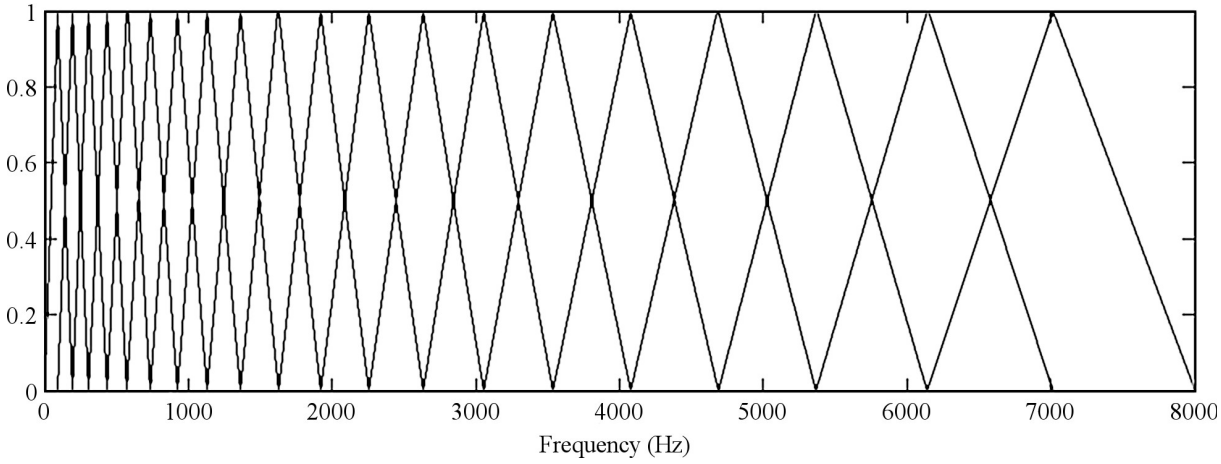

Figure 1: A filter bank consisting of 20 triangular bandpass filters equally spaced along the mel scale.

art recognizers. The novelty in MFCCs is a set of bandpass filters that are applied in the energy spectrum. They are equally spaced along the *mel scale*, which follows the auditory response of the human ear (see Figure 1). Their outputs provide a representation of the spectral content that is compact, but captures the phonetically important characteristics of speech.

MFCC calculation scheme is essentially Fourier transform → square of magnitude → mel scale filters → real logarithm → discrete cosine transform. Discrete cosine transform (DCT) is closely related to Fourier transform and inverse Fourier transform, but operates on real-valued data [1]. Here DCT is preferred over discrete Fourier transform, because DCT has the property of reducing correlation between the features. Uncorrelated features can be efficiently modeled using Gaussian distributions with diagonal covariance matrices.

3.3 Acoustic Modeling

An acoustic model estimates the probability distribution of a series of feature vectors for given word sequence, $P(X|W,\vartheta)$. Assuming the language is not extremely simple, it would obviously be impossible to record an adequate training set for estimating the distribution directly. Instead, it has to be approximated as a product of the probability distributions of some smaller units than the entire word sequence.

The smaller the basic speech recognition unit is, the fewer of them there are in the language, but the more complicated their structure gets. In an application where the vocabulary is limited, whole word units might be used. The symbols that the recognizer outputs would then be natural language words, eliminating the need for lexical decoding. [40]

In large vocabulary speech recognition, the basic unit of sound represented by the acoustic model has to be a linguistically based sub-word unit, such as a phoneme, to keep the number

of different units computationally feasible. This presents the additional problem of lexical decoding, to translate the units of sound to words of the language model and vice versa (see Section 3.6.1).

Phonemes alone seldom produce satisfactory results, because of coarticulation in human speech. The pronunciation of a sound is commonly influenced by the neighboring, conceptually isolated sounds. Thus, larger sub-word units such as diphones, syllables, etc. have been used. However, Schwartz noted that they are just attempts to model the coarticulation effects of adjacent phonemes on each other, but for example a syllable model does not model the effect on neighboring syllables [45].

The Aalto University recognizer uses the *triphone* model that Schartz proposed, perhaps more accurately called a context-dependent phoneme model. A triphone is a triplet of the actual phoneme and its left and right neighbors. The statistical model is just a phoneme model, but trained using triphone occurrences.

There are 8 vowel and 16 consonant sounds in Finnish language, listed in Table 1. For English, the numbers are much bigger and vary from dialect to dialect. All of the Finnish sounds, except /ŋ/, have their own grapheme (the second row of the table). The alveolar nasal /n/ and the velar nasal /ŋ/ are allophones. The *n* in *nk* is pronounced as /ŋ/. The long velar nasal /ŋŋ/ is written *ng*.

Table 1: A list of Finnish phonemes and the corresponding graphemes.

/a/	/b/	/d/	/e/	/f/	/g/	/h/	/i/	/j/	/k/	/l/	/m/	/n/	/ŋ/	/o/	/p/	/r/	/s/	/t/	/u/	/ʊ/	/y/	/æ/	/ø/
a	b	d	e	f	g	h	i	j	k	l	m	n	nk ng	o	p	r	s	t	u	v	y	ä	ö

The 24 different sounds can theoretically be combined in 24^3 ways to form roughly 14 000 different triphones. In comparison, a Finnish syllable model would require around 3 000 statistical models to be estimated [54]. Luckily not all the different triphones are used in the language, and in some cases coarticulation does not have a significant effect. When similar triphones are combined into clusters the number of models can easily be reduced to 3 000 [51].

The statistical model that the recognizer uses to characterize a phoneme is a particular kind of a Hidden Markov Model (HMM) [40], as in the majority of today's speech recognition systems. The sound generation is assumed to be a Markov process with unobserved (hidden) states. Each state is assumed to emit an acoustic observation according to a mixture of Gaussian distributions.

The time-varying nature of an HMM accounts for the variations in the observed feature vectors during the timespan of an acoustic modeling unit. This is important even when the

unit is a single phoneme, since pronunciation changes gradually, not in one step between phonemes. This can easily be heard particularly in diphtongs, for example in the English word low. The two adjacent vowels are pronounced continuously, while changing the shape of the tongue during the pronunciation.

3.3.1 Estimating Model Parameters

The kind of acoustic model described in the previous section is parameterized by the means, variances, and mixture weights of the Gaussian distributions for each HMM state of each triphone, as well as the transition probabilities between the HMM states. Traditionally the parameters are selected so as to maximize the likelihood of the training material, which contains acoustic features extracted from a collection of speech samples, X, and a textual transcription, W:

$$\arg\max_{\vartheta} P(X, W|\vartheta) = \arg\max_{\vartheta_A, \vartheta_L} P(X|W, \vartheta_A) P(W|\vartheta_L) \tag{7}$$

We assume that the first multiplicand, the likelihood of the sequence of acoustic observations given the transcription, is independent from the second multiplicand, the language model likelihood. ϑ includes both the acoustic model parameters ϑ_A, and the language model parameters ϑ_L. When optimizing the acoustic model parameters, we consider only the acoustic likelihood:

$$\arg\max_{\vartheta_A} P(X|W, \vartheta_A) \tag{8}$$

Under certain assumptions, the maximum likelihood estimate is consistent, meaning that having a sufficiently large amount of training data, and assuming that the real probability distribution $P(X|W)$ is indeed representable by the HMM that we use to model it, it is possible to find the correct ϑ_A with an arbitrary precision. In reality, the maximum likelihood estimate results in a suboptimal acoustic model, in terms of recognition accuracy, because

- it is not possible to find a ϑ_A so that the HMM exactly matches the real probability distribution $P(X|W)$,
- there is only a limited amount of training data available, and
- the parameter values are found using a numeric optimization method that converges to a local optimum.

Maximum likelihood training seeks for a model that gives a high probability to the correct hypothesis, without considering other, competing hypotheses. Recognition accuracy can be improved by discriminative training methods that are explicitly designed to discriminate between the correct hypothesis and any other hypothesis. The model parameters are optimized

using an objective function that tries to reduce the probability of incorrect hypotheses, or directly measures the recognition accuracy on the training data. [55]

One possible objective function is the posterior probability of the reference transcription:

$$\arg\max_{\vartheta} P(W|X,\vartheta) = \arg\max_{\vartheta_A,\vartheta_L} \frac{P(W|\vartheta_L)P(X|W,\vartheta_A)}{P(X|\vartheta_A,\vartheta_L)} \tag{9}$$

This is called the conditional maximum likelihood estimate [34]. Again, during acoustic modeling, we consider the language model parameters fixed:

$$\arg\max_{\vartheta_A} \frac{P(X|W,\vartheta_A)}{P(X|\vartheta_A,\vartheta_L)} \tag{10}$$

The criterion then becomes equal to maximizing the mutual information between the two events X and W, and the technique is usually referred to as maximum mutual information (MMI) estimation [6].

Precise computation of $P(X|\vartheta_A,\vartheta_L)$ would involve summation over all the possible hypotheses W_i allowed in the task:

$$P(X|\vartheta_A,\vartheta_L) = \sum_i P(W_i|\vartheta_L)P(X|W_i,\vartheta_A) \tag{11}$$

Enumerating all the possible word sequences is an unfeasible task even with a modest vocabulary. A viable option for small vocabulary tasks is to perform unconstrained recognition on the training data at each iteration of the optimization process to accumulate the necessary statistics [10]. For tasks with a larger vocabulary, it is necessary to approximate Equation (11). Usually all the possible hypotheses cannot be considered, but it is sufficient to consider those that the recognizer could easily confuse with the best hypothesis.

Discriminative training can be applied on large vocabulary tasks, with practical computational costs, using word lattices that encode the most likely recognition hypotheses [52]. Such lattices contain the competing words at different time instances, and their acoustic and language model likelihoods. The structure can be generated as a by-product of the recognition process, and the information can be used to estimate the necessary statistics for adjusting the acoustic model parameters. The same lattices are used repeatedly during the training process on the assumption that the set of likely hypotheses does not change.

3.4 Speaker Adaptation

A major barrier to the wide spread of early speak recognition systems was the laborious training procedure required from each user. The acoustic models were adapted exactly to the char-

acteristics of the speaker's voice. This involved hours of reciting terms from the screen. Furthermore, the systems were often not networked, meaning that users underwent the same procedure on every computer they were to use. [18]

Such strategy is called supervised training. It is easier to implement and more accurate, since the true transcription of the training data is known. It can also hinder the usability too much to be of practical value. Unsupervised training means adapting to the speaker's voice gradually while the speaker is using the application.

Adaptation data first has to be segmented into a sequence of HMM states. If the data has been manually preprocessed, a transcription may include exact timestamps indicating the HMM state positions. If the true transcription is not known, the adaptation data is first recognized, and the segmentation given by the recognizer is used. Adaptation will improve recognition even if the segmentation is not perfect.

There are two approaches to reducing the mismatch between the speaker's voice and the acoustic model. Speaker normalization maps the acoustic features into a feature space that matches the training data of the original model. Model adaptation adjusts the acoustic model parameters to better match the new speaker.

Two popular speaker adaptation methods have been implemented into the Aalto University recognizer: Vocal Tract Length Normalization (VTLN) [11] and Constrained Maximum Likelihood Linear Regression (CMLLR) [16]. Either supervised or unsupervised training can be used with both methods.

VTLN is a simple and robust speaker normalization method for compensating variations caused by differences in vocal tract length. It performs well for adaptation to a different gender or age. It assumes a model of the human vocal tract that suggests a relationship between speaker's vocal tract length, and formants, or resonances in the vocal tract. Only one parameter that describes speaker's relative vocal tract length is estimated from the adaptation data, and the frequency spectrum is warped towards a reference vocal tract length.

CMLLR is a more generic approach for model adaptation to variations both in the speaker's voice and the recording conditions. It applies a linear transformation to the Gaussian distribution parameters of the acoustic model to increase the likelihood of the adaptation data. A linear transform to acoustic features can be found that is equivalent to the model parameter transform [19]. The implementation in Aalto University recognizer uses the latter, speaker normalization approach.

3.5 Language Modeling

A speech recognizer cannot perform well without some model of the target language. The language model and vocabulary should reflect the particular lingo. Without hearing and understanding the context, even humans easily make mistakes when trying to recognize spoken

words. Imagine hearing a word and trying to recognize it, phoneme by phoneme, without knowing the language. Likewise, a person who has no medical background is unable to transcribe medical reports, because such a person is unfamiliar with medical jargon.

The language model limits the recognizer output to only those sentences that are allowed by the language model, and defines the prior probabilities of different word sequences. Depending on the application, one of these functions is usually more important.

Spoken dialog systems have traditionally represented the user interface language using a rigid grammar. Large vocabulary continuous speech recognizers use statistical models that are estimated from a large text corpus. Although simple grammars and statistical models are both manifestations of the rules of the language, their roles are fundamentally different. A rigid grammar restricts to a limited set of allowed sentences, while a statistical model represents the language in a way that is scalable for guiding a word search for large vocabulary speech recognition. [27]

3.5.1 Finite-State Grammars

Finite-state grammars (FSGs) have traditionally been used to describe programming language syntaxes. It was intuitive to use them to describe the languages used in early speech recognition systems as well. The notations used for FSGs are usually based on Backus-Naur Form (BNF). As an example, consider the following, imaginative, very simple command language for dictating patient's dentition status using English keywords.

```
          <command> ::= <s> <tooth-id> <tooth-status> </s>

         <tooth-id> ::= <first-number> <second-number>

     <tooth-status> ::= <tooth-keyword> | <surface-keyword>
                        <surface-number>

     <first-number> ::= “one” | “two” | “three” | “four”

    <second-number> ::= “one” | “two” | “three” | “four” | “five” |
                        “six” | “seven” | “eight”

    <tooth-keyword> ::= “crown” | “pivot” | “loose” | “intact” |
                        “implant”

<surface-keyword> ::= “caries” | “fractured” | “abrasion” |
                        “erosion” | “amalgam” | “hypoplasia”

      <surface-id> ::= “one” | “two” | “three” | “four” | “five” |
                        “six” | “seven”
```

The first rule states that a command consists of the special token <s> that represents sentence start, a tooth ID, a tooth status, and </s> that represents sentence end. Sentence start and end tokens obviously make more sense in continuous speech recognition, where the language

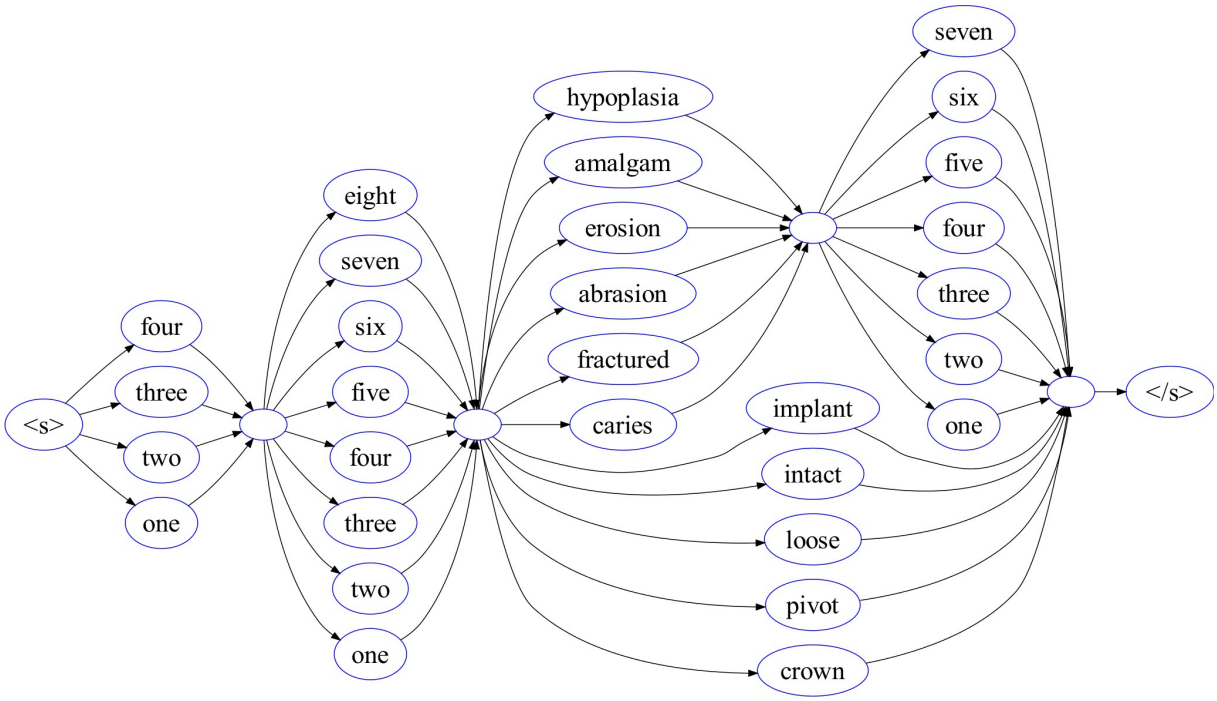

Figure 2: A word network generated from a simple finite-state grammar.

model is used in addition to some acoustic evidence to find sentence boundaries. In simple command dictation application, we might just require that a sentence start token is at the beginning of the audio, and a sentence end is at the end of the audio.

The symbols *tooth-id* and *tooth-status* are nonterminals—similar rules are used to define how they can be expanded. For example, a tooth status consists of either a tooth keyword, or a surface keyword followed by a surface number. A tooth keyword is defined by listing the alternative words of the language. The words are terminal symbols. A word network can easily be generated by replacing all the nonterminals by their definition, as illustrated in Figure 2.

The Harpy system was one of the early speech recognizer, developed at Carnegie-Mellon University in the 1970s, that represented the language using a word network. A word network was constructed from a BNF specification, and the words were replaced by phone subnetworks. The system did a graph search to find the path with the best probability. While the grammar constrained the search, the best path was selected solely based on acoustic transition probabilities. [30]

The downside of an FSG is the computational cost of performing a graph search on a complex word network. The Harpy system was tested with a 1011-word vocabulary language. For command dictation applications, such as voice-activated telecommunication services, the vocabulary was adequate, and the research around such applications concentrated on developing speaker-independent acoustic models rather than larger language models [27].

In practice even the about 10^{12} possible sentences permitted by Harpy's 1011-word vocabulary cannot be searched exhaustively. A contribution of the Harpy system was an approxi-

mate graph search method called beam search. At each iteration, it expands all the neighboring states in a breadth-first manner, but discards the paths whose probability is not close enough to the best path probability so far. Typically vocabulary in a modern continuous speech recognizer is at least an order of magnitude larger, and the search space is more complex, but the beam search principle is still commonly used [33].

3.5.2 N-Gram Language Models

Nowadays statistical language models, n-gram models in particular, have much superseded rigid grammars. At least in large vocabulary continuous speech recognition, n-gram models are most widely used. The research in Aalto University as well has focused on statistical language models.

The concept of an n-gram is simple, but practical estimation of n-gram models, and their efficient use in decoding, is more involved. An n-gram is defined as a subsequence of n words, $w_{k-n+1} \ldots w_k$, from a given sequence $W = \{w_i\}$. The term *word* here denotes the basic vocabulary unit, which could equally well be e.g. a morph.

Analytic languages tend to express only one concept with each word, and have fewer different words, so they can be naturally modeled as a sequence of words. Synthetic languages construct words from sequences of morphs that combine several concepts into one word. The different ways to construct words grow the vocabulary so much that sub-word units have to be used.

An n-gram language model is a probabilistic model that predicts a word $w = w_k$ given the $n-1$ preceding words. The preceding words are called the *word history* and denoted by $H = w_{k-n+1} \ldots w_{k-1}$. For example, the 3-gram *quick brown fox* consists of the word history H = *quick brown* and the word w = *fox*.

The probability of a word sequence can be factored as

$$P(W) = P(w_0)\, P(w_1|w_0)\, P(w_2|w_0 w_1) \ldots = \prod_i P(w_i|w_0 \ldots w_{i-1}) \quad . \tag{12}$$

An n-gram language model approximates the conditional probabilities so that they depend only on their $n-1$ word history:

$$P(w_k|w_0 \ldots w_{k-1}) \approx P(w_k|w_{k-n+1} \ldots w_{k-1}) = P(w|H) \tag{13}$$

3.5.3 Estimating N-Gram Probabilities

An n-gram language model records an estimate for the conditional probability in Equation (13) for every n-gram that appears in the training material. The naive way to estimate the probabilities would be to maximize the likelihood of the training data:

$$P(w|H)=\begin{cases}\dfrac{C(Hw)}{C(H)}, & if\ C(H)>0\\ 0, & otherwise\end{cases} \tag{14}$$

where $C(Hw)$ is the number of occurrences of the n-gram Hw, and $C(H)$ is the total number of occurrences of n-grams whose history is H, in the training data.

The maximum likelihood estimate gives zero probability to n-grams that never occur in the training data. In reality the training data never covers all the conceivable n-word sequences. Large vocabulary language models address the problem using what is called *smoothing*, by distributing some of the probability mass from the other n-grams to those that have zero probability.

The free dictation language models in this project were created using the Kneser-Ney smoothing method [28]. The probabilities of unseen n-grams are estimated using lower-order n-grams. Lower-order n-grams are more likely to occur in the training data. Several smoothing methods have been proposed that use the same principle. Let $\hat{H}$ denote the lower-order word history, i.e. the $n-2$ words preceding w, and $H=h\hat{H}$ denote the $n-1$ preceding words. The exact way in which the lower-order estimates are calculated in Kneser-Ney smoothing has been selected so that marginalizing over higher-order word histories equals to the lower-order maximum likelihood estimate from the training data:

$$\sum_h P(w|h\hat{H})=\frac{C(\hat{H}w)}{C(\hat{H})} \tag{15}$$

A constant value D is subtracted from the counts of the n-grams that occurred in the training data, i.e. the probability of a seen n-gram is calculated as

$$P(w|H)=\frac{max(C(Hw)-D,0)}{C(H)}, \quad if\ C(Hw)>0 \quad . \tag{16}$$

These constraints result in lower-order estimates that reflect the number of different words *h* that occurred before the lower-order history $\hat{H}$. Finally, the probabilities of unseen n-grams are normalized to make the conditional probabilities sum to one:

$$\sum_w P(w|H)=1 \tag{17}$$

The intuitive description of Kneser-Ney smoothing is that if an n-gram such as *slow brown fox* is not seen in the training data, its probability is estimated from the number of different words preceding *brown fox*. This gives a better estimate than the frequency of *brown fox* in the training data, since if *brown fox* is used frequently, but only in the context *quick brown fox*, it would indicate that *slow brown fox* should receive a low probability.

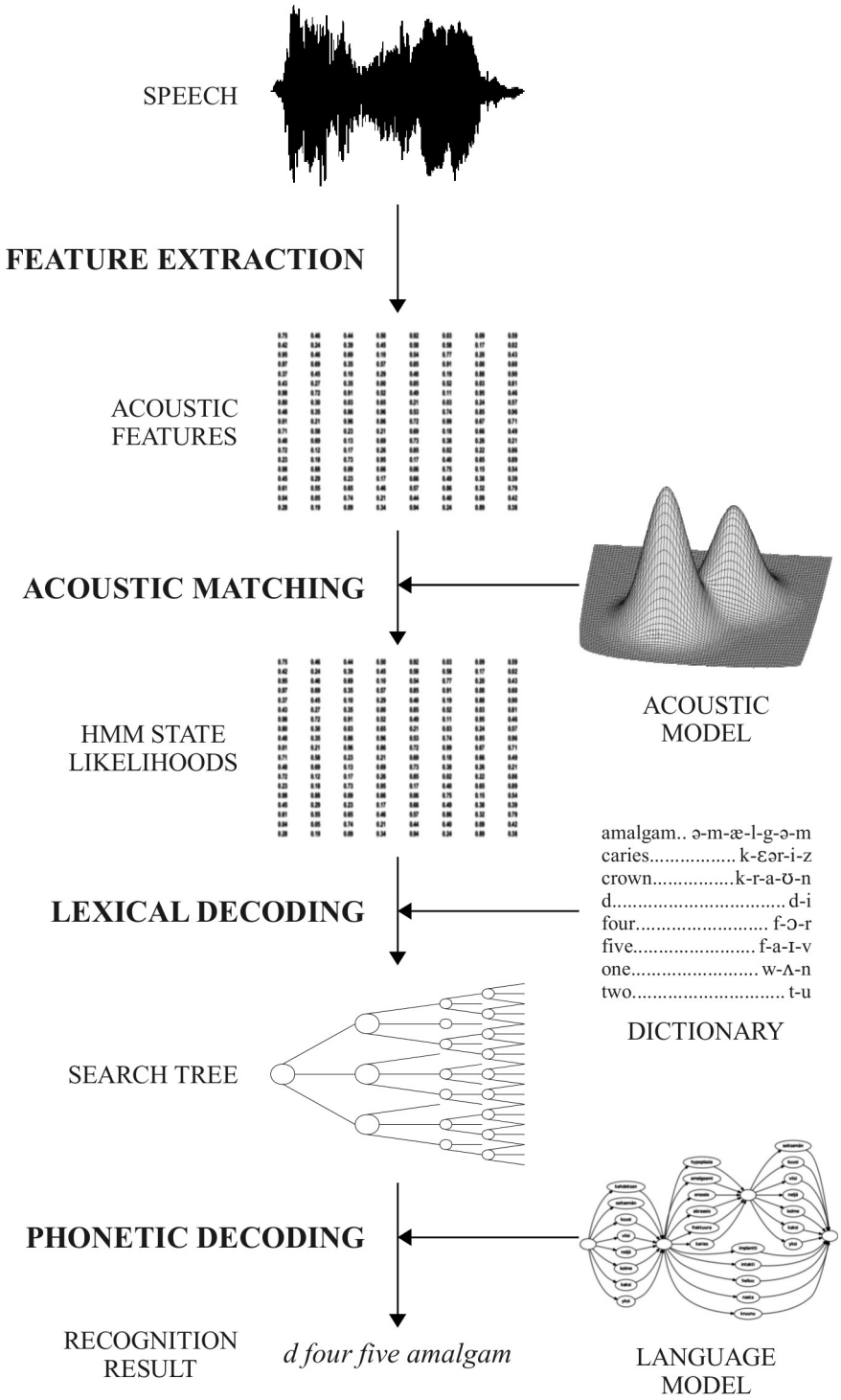

Figure 3: Block diagram of a speech recognizer.

3.6 Decoding

The actual recognition task is called decoding. As explained earlier, a statistical model is used to describe each speech recognition unit. For brevity, we will now assume that the speech recognition unit is a phoneme. During training, the model parameters are optimized so as to

best describe the training data. When the model parameters have been fixed, the recognizer can evaluate how well a sequence of observations matches each phoneme model.

The recognition process is described in the block diagram in Figure 3. Phonemes are typically modeled as an HMM, and the state likelihoods are obtained by matching the acoustic features to the phoneme models. Without any other knowledge sources, the recognizer would simply select the most likely phoneme, according to the acoustic model, at each time instance. However, the language model introduces additional knowledge that should be considered when deciding the best phoneme sequence.

Conceptually, all the possible phoneme sequences form a search tree. First, *lexical decoding* places such constraints on the search tree that the phoneme sequences correspond to words in the dictionary. Then, either the word sequences are restricted to only those allowed by a rigid grammar, or a statistical language model gives the prior probability of each word sequence.

3.6.1 Lexical Decoding

Since the recognition is based on phonemes, the recognizer needs to translate the words in the language model to phonemes, and the recognized phonemes back to words of the natural language. This mapping of words to phonemes is called a lexicon or a dictionary. For English language, we have no other option than to manually define the pronunciation of each word. In the Finnish writing system there is generally exactly one letter per phoneme and vice versa, which makes it trivial to specify a lexicon.

The majority of letters represent distinct phonemes in Finnish language. The alphabet contains also foreign letters that are not used in Finnish language. When creating the lexicon, these are mapped to the Finnish phonemes that they are considered to represent.

There are also some letters whose pronunciation changes systematically depending on the letters that occur before or after them. For example, the letter *n* is usually pronounced as the /n/ sound, but in *nk* and *ng* it is the velar nasal (the /ŋ/ sound). These exceptions do not need to be handled explicitly when using triphones, since a different phoneme model will be selected depending on the triphone context. The *n* letter, for example, in the *nk* or *ng* context has a statistical model that describes the /ŋ/ sound, but in other contexts its model describes the /n/ sound.

The dictionary is often represented as a prefix tree [33]. Edges of the tree represent phonemes, and each path through the tree determines the phonetic composition of a dictionary word. A tree representation is more compact than a flat table, since many words share the same beginning. Lexical decoding time is reduced as the shared part needs to be computed only once. However, the word identity is not known until a leaf of the tree has been reached, delaying the application of language model probabilities.

3.6.2 Search Strategies

A decoder should extract the most likely word sequence, given the acoustic and language model constraints. The theoretical number of different word combinations in a sentence of any natural language is astronomical. A large vocabulary decoder needs optimization techniques and heuristics to be of practical use.

Decoders exploit various language model properties to build as compact representation of the search space as possible [5].

- The scope of a statistical language model is typically short. Often a trigram language model is adequate, meaning that only the last two words have influence on the language model probability of the next word. Whenever there are multiple search paths that share the same two consecutive words, the decoder knows that the paths will have identically scored extensions, and can select the best one.
- N-gram language models are very sparse, and often contain redundancies. For example, only a fraction of all the possible n-grams occurs in the training data—the rest will be given a probability by a smoothing scheme, and do not have to be stored individually in the search tree.

Even after the search space has been optimized, an exhaustive search on a large vocabulary is impossible. A decoder has to rely on heuristics to discard the most unpromising paths in an early stage, so it cannot be guaranteed that the optimal solution is found. In research systems such search approximations typically have only minor effect on the overall recognition accuracy, but in real-time applications coarser approximations are needed, and decoding errors become a significant factor.

3.7 Measuring Recognition Accuracy

Accuracy of a recognition result is assessed by comparing the recognition to a reference transcription. In addition to recognizing words incorrectly, the recognizer may add extra words, and some words may be missing altogether. Commonly error measures are derived from the Levenshtein distance, i.e. the number of editing operations required to transform the recognition result into the reference transcription.

The most widely used metric for measuring speech recognition accuracy is the word error rate (WER). It is defined as the minimum number of word substitutions (S_w), deletions (D_w), and insertions (I_w) that are needed to correct the result, relative to the total number of words in the reference transcription (N_w):

$$WER = \frac{S_w + D_w + I_w}{N_w} \tag{18}$$

Letter error rate is a similar measure, but defined using the minimum number of letter substitutions (S_l), deletions (D_l) and insertions (I_l) required to correct the result:

$$LER = \frac{S_l + D_l + I_l}{N_l} \tag{19}$$

N_l is the total number of letters in the reference transcription.

When recognizing analytic languages, word error rate can be considered as the relative number of errors in the information content. Finnish is a synthetic language, meaning that it often combines multiple concepts into one word. If the recognizer makes a small error in one morpheme of a long word, not all the information conveyed in the word is lost. In such a case, letter error rate is a more accurate error measure, whereas word error rate considers the whole word incorrect [21].

An automatic speech recognizer inevitably makes some mistakes when recognizing free speech, and the user has to correct the mistakes using a keyboard. Assuming that the user is able to insert, delete, or substitute characters, letter error rate tells the exact number of editing operations the user has to make, relative to the number of characters in the spoken text.

Certainly a modern text editor is more flexible, but letter error rate gives a good approximation of the work needed to correct recognition errors. According to the definition, letter error rate may be more than 100 %. In that case, the user would need more editing operations after dictation, than it would have taken to type it using a keyboard right from the beginning.

When assessing the performance of free speech dictation, letter error rate can be compared to 100 %, at which the system would be totally useless. The results from this project can also be compared to previous results from the MobiDic project [31]. Without any adaptation, the letter error rate at laboratory experiments on Finnish text was 4.6 %. Speaker adaptation reduced the letter error rate to 4.0 %, but language model adaptation did not show significant improvement in the error rate.

In medical reports some errors are more fatal than others. Errors in numerals are hard to spot afterwards, and could be particularly harmful, as we know that in an extreme case poor communication among clinicians can even lead to the extraction of a wrong teeth [37]. In this thesis we paid particular attention to the recognition of numerals in addition to the overall letter error rate.

Letter error rate is not adequate for measuring errors in dentition status dictation. The dentition status language is not synthetic in the sense that there is a limited set of allowed words that are not related to each other morphologically. Also, the user is only concerned of whether a command is recognized correctly or not, so from a functional perspective, a proper error measure would be the rate at which commands are recognized incorrectly. Command error rate is the ratio of the number of erroneously recognized commands (E_c), and the total number of commands given (N_c):

$$CER = \frac{E_c}{N_c} \tag{20}$$

Insertion and deletion errors are not of concern, since it is the responsibility of the voice activity detector to detect where a command starts and ends. Command error rate could be too insensitive, however, to notice any advantage or disadvantage when the system is adjusted, so word error rates were recorded as well.

3.8 Assessing Confidence on Recognition Result

Spoken dialog systems need to measure their trust in the correctness of the recognition result. If a command is recognized with low confidence, the system can ask the user to repeat it. Another application that benefits from confidence assessment is unsupervised speaker adaptation. The recognizer can select the voice segments with highest confidence as adaptation data.

In this project a confidence measure was needed for dentition status dictation, to decide whether an utterance was intended as a command or not. The user does not need to switch the recognizer off while talking to the patient, but the system simply ignores utterances that were not recognized as a command with high enough confidence. The confidence measure is ignored when the recognizer is used for report dictation.

In theory, the posterior probability of a word sequence given the acoustic features, $P(W|X,\vartheta)$, would conveniently indicate the confidence on the recognition result. However, as explained in Section 3.1, the decoder decides the most probable word sequence without computing the common normalizing term, $P(X|\vartheta)$. As discussed in Section 3.3.1, precise computation of $P(X|\vartheta)$ is unfeasible for a large vocabulary task. There are methods for calculating confidence as an estimate of the posterior probability, but they either impose certain assumptions, or adopt some approximate methods [26].

Several heuristic features have been proposed for confidence calculation that can be derived from the list of *n* most likely hypotheses considered by the decoder. The n-best list can be assembled from the recognition lattice, which the recognizer can save without additional computational costs. These heuristics usually rely on the assumption that if the utterance is recognized correctly, the likelihood of the best hypothesis is clearly higher than that of the other $n-1$ hypotheses. Commonly used predictor features can be found in [17] [44].

Our experiments on measuring confidence on dentition status commands are presented in Section 5.3.2. Better confidence measures could probably be obtained using another language model that allowed sentences outside the command grammar. The strict language model would still be used for the actual recognition. A big difference between these two recognition results would indicate that the utterance is not a valid command.

4

Experimental Setup

4.1 Aalto University Speech Recognizer

The speech group in Aalto University has developed several C++ libraries that provide all the necessary functionality for automatic speech recognition. The libraries have not been ported to any other operating system than Linux. A modern speech recognizer is a complicated system. In order to function it requires two statistical models, one that models the acoustics of human voice, and one that models the statistics of the language.

Recognition is always specific to a particular language. Even within a language the vocabulary differs substantially whether the speaker is e.g. dictating a patent application or ordering a hamburger. For good results, a language model specific to the context, or field of expertise, is needed. The acoustic model is somewhat sensitive to variations in speaker voice and the acoustic environment, although using versatile training material it is possible to build a generic but less accurate model.

Vast amounts of textual material is needed to train a language model, and many hours of speech recordings along with their transcriptions are needed for the acoustic model. In addition, there are numerous parameters that have to be adjusted. They are often a compromise between two competing factors, such as recognition accuracy and computational burden. Some choices are rather heuristic based on trial-and-error. Furthermore, all the parameters are related to each other in ways that are not so obvious. Although good results have been achieved in various benchmarks, there is a wide gap between the research system and a commercial application.

Currently Aalto University speech recognizer is being integrated into the speech recognition server used in the MobiDic project [50], while the client application was developed in the

University of Tampere. The client application runs on a mobile phone, and is designed for dictating notes while traveling. It records speech, sends it to the recognition server, and receives the textual result from the server. The service is adapted to specific user groups, lawyers in its first implementation, by acoustic and language model adaptation. User experiences have not yet been received.

The recognizer will also be used in the recently started Mobster project, where research for mobile dictation and communication in health services domain is being carried out. In contrast to this project, Mobster is a more general solution for various user scenarios, and used in extremely noisy environments. Mobster project also has less training data available.

Details about the recognizer can be read from [22].

4.2 WinHIT Information System

The speech recognition functionality is integrated into WinHIT information system, developed by In Net Oy. The system includes a Windows user interface, and a database server. WinHIT has been developed specifically for dental care, and embodies a large scope of functionalities, including

- customer information management,
- patient information management,
- work time scheduling,
- invoicing, and
- reporting.

Patient information in WinHIT includes

- the anamnesis (medical history) gained by asking specific questions of the patient before the treatment,
- treatment reports, plans, and background information gathered by the dentist during the patient's visit, and
- patient's periodontal and cariological status.

The purpose of this project is to examine how automatic speech recognition could assist the dentist in entering the information into the system. Specifically, in the scope of this project, we have applied speech recognition to two tasks:

1. The dentist is able to dictate the status of the patient's dentition, while treating the patient, into a microphone.
2. The system will automatically transcribe reports that dentists and radiologists dictate using a microphone, after the patient's visit.

4.3 System Architecture

The recognizer is running on a server computer, or as a background process in a desktop computer or laptop. Running both the recognizer and the user interface in a single computer is somewhat more complicated from the software perspective, because the user interface is a Windows application, and the recognition server is a Linux application.

It is still possible to share the same computer hardware between the two applications through virtualization. A host software can be installed into a Windows computer that allows the execution of Linux operating system in a virtual machine. Modern desktop computers have adequately processing power for real-time speech recognition even in a virtual machine.

The system architecture is depicted in Figure 4. WinHIT workstation software records dentist's speech from a microphone input, and sends the audio waveform in a recognition request to the recognition server. The server will return the result as text. The recognition server and the user interface will communicate with each other through a network interface, so that it is possible to use a dedicated server or a virtual machine.

A network protocol used for communication between WinHIT and the recognizer is described in Appendix A. WinHIT detects voice activity and sends the first audio packet to the

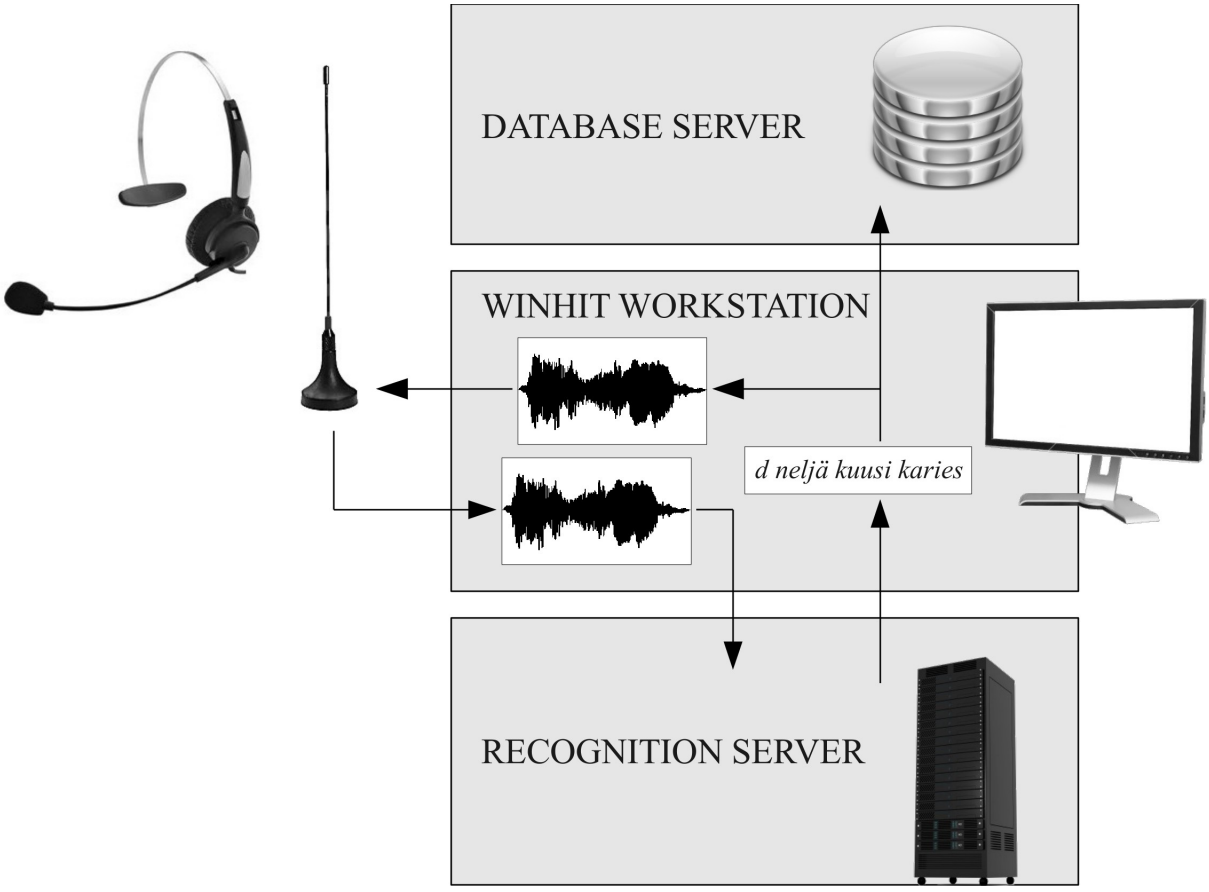

Figure 4: The system architecture.

recognizer immediately when enough audio has been recorded. When WinHIT detects that voice activity has ended, it sends an empty audio packet that indicates the end of audio.

The server needs a way to assess its confidence in the recognition result. If the confidence is lower than a threshold set by WinHIT, the server will return a *no-match*; otherwise the recognition result is returned. When dentition status dictation is active, the user does not need to switch it off in order to talk to the patient or other personnel—when the user says something that is not a proper status command, recognition confidence will be low, and WinHIT will ignore the utterance.

4.4 Audio Input and Output

The dentists are supposed to dictate the status of the patient's dentition using a wireless headset, which also gives feedback so that the dentist can be sure that the commands are recognized correctly. The low-end wireless headsets in the market generally use either Bluetooth or DECT technology. There are some consumer headsets that advertise high-quality sound output, but the models with high-quality microphone input have been designed for professional audio engineering applications, and cost considerably more.

Because of the very strict grammar used in dentition status dictation, the sound quality does not have to be as high as in free speech recognition. On the other hand, treatment reports can be dictated using a wired microphone, since the dentist is sitting in front of a computer. As a compromise, a DECT headset was selected for dentition status dictation, and a separate microphone is connected to a desktop computer for patient treatment report dictation.

4.4.1 Audio Input and Output for Dentition Status Dictation

DECT has been standardized to use the ITU-T G.726 audio codec [56]. The audio is sampled using 8 kHz sample rate and 8 bit sample size, and compressed using adaptive delta pulse code modulation (ADPCM) to 32 kbit/s net bit rate. The sound quality is noticeably poor, but was found it to be sufficient for the small vocabulary task—all the commands in our test set were recognized correctly. The commands are repeated from the headset so that the user knows whether they are correctly recognized or not.

4.4.2 Audio Input for Free Speech Dictation

For free speech dictation a wired microphone can be used, and provides high sound quality at low price. Practically all modern sound cards support sample rates up to 44.1 kHz, but the recognition does not benefit from frequencies above 8 kHz, so we use 16 kHz sample rate. The user follows the recognition from the computer screen, and there is no need for repetition from speakers.

4.5 Acoustic Model

The acoustic model that was used in our experiments was trained using about 63 hours of material from the Finnish language speech corpus of the SPEECON project [24]. 510 of the 550 adult speakers were used for training, of which half were male and half were female. In total, 38,072 utterances were used, consisting of sentences, word sequences, and some individual words.

The speech samples have been recorded at 16 kHz sampling rate and 16 bit quantization. The recordings have been captured simultaneously using four microphones at different distances from the speaker's mouth. Two microphones place at close distances, and one microphone placed at 0.5–1 meter distance, were used for training. Only one recording was used per utterance, biasing towards those recordings that had higher signal-to-noise ratio.

The model parameters were estimated using a discriminative training scheme. The parameters were first initialized using maximum likelihood estimation. Then maximum mutual information estimation using word lattices was applied to adjust the parameters [55].

There was no acoustic training data from dentists. Significant performance improvements could be gained with adaptation of the acoustic model to the specific speaker. In a similar setting, a word error rate reduction from 21.2 % to 17.5 % was reported with 20 adaptation sentences, using the CMLLR method [31].

4.6 Voice Activity Detection

We have implemented a voice activity detector (VAD) that is used both in status dictation and free speech dictation for different purposes. When dentition status dictation is active, the software needs to locate the start and end of status commands from the audio stream. Background noise such as sounds from outdoors or music from a radio receiver may be present, and the dentist may discuss with the patient and other personnel between the commands.

We assume that the dentist waits a few seconds before and after issuing a command. The audio stream can then be segmented based on signal level. Otherwise speech recognition would need to be continuously active, and the software would need to locate valid commands from the recognized text stream. The assumption is fair enough not to reduce usability, and allows a considerably simpler and more robust implementation.

Free speech dictation would not necessarily need voice activity detection, since the user activates it from the graphical user interface. However, it may take ten minutes to dictate one report, and it is good for the robustness of the recognition to split the audio into more easily digestible pieces. While dictation a report, the VAD splits the audio stream into segments on pauses, and the segments are recognized separately.

Another reason for splitting the audio stream is that the user has to get feedback while the recognition is active. The recognizer could send partial results to the user interface as soon as they are guaranteed to remain unchanged, but it would make the communication protocol more complicated.

The audio stream is split into segments of continuous speech and the silence between them is discarded. The speech segments are recognized separately. In status dictation mode, a confidence measure is used to distinguish between dictation commands and other conversations the dentist may have. Speech segments with low confidence are simply ignored.

Voice activity detection has commonly been used in combination with some noise reduction technique to allow estimation of noise statistics during speech pauses [29]. Another common application is in speech coding, where bit rate can be reduced in the absence of noise [48], reducing both traffic on the packet network and the power consumption of the device. Such approaches have typically focused on derivation of noise robust features and decision rules for classification of short audio segments as either voice or non-voice—they are not concerned on how long a pause between speech segments is. The features are often derived from a spectral representation of the audio signal.

We wanted to perform voice activity detection in the client software to reduce network usage and server computation. We wanted to avoid complicated feature calculations, since the client does not have such signal processing arsenal readily available as the recognizer. Luckily, the system is normally used in a relatively noiseless environment. The single loudest background sound source is typically a radio, but the headset will capture the dentist's voice at a higher level. Thus a simple amplitude-based feature is sufficient to separate the dentist's voice from the background noise.

Our algorithm needs to distinguish command boundaries from short pauses within a command. In addition to inter-word silences, short pauses often occur between two consonants. Similar algorithms have been developed for segmentation of long unbroken audio streams, such as news broadcasts [38]. Our algorithm detects command boundaries using the following parameters.

- max_pause_time maximum within-command pause time
- $adjust\_interval\_voice$ time to wait before readjusting the threshold on a voice segment
- $adjust\_interval\_nonvoice$ time to wait before readjusting the threshold on a non-voice segment
- $min\_threshold$ minimum threshold value
- $max\_threshold$ maximum threshold value

The signal is processed in non-overlapping frames of 80 samples. Only the average signal level is considered for each frame. Average signal level raising above a threshold value starts a voice segment. Average signal level dropping below the threshold starts a pause segment.

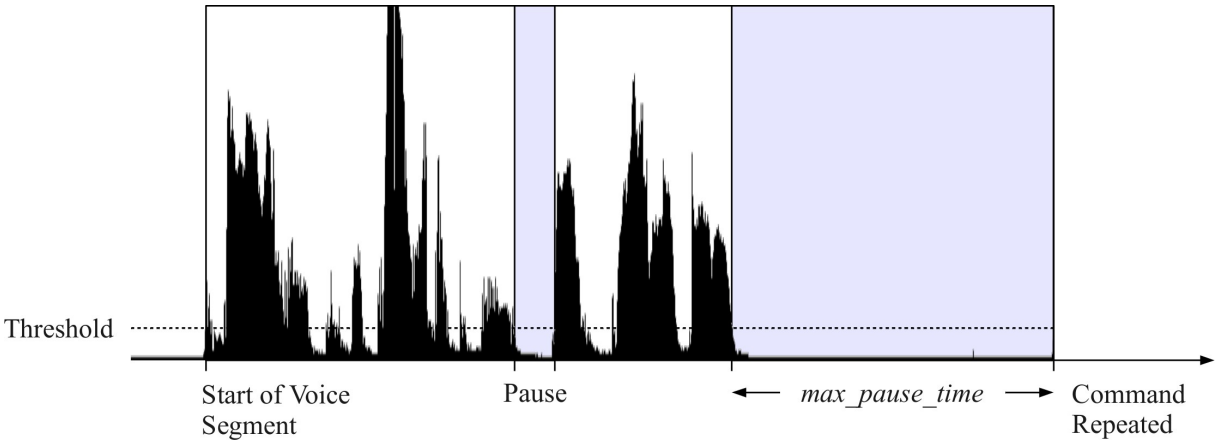

Figure 5: Voice activity detector waits until the average signal level is below the threshold for the time specified by max_pause_time. The command is then repeated through the headphones.

The system waits for max_pause_time at pause state, before assuming that the entire command has been received. max_pause_time should be longer than the maximum within-command pause length. There is no strict upper limit, as long as the user will not get frustrated, since the user always waits quietly for the command to be repeated through the headphones before issuing the next command (see Figure 5).

Voice level depends substantially on the speaker, the type of microphone used, and the placement of the microphone. Noise level may vary throughout the recording in the presence of non-stationary noise sources, such as a radio. For those reasons we did not want to fix the threshold value. The threshold is adjusted dynamically at pre-specified intervals, using the average signal level from the previous interval ($average\_level$) as a reference.

At a voice segment, the threshold is adjusted every $adjust\_interval\_voice$ milliseconds. The new threshold is the average of $average\_level$ and the current threshold value:

$$threshold \leftarrow \frac{threshold + average\_level}{2} \tag{21}$$

Since at a voice segment the signal level is above the threshold, this gives a new threshold that is closer to the average signal level. The idea is that the signal staying long enough above the threshold may indicate that the threshold is below average noise level, and should be raised.

Outside a voice segment, the threshold is adjusted every $adjust\_interval\_nonvoice$ milliseconds. The new threshold is double the average signal level from the previous interval:

$$threshold \leftarrow 2 \times average\_level \tag{22}$$

Outside a voice segment the average signal level represents the average noise level, and this gives a heuristic estimate for a good threshold.

Whenever threshold is adjusted, minimum and maximum values are enforced:

$$\begin{aligned} threshold &\leftarrow min(threshold\,, max\_threshold) \\ threshold &\leftarrow max(threshold\,, min\_threshold) \end{aligned} \tag{23}$$

Utterances starting with some consonant sounds, such as /p/, were problematic for the threshold-based voice activity detector. The start of the speech segment was often detected late, because the first sound was too quiet. The solution was to include the 0.5 seconds of audio immediately before the detected start boundary to the speech segment. Possible short silence in the beginning of a speech segment does not affect the recognition result.

4.7 Optimizing Acoustic Likelihood Computations

The recognizer processes audio one frame at a time. First it compacts the audio frame into a feature vector. For each HMM state of each triphone, the recognizer needs to calculate the likelihood of the feature vector, assuming the probability distribution of the given triphone state. This can be so time consuming, that it cannot be performed in real-time, without optimizations.

By merging triphone state distributions into cluster distributions, the computational efforts can be greatly reduced with only a small effect on recognition accuracy. Another benefit of clustering is that some triphone states are quite similar, and when there is a limited amount of training data, clustering can improve recognition accuracy.

When the acoustic model is trained, the Gaussian triphone state distributions are clustered. The distributions in each cluster are merged into a Gaussian distribution at the cluster center. When computing the likelihoods of acoustic features, only a portion of the distributions are evaluated exactly. For the rest, the triphone state distribution is approximated with the closest cluster distribution. The likelihood has to be computed only once for the set of distributions in each cluster.

Table 2 shows the effect of reducing the fraction of Gaussians that are evaluated exactly, on computation times and error rates. Acoustics time measures the CPU time used computing the triphone state likelihoods, in minutes. Decode time measures the CPU time spent decoding. Total time is the sum of these two values. The tests were performed on a collection of 52 minutes of free speech medical dictations. A desktop computer with Intel Core 2 Quad Processor at 2.83 GHz clock rate was used.

50 % of the Gaussian distributions can be approximated without any loss of accuracy, reducing acoustics time by 40 %. When more Gaussians are approximated, the error rates start to increase, and there is also a slight increase in decode time. The increase in decode time

Table 2: Effect of approximating Gaussian distributions with cluster distributions. The first column is the percentage of Gaussian distributions that were evaluated exactly. The rest were approximated with a cluster distribution.

Gaussians	Acoustics time	Decode time	Total time	LER	WER
100%	200 min	33 min	233 min	7.15%	18.8%
50%	119 min	33 min	152 min	7.15%	18.8%
15%	46 min	37 min	83 min	7.32%	19.2%
10%	35 min	40 min	74 min	7.37%	19.1%
5%	23 min	45 min	68 min	7.24%	19.4%
0%	10 min	108 min	118 min	10.6%	24.8%

could be explained by the decoder not being able to discard bad hypotheses, when many hypotheses receive the same acoustic probability.

On this specific platform, only some 15 % of the Gaussians can be evaluated exactly, in order to compute the likelihoods in real-time. Combined acoustics and decode time is more than 52 minutes, but these operations can be performed in parallel with a multi-core processor. Dropping the fraction to as low as 5 % still does not considerably increase the error rates.

4.8 Optimizing Decoding Parameters

The two most important heuristic parameters that control the decoder are beam width and language model (or acoustic model) scale factor. Both parameters have great impact on decode time and accuracy. They are not independent of each other, but need to be optimized together.

The decoder uses beam pruning, that is, at each iteration it discards the paths whose probability, P_i, is under a percentage of the best path probability so far, P_1:

$$\frac{P_1}{P_i} > e^{\beta} \tag{24}$$

The threshold for discarding a path is called beam width. Beam width values in this section are logarithmic, and denoted by β. By reducing beam width, decoding speed can be increased on the expense of decoding accuracy.

Best recognition performance is not achieved by combining the acoustic and language model likelihoods directly as in Equation (3). $P(X|W,\vartheta_A)$ and $P(W|\vartheta_L)$ are only approximations of the true probability distributions, and the parameters ϑ_A and ϑ_L are estimated independently using different knowledge sources [25]. In practice, the likelihoods from an HMM-based acoustic model dominate the decision between competing hypotheses over the language model likelihoods [53].

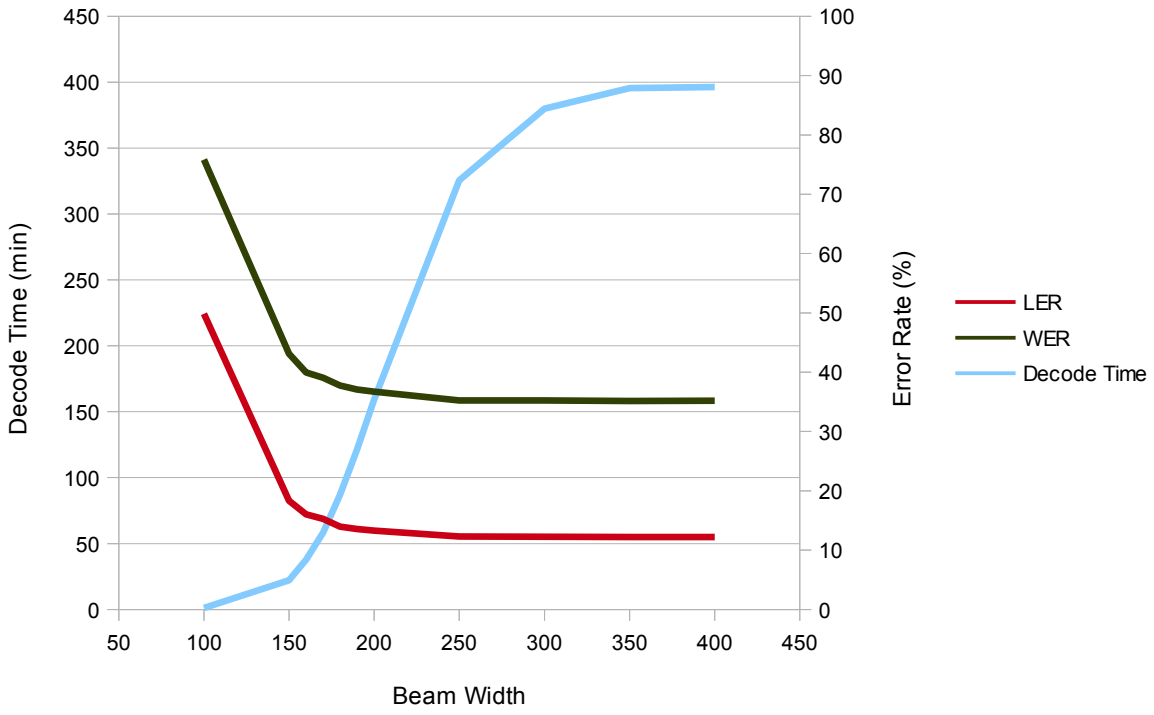

Figure 6: The effect of beam width adjustment on total decode time and error rates on 52 minutes of medical dictation, using language model scale factor 30.

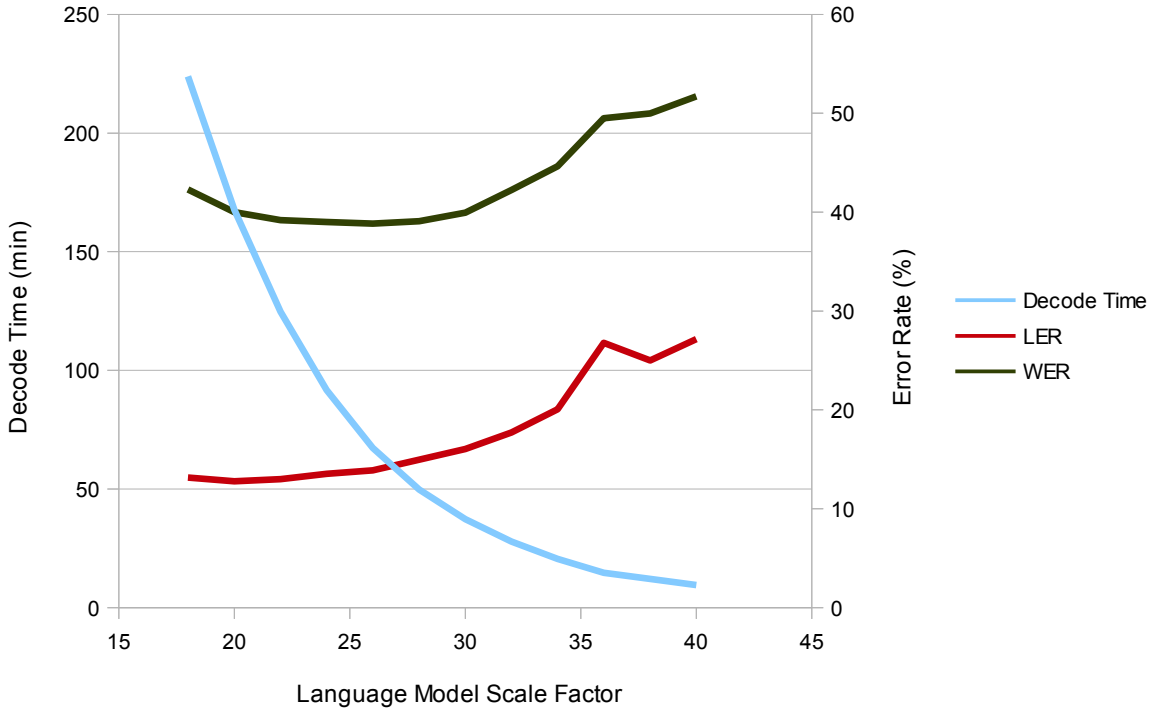

Figure 7: The effect of language model scale adjustment on total decode time and error rates on 52 minutes of medical dictation, using beam width 160.

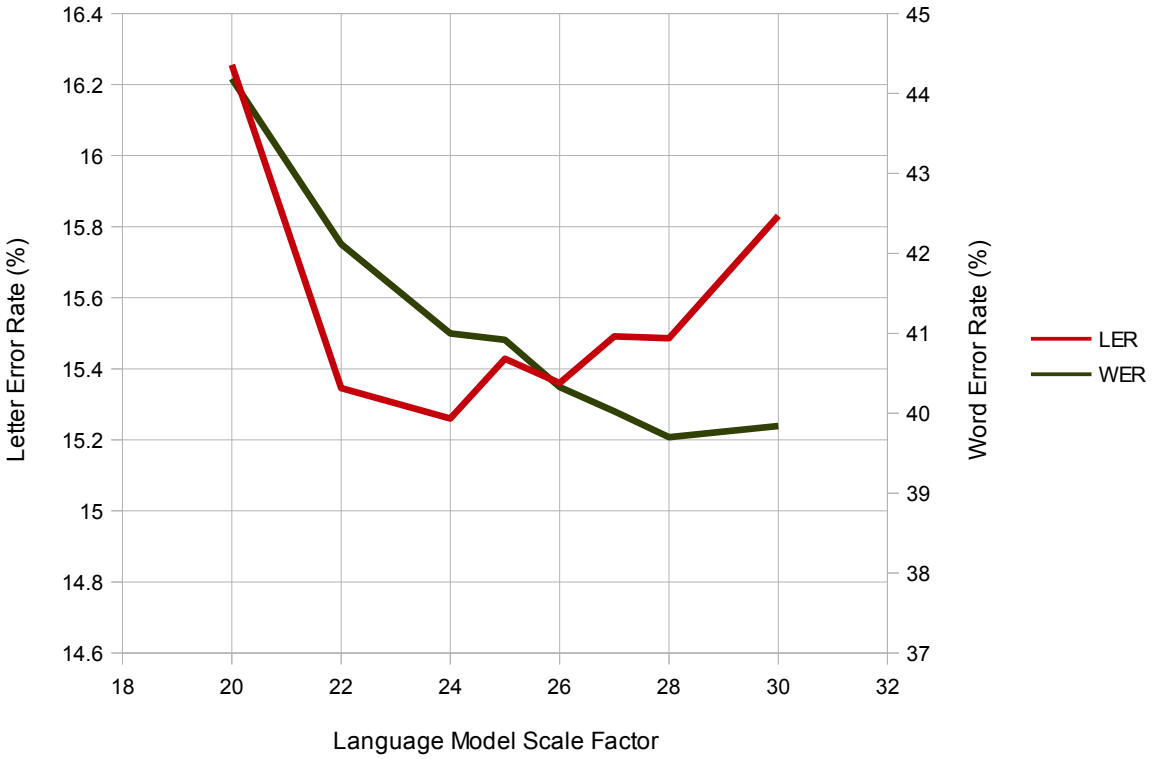

Figure 8: Error rates for different language model scale factors, when beam width is adjusted so that decode time stays at approximately 41 minutes.

The usual solution is to raise language model likelihoods to a power $\alpha>1$. The decision rule becomes

$$\arg\max_{W} P(W|\vartheta_L)^{\alpha} P(X|W,\vartheta_A) \quad . \tag{25}$$

The exponential scaling factor increases the relative distance between likelihood values. Increasing the gap between the language model likelihoods of competing hypotheses increases the contribution of the language model to the decisions made by the decoder. Another possibility is to raise acoustic likelihoods to a power $\gamma<1$.

The effect of beam width adjustment on decode time and error rates, using language model scale factor $\alpha=30$, is shown in Figure 6. The tests were performed on a collection of 52 minutes of free speech medical dictations. A desktop computer with Intel Core 2 Quad Processor at 2.83 GHz clock rate was used.

Decode time measures the so-called processor time, i.e. the amount of time that the decoding process is actively using the CPU. This gives an upper limit on the actual decode time on an identical system. Real time recognition on a similar system would naturally require a decoding time shorter than 52 minutes. A proper beam width in this case would be very close to

Table 3: Rows two to five: Language model scale factor, beam width, and decode time in minutes for several points that were used to estimate a polynomial model. Rows six and seven: The first iteration of polynomial regression (some values are missing). Rows eight and nine: The values that give the decode time closest to 41 minutes.

	LM scale	18	20	22	24	26	28	30	32	34	36	38	40
Constant beam	Beam	160	160	160	160	160	160	160	160	160	160	160	160
	Time	224	168	125	92	67	50	37	28	20	15	12	9,5
Educated guess	Beam	100			130	140	150	160		180	190	200	210
	Time	8,3			19	24	30	37		54	62	72	81
Polynomial regression	Beam		118	128	137	147	155	163					
	Time		N/A	N/A	N/A	N/A	N/A	42					
Optimal	Beam		129	136	143	149	156	162					
	Time		41	41	41	40	42	41					

$\beta=160$. Higher values make recognition unfeasible in real time, and lowering the value starts degrading recognition accuracy fast. This suggests that decoding parameters should always be selected carefully.

The effect of language model scale adjustment on decode time and error rates, using $\beta=160$, is show in Figure 7. Scaling the language model likelihoods with $\alpha>1$ also has the effect of bringing the values of the discriminant function in Equation (25) further apart from each other. If the beam width is kept constant, less hypotheses will fit inside the beam, and decoding is faster but less accurate.

Thus when optimizing recognition performance, we would like to adjust both parameters at the same time, so that decode time does not change. In practice it is difficult to find the exact relationship between the parameters and decode time. For a starting point, we fit the decode time t , language model scale factor α , and beam width β , into a polynomial model:

$$t=a_0+a_1\alpha+a_2\beta+a_3\alpha^2+a_4\beta^2 \quad , \tag{26}$$

where $\{a_i\}$ are the model coefficients. This allows us to estimate the correct beam width for given language model scale factor, when the decode time is fixed:

$$\beta=\frac{-a_2\pm\sqrt{a_2^2-4\,a_4(a_0-t+a_1\alpha+a_3\alpha^2)}}{2\,a_4} \tag{27}$$

Our goal was to find, for each language model scale factor, the beam width that gives the decode time closest to 41 minutes. This process is summarized in Table 3. We used an educated guess to come up with beam width values that would give a decode time close to the target (rows four and five). We also included the data from the previous experiment with beam width $\beta=160$ (rows two and three). These points were used to calculate the model coefficients. The beam width values predicted by the polynomial model are on row six. The new

measurements were used to calculate a better polynomial model. After a few iterations, convergence slowed down and we needed to adjust the beam widths by hand to find exactly those that are closest to the target (rows eight and nine).

The behavior of the error rates, when decode time is kept constant at 41 minutes, is seen in Figure 8. Letter error rate reaches its minimum at $\alpha = 24$. A slightly better word error rate would be achieved at $\alpha = 28$, but as explained in Section 3.7, letter error rate is a better error measure.

5

Dentition Status Dictation

This project needs two different language models: a grammar for dictating dentition status commands, and a free speech model for medical transcription. This chapter describes the grammar for dentition status commands, and the free speech model is described in Chapter 6.

5.1 Dentition Status Commands

The language for dentition status dictation includes commands for making simple notes that the dentist has observed about specific teeth or tooth surfaces. There is a predefined set of descriptive terms that the dentist can use, but the terms cover exhaustively the cariological and periodontal conditions that are commonly encountered, such as caries, crowns, bridges, implants, fractures, and root canal treatment.

Every command contains a keyword that identifies the desired action or marking to be made. Most commands also require the target of the action to be explicitly identified by numbers that correspond to teeth or tooth surfaces.

A tooth is identified by pronouncing the optional letter D followed by a pair of numbers. The numbering system is illustrated in Figure 9. The mouth is divided into four quadrants. The first number, 1 through 4, identifies the quadrant (upper right, upper left, lower left, lower right respectively). The second number, 1 through 8, is a running number from the center of the mouth to the back that identifies the tooth inside the quadrant. Baby teeth are numbered following the same logic, but the first number is now 5 through 8, and the second number is 1 through 5.

Some commands are targeted at specific surfaces of a tooth. The surfaces are then identified using a numbering system that is illustrated in Figure 10.

1. occlusal surface (the chewing surface)

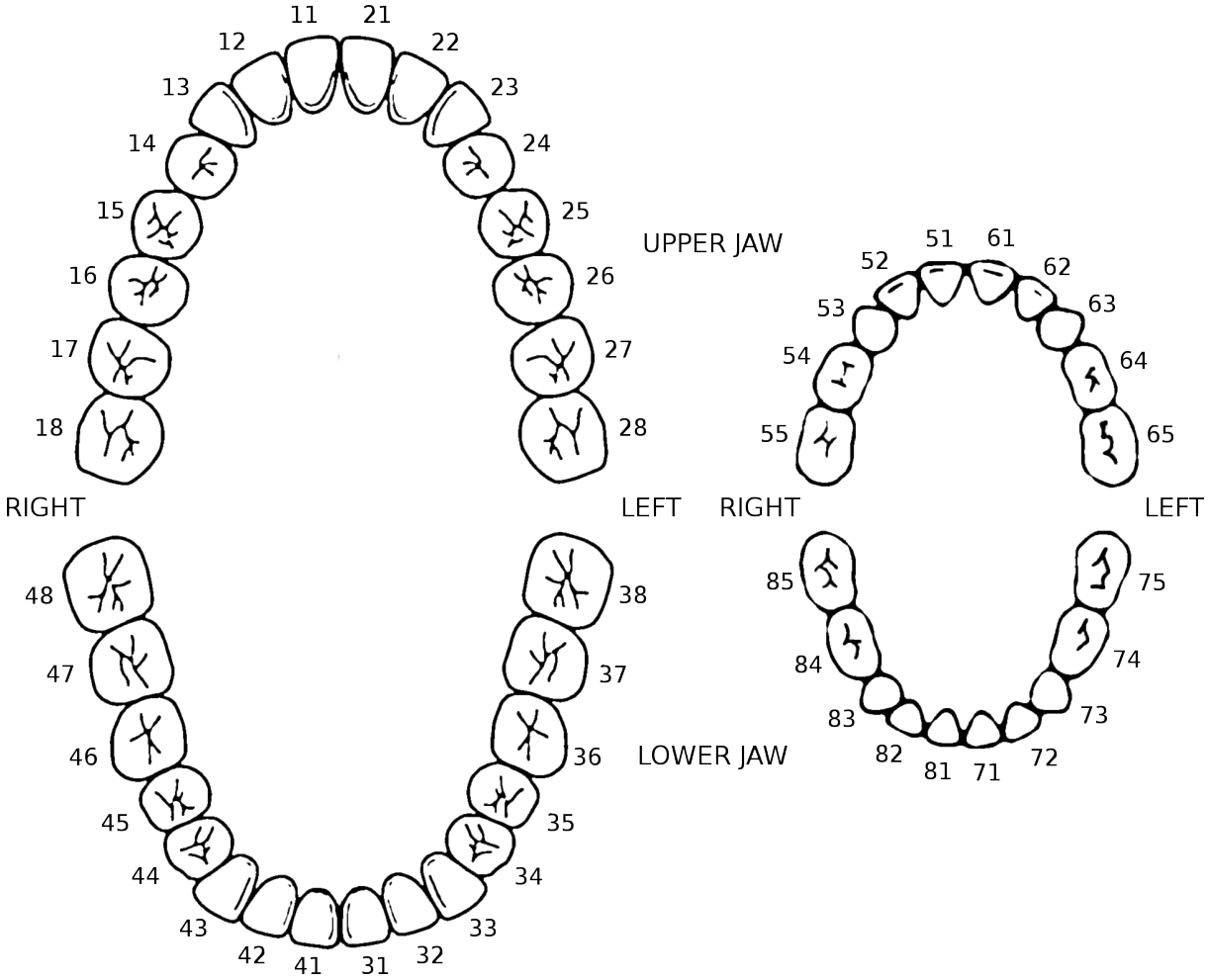

Figure 9: Adult teeth (left) and baby teeth (right). The tooth charts are viewed as mirror images, i.e. as seen by a person looking at the mouth.

2. mesial surface (the surface nearest to the mid-line of the body)
3. vestibular (facial) surface
4. distal surface (the surface furthest from the mid-line of the body)
5. lingual / palatal surface (the surface facing the tongue / palate)
6. gum boundary of surface 3
7. gum boundary of surface 5

The commands can be divided into five categories. Inside each category, the commands have a consistent structure, while the keyword changes. Between the categories the structure varies slightly, as different types of actions require different targets to be specified.

1. The most simple commands are targeted at an entire set of teeth, either the upper jaw, the lower jaw, or the entire mouth. For example, *poista alaleuan hampaat* [remove lower jaw teeth].

2. Commands may be targeted at a specific tooth by first dictating the two numbers that identify the tooth. For example, *dee neljä viisi kruunu* [dee four five crown].
3. Some commands are targeted at one or more tooth surfaces. Then the keyword is followed by tooth surface numbers. For example, *dee neljä kuusi karies kaksi kolme* [dee four six caries two three].
4. Bridge command is targeted at two teeth. For example, *silta dee neljä kolme viiva dee neljä viisi* [bridge dee four three dash dee four five].
5. Some commands can be targeted at any number of teeth. The tooth numbers are enumerated in a similar fashion.

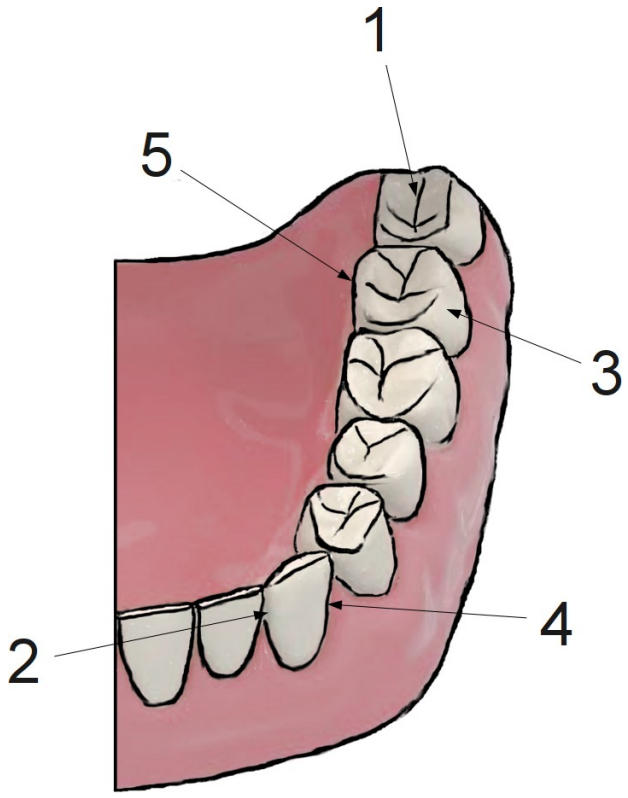

Figure 10: Numbering system for tooth surfaces.

5.2 Generating a Language Model from a Finite-State Grammar

The command language is most intuitively modeled as a finite-state grammar. Unfortunately the Aalto University recognizer is not able to read FSGs, because the development has focused on large vocabulary speech recognition. Implementing FSG support would give a perfect representation of the command language. However, we decided not to implement such feature, but convert the grammar to an n-gram language model, for several reasons.

1. The FSG describing the command language would be complex enough to demand huge computational resources for real-time speech recognition.
2. There is no need for the language model to be an exact representation of the grammar, as long as it accepts all the sentences of the grammar. If a slightly relaxed language model causes an utterance to be recognized as something that is not a valid command, chances are that the command actually was invalid. An exact language model would give the closest match that is valid according to the grammar, but it might be better to issue an error.

3. Because of the tight schedule of the project, it was necessary to get some results fast. A better language model could be adopted later, if the n-gram model was found to be inadequate.
4. It was interesting to see the performance of an n-gram language model in a command dictation application. It has potential for great speed improvements with small decrease of accuracy.

The grammar was converted to n-grams by simply enumerating all the different n-grams that the grammar allows. The grammar was modeled in extended Backus-Naur form and converted to HTK Standard Lattice Format (SLF) lattice using HParse tool. N-grams were enumerated using lattice-tool from the SRI Language Modeling Toolkit (SRILM).

The n-gram model accepts all the sentences of the grammar. Theoretically, it would be possible to construct an exact n-gram model that does not allow any other sentences, but because the language model has to accept long repetitions of tooth number sequences, the order number of such model would be too high to be computationally feasible.

Thus the limited length of the n-grams causes the model to accept some sentences that are not part of the grammar. The application has to validate the commands returned by the recognizer. If they are not in accordance with the grammar, the application issues an error or ignores the command. According to our tests (Table 4), even a 3-gram language model recognizes valid commands correctly.

The language model gives zero probability to all the n-grams that do not appear in the word lattice. For those that do, the two options below were considered for calculating the n-gram probabilities.

1. Even probability distribution between all the n-grams that begin with the same $n-1$ words.
2. Maximum likelihood estimation from the word lattice.

These two methods are evaluated in the following sections. Smoothing methods were not used, because they would distribute some probability mass from valid commands to word sequences that are not part of any valid command. A third possibility would have been to estimate the n-gram probabilities form a set of existing dictations. In that case smoothing would have been necessary, since there are too many different word sequences to get an exhaustive list of valid commands.

5.2.1 Even Distribution of N-Gram Probabilities

The first approach distributes transition probability evenly between all the n-grams that share the same $n-1$ word history. Consider an example where the dentist wants to associate some information to tooth D45 (*dee neljä viisi*). The dentist first dictates the tooth number and then one of the keywords that specifies the condition of the tooth. Below are some commands that the dentist could use.

dee neljä viisi kruunu [dee four five crown]
dee neljä viisi heiluu [dee four five loose]
dee neljä viisi intakti [dee four five intact]
dee neljä viisi tukihammas vasemmalle [dee four five abutment to the left]
dee neljä viisi tukihammas oikealle [dee four five abutment to the right]

The language model gives equal probability to all the keywords *kruunu*, *heiluu*, *intakti*, and *tukihammas*, when preceded by the three word sequence *dee neljä viisi*. In other words, the probability of an n-gram is calculated as

$$p(w|H)=\begin{cases}\frac{1}{N(H)}, & if\ N(H)>0\\ 0, & otherwise\end{cases}, \tag{28}$$

where $N(H)$ is the number of different n-grams whose word history is H in all the different paths of the word lattice.

If the previous $n-1$ words have been fixed, all the allowed choices for the next word have equal probability. It does not mean that all the different paths considered by the decoder have equal language model probability. The language model still has some effect on the recognition result.

- N-gram models penalize long word sequences, because the language model probabilities are applied for each word.
- The more allowed choices there are for the next word, the smaller the probability of each option is, penalizing paths with a large chance of confusion.

In our experiments, these properties were somewhat beneficial for the recognition result. Language model built using this method achieved perfect recognition accuracy (0 % error rate) with clearly pronounced voice samples, and was adopted for the final system.

In operational use some word sequences occur more frequently than others, and a language model that is biased towards those word sequences could give a better overall recognition accuracy. However, it is important that the dentists are able to use every command, even those that are only used in rare cases.

5.2.2 Maximum Likelihood Estimation from a Word Graph

The second approach could be argued by stating that all the commands are equally probable, so in the previous example, the keyword *tukihammas* should get twice the probability of the other keywords in the context *dee neljä viisi*. The probability of an n-gram is defined as

$$P(w|H)=\begin{cases}\dfrac{C(Hw)}{C(H)}, & if\ C(H)>0\\ 0, & otherwise\end{cases}, \tag{29}$$

where $C(Hw)$ is the number of occurrences of the n-gram Hw, and $C(H)$ is the total number of occurrences of n-grams whose history is H, in all the different paths of the word network. This is equal to considering all the allowed sentences equally probable, and computing the maximum likelihood n-gram model.

The problem with the second approach is that the variable length number sequences used to identify teeth and tooth surfaces make some command categories substantially more complex than others. Thus the number of different paths that go through a keyword varies radically between different keywords.

As an example, consider two commands from different categories that can be used to specify a denture, a prosthesis to replace missing teeth. The keyword *kokoproteesi* (complete denture) is used to specify a denture for patients who are missing all of their teeth on a particular arch, or in the entire mouth. The keyword *osaproteesi* (partial denture) is used to specify a denture for patients who are missing only some of their teeth on a particular arch.

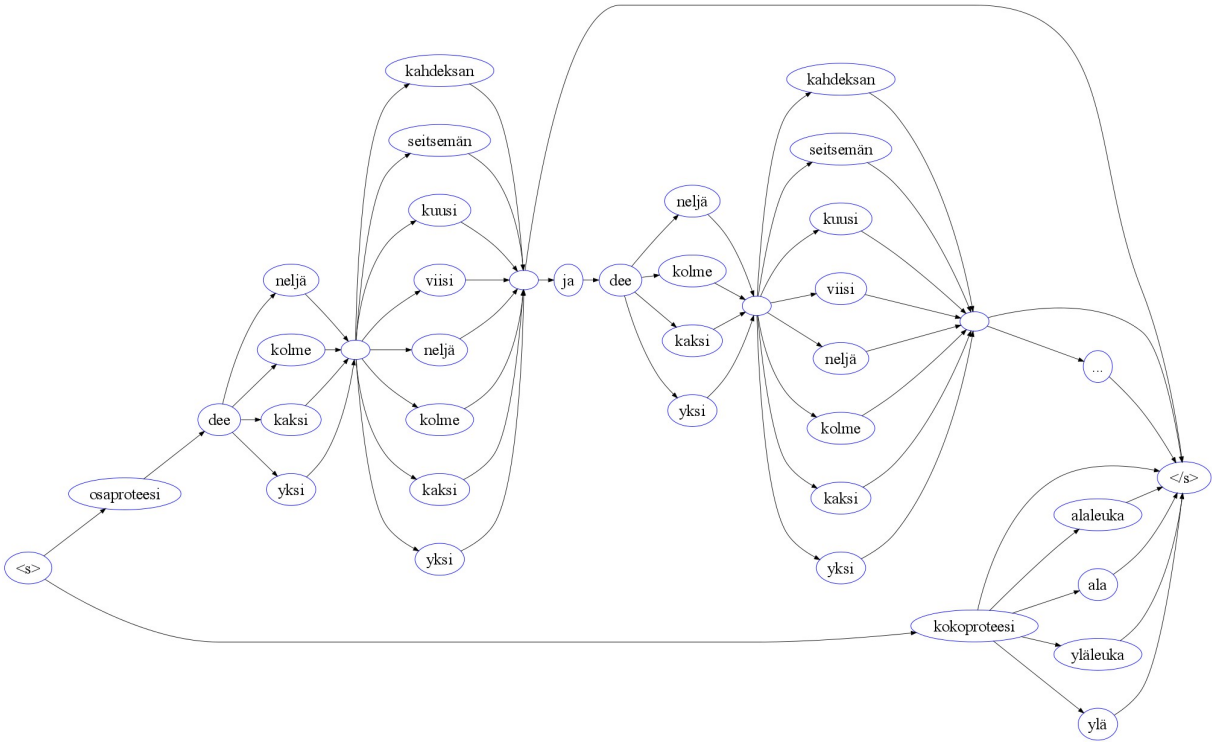

Figure 11: An illustration of the difference in complexity between command categories. The keyword osaproteesi starts a command that requires the specification of the exact teeth where a partial denture is applied. The word network for teeth specification is actually even more complex.

The former command can be addressed either to the upper jaw, lower jaw, or both, while the latter command requires that the exact teeth are identified. The word network in Figure 11 illustrates the structure of the commands. Because there are numerous paths that can be traversed from *osaproteesi* keyword through the network, but only five paths that can be taken from *kokoproteesi* keyword, the language model would strongly bias the decoder towards the *osaproteesi* branch. Acoustically the keywords are so similar that a *kokoproteesi* command would practically never be recognized. Consequently, this method was found to be useless for the task in hand.

5.3 Evaluation

5.3.1 Recognition Accuracy

Dentition status dictation was evaluated with commands recorded offline in a controlled environment, i.e. not during a patient's visit (see Table 4). As explained in Section 5.2.1, language model still has a small effect on the decisions made by the decoder, when the probabilities are distributed evenly among all the n-grams that share the same history. Language model scale factor controls the contribution of the language model to the decoding process (see Section 4.8). Different values were compared to evaluate the relevance of the language model probabilities.

Table 4: Recognition accuracy of dentition status dictation on clearly pronounced commands that were recorded offline. The used 3-gram language model was generated from a finite-state grammar.

			LM scale 0		LM scale 1		LM scale 30	
	Commands	Words	CER	WER	CER	WER	CER	WER
Speaker 1	100	491	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Speaker 2	38	166	5.26%	1.81%	5.26%	1.81%	0.00%	1.81%
All speakers	138	657	1.45%	0.46%	1.45%	0.46%	0.00%	0.46%

Setting the language model scale factor to zero essentially eliminates the use of the language model; the language model is used solely for restricting the search to the allowed sentences. In these experiments, the language model was somewhat beneficial for the recognition result. Raising the scale factor to 30 resulted in all the commands to be recognized correctly. The reason why word error rate is still higher than 0 %, is that the optional letter D prefixing a tooth number was not always recognized, but the command was still interpreted correctly.

5.3.2 Confidence Measures

During status dictation, the dentist may talk with other personnel and the patient while having the headset on. For hygienic reasons, dentists cannot press an activation button every time they want to issue a command, and they might not even have a hand available. Every utterance recorded from the headset is sent to the recognizer, and the software needs to reject those that are not valid commands.

Status dictation is a special case, since the purpose of the language model is to limit the output to the allowed command sentences, not to bias the results towards the most used word sequences. Confidence values were calculated using only acoustic likelihoods. In our experiments, four basic confidence measures [17] were employed:

Two-best Measures the acoustic likelihood ratio of the two best hypotheses:

$$1-\frac{P_2}{P_1} \quad , \tag{30}$$

where P_i is the acoustic likelihood of the i:th hypothesis. The likelihoods were scaled by an exponential parameter $\gamma<1$ in order to make the results more easily readable (see Section 4.8).

N-avg-best Measures the ratio of the average acoustic likelihood over the N best hypotheses to the acoustic likelihood of the best hypothesis:

$$1-\frac{\sum_{i=1}^{N} P_i}{N P_1} \tag{31}$$

N-sum-best Measures the ratio of the overall probability mass of the N best hypotheses to the best hypothesis:

$$\frac{P_1}{\sum_{i=1}^{N} P_i} \tag{32}$$

This differs from N-avg-best essentially only if the decoder returns less than N hypotheses for some utterances.

Avg-acoustic Measures the average acoustic likelihood per time frame, as given by the decoder.

$$\frac{P_1}{T} \quad , \tag{33}$$

where T is the length of the utterance in time frames.

N-avg-best and N-sum-best measures were calculated using $N=100$ and $N=10$ best hypotheses. Note that the first three measures may have a negative value, if the hypothesis with the highest posterior probability does not have the highest acoustic likelihood. Three different test sets were used:

- *Invalid* A set of 16 obviously invalid sentences according to the status dictation grammar.
- *Mispronounced* A set of 6 invalid sentences that were obtained by slightly altering the pronunciation of valid sentences.
- *Valid* A set of 138 valid sentences from two different speakers.

The purpose was to find a confidence measure that could distinguish valid commands from other discussion, so that the recognizer would reject utterances that are not meant as status commands. We also wanted to find out if the confidence measure could be used to reject commands with slight mistakes or unclear pronunciation that could cause them to be interpreted incorrectly.

Table 5 lists statistics of the six different confidence measures on the three data sets. True positive rate is defined as the percentage of valid commands that would be classified correctly, i.e. the sensitivity, using an optimal decision. True negative rate is the percentage of invalid utterances that would be classified correctly, i.e. the specificity. An optimal decision threshold is one that yields the highest combined sensitivity and specificity.

The mispronounced set is very small, but enough to show that the recognizer cannot distinguish slightly mispronounced commands from valid commands. The confidences of the mispronounced samples are clearly in the same range with the valid commands, regardless of the confidence measure. This is not a major problem for usability, since the users can verify

Table 5: Statistics of confidence measures on invalid, mispronounced, and valid commands. True positive rate and true negative rate are measured on an optimal decision threshold, i.e. one that gives the highest combined value.

	10-avg-best	100-avg-best	10-sum-best	100-sum-best	Two-best	Avg-acoustics
Invalid minimum	-127.06	-161.46	0.00	0.00	-16.90	0.705
Invalid average	-19.94	-22.24	0.40	0.23	-0.71	0.744
Invalid maximum	0.89	0.99	0.93	0.90	0.97	0.785
Mispronounced minimum	0.39	0.39	0.82	0.82	0.49	0.827
Mispronounced average	0.79	0.85	0.96	0.96	0.86	0.836
Mispronounced maximum	0.90	0.99	1.00	1.00	0.98	0.851
Valid minimum	-1.28	0.50	0.04	0.04	0.10	0.805
Valid average	0.87	0.96	0.97	0.94	0.92	0.831
Valid maximum	0.90	0.99	1.00	1.00	1.00	0.870
Optimal decision threshold	0.893	0.986	0.933	0.71	0.781	0.8
True positive rate	83%	83%	91%	93%	86%	100%
True negative rate	100%	94%	100%	94%	81%	100%

that the command was interpreted correctly, when the command is repeated through the headphones.

It is more important to distinguish general speech from command dictation. Valid and invalid sets are separable only by Avg-acoustics measure. In above experiments, it gave the best results, although there is little theoretical justification for using the acoustic likelihood without normalizing it with regard to the unconditional acoustic likelihood of the observed sequence, $P(X|\vartheta)$ (see Section 3.8).

For example, a speaker whose voice matches the acoustic model poorly, will systematically get lower confidences. A more robust classifier could be implemented by using an individual rejection threshold for each speaker. A proper value would be tuned by a few test samples. When testing the final product, clicks and crackle recorded by the microphone were occasionally recognized as short commands, with high acoustic likelihood, indicating that some sort of normalization would still be needed.

The first three confidence measures are calculated relative to different hypotheses, but the alternative hypothesis are restricted by the command language, so invalid utterances may receive a high confidence. A laborious solution would be to run two recognitions in parallel. The actual recognition would be performed using the command grammar, while a more relaxed language model would be used for confidence calculation.

6

Report Dictation

This chapter describes the language model used for transcription of medical records, such as patient treatment reports and plans. The dentition status commands, described in Chapter 5, obey a simple grammar that limits the possible word sequences from which the recognizer chooses the one that is most probable, given the observed speech signal. Treatment reports cannot be constrained by such a grammar, so the recognizer has to, in principle, consider every possible word sequence.

The vocabulary is not limited either, although in practice the jargon may be relatively limited, some words occurring substantially more frequently than others. It is likely that users dictate lengthy reports with more fluent speech than the status commands, so the recognizer has to be able to find where the words begin, even if they are pronounced together. These factors make it substantially more difficult to obtain an accurate transcription of continuous speech, than to recognize one of a distinct set of commands.

Not every word sequence occurs in Finnish language, and some occur more frequently than others. In free dictation, the purpose of the language model is to give prior probabilities to different word sequences. The probabilities are estimated from existing text material. A comprehensive, widely used corpus is the Finnish Language Text Collection from CSC[1]. The latest version contains 180 million tokens of literary Finnish, collected from Finnish journals, books, and newspapers from the 1990s.

To adapt the language model to the task in hand, the client collected a vast set of dental reports. The reports contain lots of acronyms, codes, and typos that add noise to the language model. It is difficult to give a precise number of proper words in the source material, but after preprocessing, the corpus contained 200 million words. Although such reports would in some cases be written in literary Finnish, the reports gathered by the client are not—simply because

1 The Finnish Language Text Collection is an electronic document collection available through CSC – IT Center for Science Ltd, http://www.csc.fi/.

it is faster to write colloquial Finnish, and the dentists are not professional typewriters. The vocabulary is often a mixture of colloquial and standard words.

6.1 Text Preprocessing

Before a text corpus can be used to build an n-gram language model, the written text has to be converted into spoken form. This consists of expanding abbreviations, acronyms, units, times, dates and such into proper words. Any words that are not in the form in which they are pronounced will add noise to the language model.

The language of the treatment reports is informal and meaningful only to other dental professionals. The reports contain

- many acronyms and abbreviations,
- dental jargon,
- codes, number sequences, and brand names,
- informal language, and
- typographical errors and spelling errors.

The acronyms and abbreviations are uncommon and can only be resolved when the context is understood. Constructing universal rules for expanding acronyms in any kind of text from newspapers to such jargon is next to impossible; some acronyms are used in more than one meaning even within the collection of treatment reports.

Acronyms, such as *NATO*, are pronounced as a single word and should not be expanded. Initialisms, such as *HTML*, are pronounced as individual letters, and should be expanded as a sequence of letter names. Some abbreviations, such as *URL*, can be pronounced either way. Also, in speech abbreviations may be translated into what they sand for. For example, one may use "*that is*" when speaking, but write "*i.e.*" instead.

Expansion rules for the abbreviations had to be relaxed, because the language in the reports does not follow Finnish grammar. The expansion rules cover only common abbreviations. Correcting typographical errors would require manually reading through the entire corpus. Clearly such work would not be expedient. The errors are accepted as noise in the language model.

6.2 Punctuation

Punctuation marks such as comma, period, and colon, are used to structure written text in the same way as intonation and pauses are used in speaking. Their most common purpose is to make the text easier to perceive, but they may also disambiguate its meaning. Thus their de-

tection is a practical issue in speech-to-text transcription. For example, consider the following sentences.

She bought another, black hat.
She bought another black hat.

The latter says that the woman already owned a black hat, and then bought another one. The former means that the woman owned a hat in another color, and then bought a black hat.

Although the meaning in the previous example depends on the presence of the comma—or intonation and pause length when speaking—it may be objected that a text whose meaning depends on punctuation is badly off and should be formulated in other words. If the recognizer needs nonetheless detect punctuation marks, there are two general strategies:

1) The user has to dictate punctuation marks explicitly, using predefined keywords, such as "comma" and "period". Similar commands may provide basic text editing facilities, such as "new line". Contextual clues may be used to discriminate between punctuation commands and the command keywords appearing as part of the dictated text [42].
2) The recognizer may try to deduce sentence boundaries automatically using a language model and acoustic indicators. An n-gram sentence boundary language model and acoustic pause duration model was tested on broadcast speech [20]. In the experiments pause duration was a more important indicator than the language model, but the language model information could be combined to improve the accuracy.

Often medical reports could hardly be understood without punctuation marks. Punctuation is used extensively, and not only for structuring the text, but also for compensating grammatical imperfections. Typical examples are:

- A period is used to denote boundary between a sentence and a heading.
- Commas are used to separate items in a list.
- A question mark turns a statement into a question.
- Parentheses are used for structuring and adding supplementary information.
- A comma is used as a decimal separator in Finnish language.
- A dash is used to indicate a range of values.

Because these punctuation marks are important to be recognized correctly, we decided that the user should dictate them explicitly. Thus, in the preprocessing step they were expanded as pronounced.

6.3 Segmenting Words into Morphs

Because Finnish words are commonly formed by concatenating long morph sequences, the huge number of different words that can be formed becomes a problem for language modeling. A word based language model cannot be estimated reliably.

The literary Finnish corpus considered in our experiments contains 140 million words, and 4.1 million unique words. The dental reports contain 1.3 million unique words. When combined, there are 5.1 million unique words (see Table 7). Thus, most of the words that are used in the dental reports do not exist in the literary Finnish collection. To some extent the difference can be explained by the fact that dentists use different vocabulary than newspapers, but mostly the new vocabulary is just different inflections or compounds derived from the same words.

Thus simply collecting more training material will not solve the problem. There is a practically endless amount of word forms that can be derived, and some of them will have no examples at all in the text. By constructing the language model from sub-word units, we can get more reliable estimates for their probabilities.

The vocabulary in an analytic language is more constrained, which is why a word is a suitable unit for English language models. For example, the British National Corpus, a 100 million word corpus of 20th century English, contains only approximately 250,000 unique words including plurals of nouns and verbs in different tenses [2].

The speech group in Aalto University has developed tools suitable for modeling Finnish and other synthetic languages [47]. The modeling is based on sub-word vocabulary units that we refer to as *statistical morphs*. Before training the language model, words are segmented into morphs. The model is then trained from the morph sequence.

We do not try to find morphs in the linguistic sense, since it would provide no additional benefit. The aim is simply to find substrings that occur frequently enough in different words in the corpus. The open source software package Morfessor has been developed for the task. The general idea is to find an optimal balance between the compactness of the morph lexicon, and the compactness of the representation of the corpus. It has been shown that segmentations created this way resemble linguistic morpheme segmentations. [13]

6.4 Estimating a Language Model from Segmented Text

After a convenient segmentation has been found for every word, the whole text material is segmented accordingly. As the text is now represented as a sequence of morphs, we need some means of deciding which morphs belong to the same word. A trivial solution is to introduce a special morph that denotes a word boundary.

Even though punctuation marks are dictated explicitly, we also use special morphs to denote sentence boundaries. The decoder will reset its context at these points. The sentence below has been taken from a preprocessed dental report. <w> denotes a word boundary, <s> denotes sentence start and </s> denotes sentence end.

<s> limakalvo illa <w> punoitusta <w> pilkku <w> hyperkeratoosi a <w>
kysymys merkki </s>
[redness on the linings comma hyperkeratosis question mark]

The morphs between adjacent word boundaries belong to the same word, and will be concatenated in the recognition result. The word boundary symbol is included in the vocabulary. Because word boundaries are explicit in the training data, the n-gram language model assigns a probability to a word boundary as well as any other morph, given the $n-1$ preceding vocabulary units. Silence is used as an acoustic clue of a word or sentence boundary, but sometimes words are pronounced together without a pause.

The segmented text could be used to estimate an n-gram model using any statistical language modeling tools, such as the freely available SRI Language Modeling Toolkit (SRILM) [49]. The problem with morph models is that a high model order is needed, because some words consists of long sequences of short morphs. On the other hand, some of the high-order n-grams are used very rarely. Traditionally all the n-grams up to the model order that were seen in the training data are included in the language model, which makes high-order language models inefficient.

We used VariKN language modeling toolkit, developed in Aalto University, for estimating a so-called variable order n-gram language model: the model order is high, but n-grams that do not significantly affect the overall probability distribution are pruned away from the model. This makes it especially well suited for sub-word vocabulary units. VariKN uses Kneser-Ney smoothing for n-gram probabilities, described in Section 3.5.3.

6.5 Building a Lexicon

Because Finnish phonemes generally correspond to letters, it is trivial to derive the pronunciation of Finnish words as a sequence of triphones. These pronunciations are collected into a lexicon. Some foreign, especially English, words need to be added manually.

English language has far more phonemes than the 24 that exist in Finnish. Thus, when the acoustic model that we are using is based on Finnish sounds, we cannot specify the exact pronunciations of all the English words. However, native Finnish speakers commonly pronounce English words using Finnish phonemes anyway, so adding transliterations of the words to the lexicon will improve the recognition.

It is not practical to include an exhaustive set of English vocabulary in the lexicon, because the English vocabulary would interfere with the recognition of Finnish words. Only those foreign words that were commonly used in the reports were added to the lexicon, at the expense of generality.

Numbers are used extensively in dental reports to refer to the patient's teeth. In the written reports they exist as number symbols, but as explained in Section 2.3.2, there are several different words in colloquial Finnish that can be used to refer to the same number. When speaking, the dentists use the different forms of the numerals interchangeably, and the speech recognizer should be able to recognize every form correctly. An exhaustive set of numerals and their inflections were added to the lexicon. This turns colloquial numerals into the standard form in the recognition result, which the recognizer can translate into number symbols in post-processing.

6.6 Evaluation

Both literary and colloquial Finnish are used in the training material, and both language varieties were seen in the recognition result. These were included as alternative spellings in the reference transcriptions. Also, in the medical transcriptions, several different spellings of some Latin loan words have been used. For example, the Finnish equivalent for *diastema* (the gap between two teeth) is *diasteema*, but both of these words are used in Finnish reports interchangeably. When assessing recognition performance, any of the spellings was considered correct.

6.6.1 Colloquial Numerals

To improve the recognition of numerals, an exhaustive set of their colloquial variants and inflections were added to the lexicon. Because the additional pronunciations cause also false numerals to be recognized more easily, this slightly increases error rates of non-numeric words. The effect on overall error rates was not substantial. Letter error rate decreased from 7.0 % to 6.7 % and word error rate decreased from 19.0 % to 18.2 % (see Table 6). However, because correct recognition of numerals is particularly important for dental transcription, a more important result is that the percentage of numbers that were recognized incorrectly or not recognized at all, dropped from 13 % to 5.7 %.

6.6.2 Selection of Training Material

Our purpose is to optimize the language model for dental reporting. Since the grammatical quality of the collected reports is poor, it might not be adequate training material alone. Also, the reports often include background information that is not directly related to dental health care, so we expected to gain advantage from a broad collection of training material.

Table 6: Recognition results from three speakers, with only standard pronunciations in the lexicon (columns two to five), and with additional colloquial pronunciations of numerals and an exhaustive set of their inflections added to the lexicon (columns six to nine).

	Standard pronunciations for numerals				Added colloquial pronunciations			
	Letters	Words	LER	WER	Letters	Words	LER	WER
Speaker 1	2,218	314	19%	41%	2,208	314	18%	40%
Speaker 2	2,199	343	3.5%	12%	2,240	343	1.8%	5.0%
Speaker 3	29,400	4,271	6.4%	18%	29,383	4,271	6.2%	18%
All speakers	33,817	4,928	7.0%	19%	33,831	4,928	6.7%	18%
Average	11,272	1,643	9.7%	24%	11,277	1,643	8.7%	21%

Our baseline language model has been constructed using 140 million words from the Finnish Language Text Collection. The material includes mostly newspaper articles that strictly follow Finnish grammar. We compared the recognition performance, when the language model has been trained using only the literary Finnish collection, only dental reports, and the combination of these two (see Table 7).

Table 7: Word counts of the Finnish Language Text Collection, the collection of dental reports, and their combination. Error rates obtained using language models estimated from each corpus on 92 minutes of medical dictations from four different speakers.

	FLTC	Reports	Combined
Millions of words	144	201	344
Millions of unique words	4.1	1.3	5.1
Letter error rate	31%	12%	13%
Word error rate	72%	26%	30%

The baseline language model gave inferior results. A language model trained using newspaper articles is not useful for medical transcription, because the vocabulary is too different. A surprising result was that the reports alone resulted in a better language model than their combination with the Finnish Language Text Collection. The report collection was large enough to be used for the task in hand without additional standard language text material. The final tests were performed with the language model estimated from the dental reports alone.

6.6.3 Recognition Accuracy

Recognition accuracy was measured with a collection of 92 minutes of dental reports, dictated by four different dental professionals (see Table 8). Utterances from Speaker 4 were not used for making design decisions, which partly explains the worst recognition accuracy. Utterances from the other speakers were instrumental in creating expansion and preprocessing scripts, but they were not used for estimating the language model. Recognition accuracy of the first three speakers was also used as guidance in some decisions regarding the acoustic model, but they were not used as training data. Therefore it seems that the poor results of Speaker 4 are mostly due to his speaking style.

Table 8: Recognition results on 92 minutes of medical dictations from four different speakers.

	Letters	Words	LER	WER
Speaker 1	2,213	314	14%	34%
Speaker 2	2,240	343	1.3%	5.2%
Speaker 3	29,377	4,271	7.8%	20%
Speaker 4	27,958	3,584	17%	35%
All speakers	61,788	8,512	12%	26%
Average	15,447	2,128	10%	24%

The disparity in accuracy between individual speakers is striking. The acoustic quality of the recordings does not explain the differences. Subjectively judging, all the recordings were high quality, but Speaker 2 had the loudest background noise from a radio. Partly the differences between speakers are explained by the fluency of their speech. Speaker 2 has very fluent speaking style and clear voice. All the other speakers try to speak clearly as well, but dictate longer notes and are not able to maintain as fluent speech. Still, by judging from the voice samples, one would not expect such radical differences.

7

Conclusion

This thesis studied the integration of the Aalto University speech recognizer into WinHIT dental information system. The main focus of the work was on language modeling.

An n-gram language model was constructed from a command grammar, allowing commands to be recognized using a statistical speech recognizer. Our benchmarks were calculated from commands dictated by end users, but not during a patient's visit. In this evaluation, all commands were recognized correctly.

A statistical language model was estimated from a collection of dental reports. Coupling the error reports with a literary Finnish corpus did not improve recognition performance, but slightly increased the error rates. The error rates of free speech dictation were higher than in the MobiDic benchmarks, where 4.6 % error rate was achieved without adaptation. The vocabulary and lack of language structure made transcription of dental reports more difficult.

There were huge differences in recognition accuracy from speaker to speaker, without obvious reason. The best letter error rate was 1.3 % and the worst was 17 %. There were some differences in the fluency of the speech, but all speakers had a relatively clear voice and speaking style. Currently the client is testing the final system with dental professionals.

Aalto University found several research topics for speech recognition that would be of practical interest:

- recognition of people's names during free speech transcription
- recognition of numbers during free speech transcription
- voice activity detection

7.1 Future Improvements

Better results from report dictation were constantly obtained by improving the preprocessing of the training material. There still remains room for improvement, but manually finding the expansion rules takes a lot of work.

People's names occur in the reports infrequently, but dentists often report their own name and the patient's name. There are 260,000 different last names that Finnish people have, but only a handful of these are represented in the corpus. As a solution, a secondary language model could be constructed from a list of Finnish names, and combined with the actual language model.

The average acoustics confidence measure, described in Section 5.3.2, is calculated as the average acoustic likelihood of the utterance. The confidence measure could be made more robust by normalizing the values using something that reflects the unconditional acoustic likelihood of the observation. The rejection threshold could also be made user-specific.

There are striking differences in recognition accuracy between different speakers. The differences we measured could not be explained by the acoustic quality of the recordings, background noise, or used vocabulary. However, they show that significant performance improvements could be gained by speaker adaptation. Ideally, the user interface would report the user name to the recognizer, and the recognizer would use and update the adaptation parameters of the current user. Adaptation was left out from this first prototype because of the amount of work its implementation would have required. For a constant recording environment, another possibility is to record some adaptation data and adjust the acoustic model beforehand.

Acoustic likelihood computation and decoding were separated into different threads of control, balancing the computational load quite evenly between the two threads. The computation could be parallelized to a much greater extent for example by distributing the evaluation of triphone models among multiple threads of control [41].

The usability could be improved by allowing partial results to be retrieved, and displaying them on the user interface, while the server is recognizing.

A

Communication Protocol

A.1 Design Principles

The network communication protocol between the user interface software and the recognition server is based on Media Resource Control Protocol (MRCP), developed by Cisco Systems, Nuance Communications, and Speechworks [46]. MRCP is a complex protocol for controlling a wide range of media processing resources, most notably automatic speech recognizers and speech synthesizers. It has been adopted by many commercial speech recognition and text-to-speech engines, such as IBM WebSphere Voice Server and Microsoft Speech Server, and commonly used in interactive voice response (IVR) applications.

MRCP relies on another protocol, such as Session Initiation Protocol (SIP) or Real Time Streaming Protocol (RTSP), for establishing a control sessions and an audio stream. By convention, Real-time Transport Protocol (RTP) is used for audio delivery. The control messages use a clear-text structure that resembles Hypertext Transfer Protocol (HTTP).

The protocol is clearly designed with IVR applications in mind. Such applications have typically used finite state grammars, partly because of the considerable expense of gathering relevant material for training a statistical language model [9]. MRCP servers are required to support the Extensible Markup Language (XML) form of the Speech Recognition Grammar Specification (SRGS) [23], and other formats may be supported subject to the implementation. MRCP servers may support statistical language models, but the n-gram language model used in this project is several hundreds of megabytes large, and it would be impractical to transfer it through the network.

A much simpler protocol was specified for this project. It assumes an underlying protocol that provides reliable, ordered delivery of a data stream. We have used Transmission Control

Protocol (TCP). All the communication is performed synchronously, and request identifiers are not needed. The message format is text based, but in contrast to MRCP, audio data delivery takes place in-band with the control data. This simplifies the implementation considerably. Audio is split into packets that are embedded in messages as binary data.

Instead of sending an entire grammar over the network, the client system selects a configuration file from a predefined set of options. There are currently two choices, one for free speech dictation, and one for dentition status dictation. The configuration files and the language model files reside on the server.

Another deficiency in MRCP with regard to free speech recognition is the lack of a possibility to retrieve a partial result while the recognition is in progress. Our implementation does not support retrieving partial results yet, but it can be implemented in the future, in order to be able to refresh the user interface more frequently in free speech dictation.

A.2 Detailed Protocol Specification

The communication consists of requests sent by the client, responses it receives from the server, and *RECOGNITION-COMPLETE* events where the client receives the recognition result. No other kinds of events are implemented. Every message consists of a *start-line*, a *message-header*, and a *message-body*. Lines are terminated by a *CRLF* sequence (hex 0D, 0A). The fields of a *start-line* are separated by an *SP* (hex 20). The protocol is presented in BNF notation below.

```
          <message> ::= <start-line>
                        <message-header>
                        <CRLF>
                        <message-body> | “”

       <start-line> ::= <request-line> | <response-line> |
                        <event-line>

     <request-line> ::= <message-length> <SP> <method-name>

      <method-name> ::= “SET-PARAMS” | “DEFINE-GRAMMAR” |
                        “RECOGNIZE” | “AUDIO”

    <response-line> ::= <message-length> <SP> <status-code>

       <event-line> ::= <message-length> <SP> <event-name>

   <message-header> ::= <field-name> “:” <field-value> CRLF
```

All the three message types, request, response, and event, start with message length. It specifies the total message length, including the start line. The field may be zero-padded to keep its

length constant for total message length calculation. Parameters for the methods are specified in the header fields, and data can be transferred in message body.

The start line of a request specifies the method to be performed:

- *SET-PARAMS* Not currently used, but reserved for specifying recognition parameters, such as speaker name.
- *DEFINE-GRAMMAR* Selects the language model from a predefined set of options.
- *RECOGNIZE* Starts recognition.
- *AUDIO* An audio packet. Audio data is transferred in the message body. An empty audio packet indicates end of audio.

The recognizer responds to every request with a response message. A status code indicates whether the method was successfully finished or not. Error codes indicate an error in the request received from the client, or a server internal error. There is only one kind of event that the server may generate, *RECOGNITION-COMPLETE*, which is used to transfer the recognition result in the message body.

The following header fields may be specified in a message:

- Voice-Name This attribute could be used to specify the speaker's name for speaker adaptation in a *SET-PARAMS* request.
- *Content-Location* Used to specify the language model configuration file in a *DEFINE-GRAMMAR* request.
- *Content-Length* Used to specify the length of the message body in bytes.

Bibliography

[1] Ahmed, N., Natarajan, T., and Rao, K. (1974). Discrete Cosine Transform. In *IEEE Transactions on Computers* **C-23**, p. 90-93.

[2] Al-Sayed, R., and Ahmad, K. (2003). Special Languages and Shared Knowledge. In *Electronic Journal of Knowledge Management* **1**, p. 197-212.

[3] Atal, B.S. (1974). Effectiveness of linear prediction characteristics of the speech wave for automatic speaker identification and verification. In *The Journal of the Acoustical Society of America* **55**, p. 1304-1312.

[4] Atal, B.S., and Hanauer, S.L. (1971). Speech Analysis and Synthesis by Linear Prediction of the Speech Wave. In *The Journal of the Acoustical Society of America* **50**, p. 637-655.

[5] Aubert, X.L. (2000). A Brief Overview of Decoding Techniques for Large Vocabulary Continuous Speech Recognition. In *Proceedings of the ISCA Workshop: Automatic Speech Recognition: Challenges for the new Millennium, ASR-2000*. Paris, France.

[6] Bahl, L., Brown, P., de Souza, P., and Mercer, R. (1986). Maximum Mutual Information Estimation of Hidden Markov Model Parameters for Speech Recognition. In *Proceedings of the 1986 IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP '86* ***11***, p. 49-52.

[7] Bogert, B., Healy, M., and Tukey, J. (1963). The Quefrency Alanysis of Time Series for Echoes: Cepstrum, Pseudo-Autocovariance, Cross-Cepstrum and Saphe Cracking. In *Proceedings of the Symposium on Time Series Analysis*, p. 209-243. New York.

[8] Brinton, L.J. (2000). *The Structure of Modern English: A Linguistic Introduction*. John Benjamins Publishing Company, Amsterdam / Philadelphia.

[9] Burke, D. (2007). *Speech Processing for IP Networks: Media Resource Control Protocol (MRCP)*. John Wiley & Sons, Inc.

[10] Cardin, R., Normandin, Y., and De Mori, R. (1991). High Performance Connected Digit Recognition Using Maximum Mutual Information Estimation. In *Proceedings of the 1991 IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP-91* ***1***, p. 533-536.

[11] Claes, T., Dologlou, I., ten Bosch, L., and van Compernolle, D. (1998). A Novel Feature Transformation for Vocal Tract Length Normalization in Automatic Speech Recognition. In *IEEE Transactions on Speech and Audio Processing* **6**, p. 549-557.

[12] Clark, H.H., and Fox Tree, J.E. (2002). Using uh and um in spontaneous speaking. In *Cognition* **84**, p. 73-111.

[13] Creutz, M., and Lagus, K. (2002). Unsupervised Discovery of Morphemes. In *Proceedings of the Workshop on Morphological and Phonological Learning of ACL-02*, p. 21-30. Philadelphia, Pennsylvania, USA.

[14] Davis, S., and Mermelstein, P. (1980). Comparison of Parametric Representations for Monosyllabic Word Recognition in Continuously Spoken Sentences. In *IEEE Transactions on Acoustics, Speech and Signal Processing* **28**, p. 357-366.

[15] Denham, K., and Lobeck, A. (2010). *Linguistics for Everyone: An Introduction*. Cengage Learning, Boston, MA.

[16] Digalakis, V., Rtischev, D., and Neumeyer, L. (1995). Speaker Adaptation Using Constrained Estimation of Gaussian Mixtures. In *IEEE Transactions on Speech and Audio Processing* **3**, p. 357-366.

[17] Dolfing, J.G.A., and Wendemuth, A. (1998). Combination of Confidence Measures in Isolated Word Recognition. In *Proceedings of the 5th International Conference on Spoken Language Processing 7*, p. 3237-3240. Sydney, Australia.

[18] Drevenstedt, G., McDonald, J., and Drevenstedt, L. (2005). The role of voice-activated technology in today's dental practice. In *The Journal of the American Dental Association* **136**, p. 157-161.

[19] Gales, M.J.F. (1998). Maximum Likelihood Linear Transformations for HMM-Based Speech Recognition. In *Computer Speech and Language* **12**, p. 75-98.

[20] Gotoh, Y., and Renals, S. (2000). Sentence Boundary Detection in Broadcast Speech Transcripts. In *Proceedings of the ISCA Workshop: Automatic Speech Recognition: Challenges for the new Millennium, ASR-2000*, p. 228-235. Paris, France.

[21] Hirsimäki, T. (2009). *Advances in Unlimited-Vocabulary Speech Recognition for Morphologically Rich Languages*. Ph.D. thesis, Helsinki University of Technology.

[22] Hirsimäki, T., Pylkkönen, J., and Kurimo, M. (2009). Importance of High-Order N-Gram Models in Morph-Based Speech Recognition. In *IEEE Transactions on Audio, Speech, and Language Processing* **17**, p. 724-732.

[23] Hunt, A., and McGlashan, S. (2004). Speech Recognition Grammar Specification Version 1.0. Online. The World Wide Web Consortium. Available http://www.w3.org/TR/speech-grammar/.

[24] Iskra, D., Grosskopf, B., Marasek, K., van den Heuvel, H., Diehl, F., and Kiessling, A. (2002). SPEECON - Speech Databases for Consumer Devices: Database Specification and Validation. In *Proceedings of the Third International Conference on Language Resources and Evaluation, LREC-2002*, p. 329-333. Paris, France.

[25] Jelinek, F. (1996). Five speculations (and a divertimento) on the themes of H. Bourlard, H. Hermansky, and N. Morgan. In *Speech Communication* **18**, p. 242-246.

[26] Jiang, H. (2005). Confidence measures for speech recognition: A survey. In *Speech Communication* **45**, p. 455-470.

[27] Juang, B.H., and Rabiner, L.R. (2005). Automatic Speech Recognition - A Brief History of the Technology Development. In *Elsevier Encyclopedia of Language and Linguistics*.

[28] Kneser, R., and Ney, H. (1995). Improved Backing-Off for M-Gram Language Modeling. In *Proceedings of the 1995 IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP-95* ***1***, p. 181-184.

[29] Le Bouquin Jeannes, R., and Faucon, G. (1994). Proposal of a voice activity detector for noise reduction. In *Electronics Letters* **30**, p. 930-932.

[30] Lowerre, B. (1990). The Harpy Speech Understanding System. In Waibel, A., and Lee, K.-F. (Ed.), *Readings in speech recognition*, Morgan Kaufmann Publishers Inc..

[31] Mansikkaniemi, A. (2009). *Acoustic Model and Language Model Adaptation for a Mobile Dictation Service*. M.Sc. thesis, Helsinki University of Technology.

[32] Määttänen, L.M. (2007). *Puheenomaisten piirteiden ilmeneminen erityyppisissä suomalaisissa kirjoitetuissa teksteissä*. M.Sc. thesis, Rijksuniversiteit Groningen.

[33] Ney, H., Haeb-Umbach, R., Tran, B.-H., and Oerder, M. (1992). Improvements in Beam Search for 10000-Word Continuous Speech Recognition. In *Proceedings of the 1992 IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP-92* ***1***, p. 9-12.

[34] Nádas, A. (1983). A Decision Theoretic Formulation of a Training Problem in Speech Recognition and a Comparison of Training by Unconditional Versus Conditional Maximum Likelihood. In *IEEE Transactions on Acoustics, Speech and Signal Processing* **31**, p. 814-817.

[35] Oppenheim, A., and Schafer, R. (1968). Homomorphic Analysis of Speech. In *IEEE Transactions on Audio and Electroacoustics* **16**, p. 221-226.

[36] Paunonen, H. (1994). The Finnish Language in Helsinki. In Nordberg, B. (Ed.), *The Sociolinguistics of Urbanization: The Case of the Nordic Countries*, Walter de Gruyter.

[37] Peleg, O., Givot, N., Halamish-Shani, T., and Taicher, S. (2011). Wrong tooth extraction: root cause analysis. In *British Dental Journal* **210**, p. 163.

[38] Pfeiffer, S. (2001). Pause concepts for audio segmentation at different semantic levels. In *Proceedings of the ninth ACM international conference on Multimedia*, p. 187-193. New York, NY, USA.

[39] Rabiner, L., Levinson, S., Rosenberg, A., and Wilpon, J. (1979). Speaker-Independent Recognition of Isolated Words Using Clustering Techniques. In *IEEE Transactions on Acoustics, Speech and Signal Processing* **27**, p. 336-349.

[40] Rabiner, L.R. (1989). A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. In *Proceedings of the IEEE* **77**, p. 257-286.

[41] Ravishankar, M. (1993). Parallel Implementation of Fast Beam Search for Speaker-Independent Continuous Speech Recognition. Technical Report submitted to Indian Institute of Science, Bangalore, India.

[42] Rosenthal, D.I., Chew, F.S., Dupuy, D.E., Kattapuram, S.V., Palmer, W.E., Yap, R.M., and Levine, L.A. (1998). Computer-Based Speech Recognition as a Replacement for Medical Transcription.. In *American Journal of Roentgenology* **170**, p. 23-25.

[43] Saukkonen, P. (1970). Puhekielen luonteesta. In *Kielikello* **3**, p. 3-9.

[44] Schaaf, T., and Kemp, T. (1997). Confidence Measures For Spontaneous Speech Recognition. In *Proceedings of the 1997 IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP-97* ***2***, p. 875-878.

[45] Schwartz, R., Chow, Y., Roucos, S., Krasner, M., and Makhoul, J. (1984). Improved Hidden Markov Modeling of Phonemes for Continuous Speech Recognition. In *Proceedings of the 1984 IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP '84* ***9***, p. 21-24.

[46] Shanmugham, S., Monaco, P., and Eberman, B. (2006). Request for Comments 4463: A Media Resource Control Protocol (MRCP) Developed by Cisco, Nuance, and Speechworks. Online. Internet Engineering Task Force. Available http://www.ietf.org/rfc/rfc4463.txt.

[47] Siivola, V., Creutz, M., and Kurimo, M. (2007). Morfessor and VariKN machine learning tools for speech and language technology. In *Proceedings of INTERSPEECH 2007*, p. 1549-1552.

[48] Sohn, J., and Sung, W. (1998). A Voice Activity Detector Employing Soft Decision Based Noise Spectrum Adaptation. In *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'98* ***1***, p. 365-368.

[49] Stolcke, A. (2002). SRILM - An Extensible Language Modeling Toolkit. In *Proceedings of the International Conference on Spoken Language Processing*, p. 257-286. Denver, Colorado.

[50] Turunen, M., Melto, A., Kainulainen, A., and Hakulinen, J. (2008). MobiDic - A Mobile Dictation and Notetaking Application. In *Proceedings of INTERSPEECH 2008*, p. 500-503. Brisbane, Australia.

[51] Ursin, M. (2002). *Triphone clustering in Finnish continuous speech recognition*. M.Sc. thesis, Helsinki University of Technology.

[52] Valtchev, V., Odell, J.J., Woodland, P.C., and Young, S.J. (1997). MMIE training of large vocabulary recognition systems. In *Speech Communication* **22**, p. 303-314.

[53] Varona, A., and Torres, M.I. (2004). Scaling Acoustic and Language Model Probabilities in a CSR System. In Sanfeliu, A., Martínez Trinidad, J., and Carrasco Ochoa, J. (Ed.), *Progress in Pattern Recognition, Image Analysis and Applications*, Springer Berlin / Heidelberg.

[54] Wiik, K. (1977). On Finnish syllables. In *Virittäjä*, Kotikielen Seura.

[55] Woodland, P.C., and Povey, D. (2002). Large scale discriminative training of hidden Markov models for speech recognition. In *Computer Speech and Language* **16**, p. 25-47.

[56] ITU-T (1990). *Recommendation G.726*. http://www.itu.int/rec/T-REC-G.726/e.