TKK Dissertations in Information and Computer Science
Espoo 2009 TKK-ICS-D14

ADVANCES IN UNLIMITED-VOCABULARY SPEECH RECOGNITION FOR MORPHOLOGICALLY RICH LANGUAGES

Teemu Hirsimäki

Dissertation for the degree of Doctor of Science in Technology to be presented with due permission of the Faculty of Information and Natural Sciences for public examination and debate in Auditorium TU1 at Helsinki University of Technology (Espoo, Finland) on the 6th of August, 2009, at 12 o'clock noon.

Helsinki University of Technology
Faculty of Information and Natural sciences
Department of Information and Computer Science

Teknillinen korkeakoulu
Informaatio- ja luonnontieteiden tiedekunta
Tietojenkäsittelytieteen laitos

Distribution:
Helsinki University of Technology
Faculty of Information and Natural Sciences
Department of Information and Computer Science
P.O.Box 5400
FI-02015 TKK
FINLAND
Tel. +358-9-451 3272
Fax +358-9-451 3277
Email: series@ics.tkk.fi

ISBN 978-951-22-9976-8 (Print)
ISBN 978-951-22-9977-5 (Online)
ISSN 1797-5050 (Print)
ISSN 1797-5069 (Online)
URL: http://lib.tkk.fi/Diss/2009/isbn9789512299775/

Multiprint Oy
Espoo 2009

Hirsimäki, T. (2009): **Advances in unlimited-vocabulary speech recognition for morphologically rich languages**. Doctoral thesis, Helsinki University of Technology, Dissertations in Information and Computer Science, TKK-ICS-D14, Espoo, Finland.

Keywords: Speech recognition, language modelling, n-gram models, morphology, error analysis.

ABSTRACT

Automatic speech recognition systems are devices or computer programs that convert human speech into text or make actions based on what is said to the system. Typical applications include dictation, automatic transcription of large audio or video databases, speech-controlled user interfaces, and automated telephone services, for example. If the recognition system is not limited to a certain topic and vocabulary, covering the words in the target languages as well as possible while maintaining a high recognition accuracy becomes an issue.

The conventional way to model the target language, especially in English recognition systems, is to limit the recognition to the most common words of the language. A vocabulary of 60 000 words is usually enough to cover the language adequately for arbitrary topics. On the other hand, in morphologically rich languages, such as Finnish, Estonian and Turkish, long words can be formed by inflecting and compounding, which makes it difficult to cover the language adequately by vocabulary-based approaches.

This thesis deals with methods that can be used to build efficient speech recognition systems for morphologically rich languages. Before training the statistical n-gram language models on a large text corpus, the words in the corpus are automatically segmented into smaller fragments, referred to as morphs. The morphs are then used as modelling units of the n-gram models instead of whole words. This makes it possible to train the model on the whole text corpus without limiting the vocabulary and enables the model to create even unseen words by joining morphs together. Since the segmentation algorithm is unsupervised and data-driven, it can be readily used for many languages.

Speech recognition experiments are made on various Finnish recognition tasks and some of the experiments are also repeated on an Estonian task. It is shown that the morph-based language models reduce recognition errors when compared to word-based models. It seems to be important, however, that the n-gram models are allowed to use long morph contexts, especially if the morphs used by the model are short. This can be achieved by using growing and pruning algorithms to train variable-length n-gram models. The thesis also presents data structures that can be used for representing the variable-length n-gram models efficiently in recognition systems.

By analysing the recognition errors made by Finnish recognition systems it is found out that speaker adaptive training and discriminative training methods help to reduce errors in different situations. The errors are also analysed according to word frequencies and manually defined error classes.

Hirsimäki, T. (2009): **Menetelmiä rajattoman sanaston puheentunnistukseen morfologisesti rikkaissa kielissä**. Väitöskirja, Teknillinen korkeakoulu, Dissertations in Information and Computer Science, TKK-ICS-D14, Espoo, Suomi.

Avainsanat: Puheentunnistus, kielimalli, n-gram-malli, morfologia, virheanalyysi.

TIIVISTELMÄ

Automaattiset puheentunnistusjärjestelmät muuttavat ihmisen puhetta tekstiksi tai tekevät muita toimintoja sen perusteella, mitä käyttäjä sanoo järjestelmälle. Tavallisimpia sovelluksia ovat esimerkiksi sanelukoneet, suurten ääni- ja videoaineistojen automaattinen tekstitys, puheohjattavat käyttöliittymät ja automaattiset puhelinpalvelut. Jos tunnistusjärjestelmää ei ole rajattu tiettyyn aihealueeseen tai sanastoon, muodostuu tärkeäksi ongelmaksi, kuinka kohdekielen sanasto voidaan kattaa laajasti heikentämättä tunnistimen tarkkuutta.

Perinteinen tapa mallintaa kieltä — erityisesti englannin kielelle tarkoitetuissa puheentunnistusjärjestelmissä — on rajoittaa tunnistimen sanasto kattamaan vain kielen yleisimmät sanat. Jo 60 000 sanan sanasto riittää tyypillisesti kattamaan riittävästi suurimman osan aiheista. Näin ei kuitenkaan ole morfologisesti rikkaissa kielissä kuten suomi, viro ja turkki. Näissä kielissä runsas yhdyssanojen ja taivutusmuotojen käyttö tekee sanoista pitkiä ja sanastosta hyvin laajan, minkä vuoksi kielen kattaminen sanastopohjaisilla menetelmillä on hankalaa.

Tässä väitöskirjassa käsitellään menetelmiä, joilla voidaan parantaa morfologisesti rikkaille kielille suunniteltuja puheentunnistusjärjestelmiä. Tilastollisten kielimallien opettamiseen käytetyt suuret tekstiaineistot pilkotaan automaattisella menetelmällä pienempiin osiin, morfeihin. Näitä morfeja käytetään sanojen sijaan tilastollisten kielimallien perusyksiköinä, joita yhdistelemällä voidaan mallintaa koko opetusaineisto ilman että sanastoa täytyy rajoittaa. Näin voidaan mallintaa myös sanoja, jotka eivät ole esiintyneet ollenkaan opetusaineistossa. Ohjaamaton datalähtöinen menetelmä on pitkälti myös kieliriippumaton.

Menetelmiä vertaillaan useissa suomenkielisissä puheentunnistustehtävissä, ja osa kokeista toistetaan vironkielisessä tehtävässä. Kokeet osoittavat että morfipohjaisilla tunnistimilla saavutetaan korkeampi tunnistustarkkuus sanastopohjaisiin menetelmiin verrattuna. Erityisesti tunnistustarkkuus paranee sanoissa, jotka ovat sanastopohjaisten järjestelmien sanaston ulkopuolella. Koska tyypillisesti osa morfeista on hyvin lyhyitä, on tärkeää että tilastolliset kielimallit pystyvät käyttämään vaihtelevan pituista kontekstia tarpeen mukaan. Työssä esitetään tämän kaltaisille kielimalleille tehokkaita tietorakenteita.

Väitöskirjan loppuosassa tarkastellaan suomenkielisen tunnistusjärjestelmän tekemiä virheitä sen mukaan, mikä rooli äännemalleilla ja kielimalleilla on virhekohdissa. Havaitaan että puhuja-adaptaation käyttö tunnistimen opetuksen ja käytön aikana vähentää virheitä erityisesti niissä kohdissa, joissa äännemallit ovat suosineet väärää hypoteesia. Käyttämällä diskriminatiivista opetusmenetelmää adaptaation ohella vähentyvät virheet erityisesti kohdissa, joissa kielimalli on suosinut väärää hypoteesia. Virheanalyysissa tutkitaan myös millaisten sanojen kohdalla tunnistin tekee eniten virheitä.

PREFACE

This work has been carried out in the Adaptive Informatics Research Centre (formerly Neural Networks Research Centre) of the Department of Information and Computer Science, Helsinki University of Technology. I am grateful for the funding provided by the Academy of Finland in the projects Adaptive Informatics and New Adaptive and Learning Methods in Speech Recognition, the EMIME project in the European Community Framework Program 7, Helsinki Graduate School in Computer Science and Engineering, and the former Department of Computer and Information Science. I am also grateful for the personal grants from the Nokia Foundation and the KAUTE Foundation.

There are several people who have helped me greatly during my thesis work. First of all, I thank my instructor Dr. Mikko Kurimo for giving me the possibility to work in the speech group. Ever since I joined the group, Mikko has been very encouraging and always available for discussion. He has been a co-author in most of the publications and arranged a major part of the funding. I also thank my supervisor Prof. Erkki Oja for encouragement and valuable comments on this thesis. As the head of the research centre, he has created an outstanding environment for research.

I am grateful to all my colleagues, who have made the working atmosphere very inspiring. Especially, I want to thank my good friend Dr. Vesa Siivola, with whom I have shared an office for several years. Vesa has helped me greatly, and much of the research presented in the thesis has been conducted with him. I also thank Dr. Mathias Creutz, Janne Pylkkönen, Sami Virpioja and Matti Varjokallio for interesting discussions and fruitful collaboration, and the whole speech group for support and help.

I thank Prof. Anssi Klapuri and Dr. Dilek Hakkani-Tür for reviewing the manuscript. Their valuable comments helped me to improve the work.

Finally, I thank my dear parents Launo and Kerttu, who have supported me and my education tremendously, and my beloved family: Kristel, Nea and Joel.

Otaniemi, June 2009
Teemu Hirsimäki

CONTENTS

PUBLICATIONS OF THE THESIS

The thesis consists of an introductory part and the following scientific publications.

I Vesa Siivola, Teemu Hirsimäki, Mathias Creutz and Mikko Kurimo. Unlimited vocabulary speech recognition based on morphs discovered in an unsupervised manner. In *Proceedings of the 8th European Conference on Speech Communication and Technology (Eurospeech)*, 2003, pp. 2293–2296.

II Teemu Hirsimäki, Mathias Creutz, Vesa Siivola, Mikko Kurimo, Sami Virpioja, Janne Pylkkönen. Unlimited vocabulary speech recognition with morph language models applied to Finnish. *Computer Speech and Language*, 20(4), October 2006, pp. 515–541.

III Vesa Siivola, Teemu Hirsimäki, and Sami Virpioja. On growing and pruning Kneser-Ney smoothed n-gram models. *IEEE Transactions on Audio, Speech and Language Processing*, 15(5), July 2007, pp. 1617–1624.

IV Teemu Hirsimäki. On compressing n-gram language models. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2007, pp. IV-949–952.

V Mathias Creutz, Teemu Hirsimäki, Mikko Kurimo, Antti Puurula, Janne Pylkkönen, Vesa Siivola, Matti Varjokallio, Ebru Arısoy, Murat Saraçlar and Andreas Stolcke. Morph-based speech recognition and modeling of out-of-vocabulary words across languages. *ACM Transactions on Speech and Language Processing*, 5(1), December 2007, pp. 3:1–3:29.

VI Teemu Hirsimäki, Janne Pylkkönen and Mikko Kurimo. Importance of high-order n-gram models in morph-based speech recognition. *IEEE Transactions on Audio, Speech and Language Processing*, 17(4), May 2009, pp. 727–732.

VII Teemu Hirsimäki and Mikko Kurimo. Analysing recognition errors in unlimited-vocabulary speech recognition. *Proceedings of Human Language Technology, Conference of the North American Chapter of the Association for Computational Linguistics, HLT-NAACL*, 2009, pp. 193–196.

LIST OF ABBREVIATIONS

AD	Absolute discounting
AM	Acoustic model
ASR	Automatic speech recognition
CMLLR	Constrained maximum likelihood linear regression
CMS	Cepstral mean subtraction
DCT	Discrete cosine transform
EP	Entropy-based pruning
ERA	Error region analysis
FFT	Fast Fourier transform
GT	Good-Turing
HMM	Hidden Markov model
IPA	International phonetic alphabet
KN	Kneser-Ney
KNG	Kneser-Ney growing
KP	Kneser pruning
LER	Letter error rate
LM	Language model
MFCC	Mel-frequency cepstral coefficient
ML	Maximum likelihood
MLLT	Maximum likelihood linear transform
MPFE	Minimum phone frame error
OOV	Out-of-vocabulary
RKP	Revised Kneser pruning
SAT	Speaker adaptive training
WER	Word error rate

LIST OF SYMBOLS

A	Adaptation matrix
b	Adaptation bias
$C(\boldsymbol{h}w)$	Number of times n-gram $\boldsymbol{h}w$ appears in training data
$C'(\boldsymbol{h}w)$	Modified counts used in Kneser-Ney smoothing
D	Discounting term in absolute discounting
$\boldsymbol{h}$	Word history
$\hat{\boldsymbol{h}}$	Word history with the first word removed
O	Sequence of observations
o_1^t	Sequence of observations: $o_1^t = (o_1, \ldots, o_t)$
P_{AM}	Acoustic model
P_{DUR}	Duration model
P_{KN}	Kneser-Ney smoothed model
P_{AD}	Absolute-discounted model
P_{LA}	Language model look-ahead
P_{LM}	Language model
S	State sequence
S_r	State sequence corresponding to the reference transcription
T	Set of training sentences
v, w	Word
w_0	Start-of-sentence symbol
w_*	End-of-sentence symbol
w_i^j	Sequence of words: $w_i^j = (w_i, \ldots, w_j)$
W	Sequence of words
$\lvert W \rvert$	Number of words in sequence W
$\alpha(\boldsymbol{h}w)$	Explicit estimate for $P(w\lvert\boldsymbol{h})$ stored in the language model
$\beta(\boldsymbol{h})$	Back-off coefficient
$\beta'(\boldsymbol{h})$	Logarithm of the back-off coefficient stored in the model
γ	Interpolation coefficient
δ	Duration model scaling factor
κ	Posterior scaling factor in discriminative training
λ	Language model scaling factor
Λ	Parameters of the HMMs

CHAPTER 1
INTRODUCTION

1.1 Automatic speech recognition

Speech is one of the most fundamental ways for humans to communicate with each other. All kinds of intentions, thoughts, and emotions can be conveyed in speech naturally, albeit that misunderstandings also happen. Since the advent of computers, there has been strong interest in making computers, too, able to process and understand human speech. The systems that are able to transcribe human speech are called automatic speech recognition (ASR) systems.

ASR technology can be utilised in a variety of applications. The most typical example is perhaps a dictating machine. Instead of typing text with a keyboard, or recording speech on tape that is transcribed manually later, the text is directly dictated to an ASR system, which then converts the speech into text automatically.

The direct speech-to-text conversion is also useful in search engines for large audio databases. ASR systems can be used for transcribing all speech from the material, after which the material can be browsed in textual form or users can make textual queries in order to fetch relevant audio clips from the database. This application is often called spoken document retrieval, and it is becoming increasingly relevant as more and more audio and video material is created, collected, and published every day. A retrieval system for British and North American broadcast news, for example, is presented by Renals et al. (2000), and Hansen et al. (2005) describe a large-scale system for accessing historically significant speeches, news broadcasts, and recordings from the U.S.

Another application related to spoken document retrieval is automatic transcription of meetings. Besides transcribing just the speech, the aim is to track who is speaking, and to index and summarise the content of the meeting for later retrieval. Recent advances in meeting transcription systems are described, for example, by Renals et al. (2007) and Tur et al. (2008).

There are also applications where the aim of the system is not to convert the user's speech into text, but to understand the message in the speech and act accordingly. In the simplest case, a restricted set of spoken commands can be used to control a device or a computer program. It is also possible to build more complicated spoken dialog systems, which allow people to use spoken language to accomplish tasks, such as querying timetables or buying train tickets (see, e.g., Williams and Young, 2007). Speech synthesis technology is used to provide feedback to the user in speech. The largest challenge in these systems is to model the course of the dialog reliably, so that the system can provide relevant answers and ask clarifications from the user when necessary.

Depending on the application and the environment in which the ASR system is used, the recognition accuracy of the system can be anything between perfect and very poor. The difficulty of a recognition task is affected largely by the following

factors:

1. *Vocabulary*: If the vocabulary used in the recognition task is small and the words are acoustically dissimilar, it is easier to recognise the words correctly. On the other hand, if the vocabulary is large, it is more likely that there are many words the pronunciation of which are identical or very close to each other: *night* and *knight*, *to* and *too*, for example.

2. *Type of speech*: In a simple spoken-command user interface, the user of the system may be required to keep short pauses between the words. This makes it easier for the recogniser to locate where the words begin and end. In fluent speech, often referred to as *continuous speech* in the context of automatic speech recognition, the words are usually pronounced together without distinct pauses in between making the recognition task harder. Furthermore, it is more difficult to recognise conversations than planned speech, because speakers are more likely to hesitate, use incomplete sentences, interrupt other speakers, and correct mistakes by starting over.

3. *Speaker dependency*: Most recognition systems are designed to be speaker independent nowadays. Although every person has a distinct voice, and someone's speech may be easier to recognise than someone else's, speaker-independent systems are able to recognise anyone's speech. However, if the user of the system is known beforehand, it is possible to train or adapt a recognition system for a particular speaker and increase the recognition accuracy considerably.

4. *Noise*: All kinds of background noise makes it harder to recognise the words correctly. Humans are surprisingly good at using context and visual hints in order to understand speech in noisy environments, but present ASR systems tend to be very sensitive to noise.

The speech recognition technology has gradually matured during the latter half of the 20th century, as the continuously increasing computer power and capacity has made it possible to design more and more complex recognition systems. The early systems, developed in the 1950's, were mainly designed to recognise phonemes, syllables or isolated digits. During the 1960's and 1970's, the vocabularies used in isolated word recognition grew from 10–100 words up to 1000 words. In the 1980's, the methodology shifted towards statistical methods and large-vocabulary systems recognising continuous speech. Since then, further development of training methods for statistical acoustic and language models has enabled building speaker-independent systems that can recognise even conversational speech with reasonable accuracy. A good review on the history of ASR research is given by Juang and Rabiner (2005).

1.2 The scope of the thesis

This thesis deals with methods that improve the accuracy and efficiency of ASR systems in general, focusing on systems that recognise continuous speech using very large or even unlimited vocabularies. The issues related to very large vocabularies are especially relevant when speech recognition systems are designed for morphologically rich languages, where long words are formed by compounding and

inflecting. The vocabulary growth caused by rich morphology makes it difficult to cover the words of the language adequately with traditional word-based methods. This work considers language models based on segmenting words into shorter units (called morphs) and forming words by joining the morphs together. The speech corpora used in the speech recognition experiments contain mostly planned speech with little background noise. Most of the experiments are performed on Finnish corpora, but some experiments on Estonian and Turkish corpora are also considered. The presented methods are not aimed for any specific application, but the recognition system described in this thesis is perhaps best suited for applications where speech-to-text conversion is desired.

1.3 Contributions of the thesis

The main contributions of this work are the following.

- The interaction between morph segmentation and the context length used in n-gram modelling is studied. High-order variable-length n-gram models are shown to be essential in speech recognition when the language models are based on morphs, especially when the morphs are short. It is also shown that morph-based language models improve the recognition accuracy specifically for out-of-vocabulary words.
- A recognition system that is able to handle high-order variable-length n-gram models is described.
- A compact data structure presented earlier in the literature is extended so that the memory footprint of variable-length n-gram models becomes smaller. An alternative structure allowing efficient decoding is also described.
- The roles of the acoustic and language models are analysed in the errors made by Finnish speech recognition systems. It is shown that speaker adaptive training and discriminative training correct different types of errors. Based on a manual error classification, the most likely error sources are also discussed.

1.4 Structure of the thesis

This thesis consist of an introductory part, together with seven publications. The introductory part presents the main ideas, experiments, and results from the publications as a coherent story, but some issues are discussed in greater detail in the publications.

1.4.1 The introductory part

The rest of the introductory part is organised as follows. Chapter 2 describes the most current version of the recognition system that has been continuously developed during the research work presented in this thesis. The speech and language corpora used in the speech recognition experiments, the evaluation tasks, and the experimental settings are also described. Chapter 3 introduces the n-gram language models and describes the standard techniques for training statistical language models using large text corpora. Recent advances in using growing and

pruning algorithms for creating high-order variable-length models are presented, and efficient data structures for n-gram models are proposed. Chapter 4 tackles the problems that make recognition of morphologically rich languages challenging, and describes how efficient language models can be trained by combining automatic word segmentation algorithms and n-gram modelling techniques. Chapter 5 analyses what kind of recognition errors the system makes after all methods presented in the thesis are applied in recognition. Final conclusions are presented in Chapter 6.

1.4.2 The publications and author's contributions

Publication I compares traditional word-based n-gram models to models based on two kinds of subword-units: syllables and so called statistical morphs obtained with the first version of the Morfessor algorithm. The Finnish speech recognition results suggest that the morphs improve the recognition accuracy considerably. The author was responsible for developing the morph-based decoder of the recogniser and participated in designing the experiments.

Publication II extends Publication I by evaluating the morph-based approach by using higher-order n-gram models (up to 5-grams) in two Finnish speech recognition tasks. Comparison is also made to models based on linguistic morphemes obtained using an automatic morphological analyser and to word-based models using a 410 000-word vocabulary. The Morfessor algorithm and the recognition system are described in detail. It is shown that the morph-based model gives the best recognition accuracy, and the reasons behind the results are analysed. The author was the main writer of the publication and designed a major part of the experiments.

Publication III presents algorithms for growing and pruning n-gram models while maintaining the properties of Kneser-Ney smoothing. It is shown that the proposed algorithms improve the models obtained with earlier pruning algorithms. The author contributed to the analysis and formulation of the proposed algorithms. The experiments and writing the article were done jointly with the main author.

Publication IV presents a data structure that can be used to store variable-length n-gram language models compactly in a speech recognition system. It extends an earlier compact structure by removing redundancy present in variable-length n-gram models. The author was solely responsible for this work.

Publication V compiles Finnish, Estonian, Turkish, and Arabic recognition experiments reported in earlier publications and analyses the reasons why the morph-based models outperform word-based models. The author contributed to the original Finnish recognition experiments and had a minor contribution in writing the article.

Publication VI reviews speech recognition experiments performed on many morphologically rich languages. The results reported in the literature have been varying: in some experiments morph-based approaches have given better results, while word-based approaches have worked better in others. Based on the review, the interaction between n-gram model order and morph segmentation is studied. The results of Finnish and Estonian recognition tasks suggest that, depending on the length of the morphs, high-order n-gram models may be crucial, which partly explains the varying results in the literature. The author was the main writer of the article, contributed to designing the experiments, and was responsible for the error analysis.

Publication VII analyses recognition errors made by Finnish recognition systems. It is shown that speaker adaptive training and discriminative training help to reduce errors in different situations. By analysing errors according to word frequencies and manually classified error types, most potential areas for further improvement are discussed. The author did the experiments and was the main writer of the publication.

CHAPTER 2
RECOGNITION SYSTEM AND EVALUATION TASKS

Modern speech recognition systems are very complex. The methods used for preprocessing speech signals, training the probabilistic phoneme and language models, and performing the actual recognition are a result of decades' active theoretical and experimental research. This chapter introduces the main methods used in modern recognition systems and describes the research system used in this work.

2.1 Overview

Formally, the speech recognition problem can be defined as a probabilistic optimisation problem. Given the observed speech signal O, the task is to find the most probable word sequence $\hat{W}$:

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

$$= \arg\max_W \frac{P(W)P(O|W)}{P(O)} \tag{2.2}$$

$$= \arg\max_W P(W)P(O|W). \tag{2.3}$$

Before this formulation can be used to recognise speech, one needs to define the *language model* (LM) $P(W)$ and the *acoustic model* (AM) $P(O|W)$. This process is called *training* the recogniser (or the models). When the models are available, new speech can be recognised as illustrated in Figure 2.1. First, signal processing techniques are used to extract relevant features from the speech signal. Then, given the feature vectors, the acoustic model and the language model, the most probable word sequence is searched. This process is often called *decoding*, since the system, in a way, decodes the original message that is encoded as a speech signal.

2.2 Feature extraction

The aim of the feature extraction phase is to reduce the dimensionality of the input signal and extract features that contain relevant information of the underlying phonetic structure. The extraction process is illustrated in Figure 2.2. The signal is first divided in frames using a 25 ms Hamming window with 8 ms frame shift. The short-time power spectrum is computed with the fast Fourier transform (FFT). Using Mel-scaled (Picone, 1993) filter banks and discrete cosine transform (DCT), 12 Mel-frequency Cepstral Coefficients (MFCCs) are computed and combined with the average power. Cepstral mean subtraction (CMS) (Rosenberg et al., 1994) removes the mean of the 150 surrounding frames from the feature vector. A 5-frame window is used to compute the first and second time-derivatives resulting

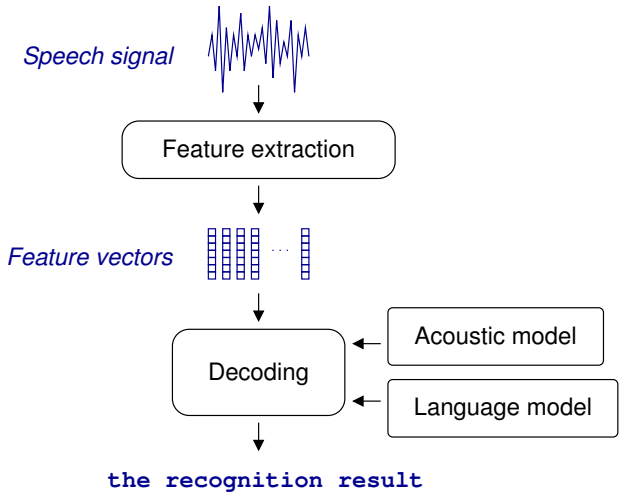

Figure 2.1: Overview of the recognition process.

in a 39-dimensional vector. Finally, three linear transforms are applied: a global mean and variance normalisation, a maximum likelihood linear transform (MLLT) (Gales, 1999), and an optional speaker-dependent transform if speaker adaptation is desired.

2.3 Acoustic models

The Finnish phoneme models of the system are almost directly based on the letters used in writing. Table 2.1 shows the phoneme labels, the corresponding phoneme symbols according to the international phonetic alphabet (IPA), and example words. Even if the system does not have a separate model for [ŋ], it will be implicitly modelled by the context-dependent model as described later. In order to handle more infrequent letters (c, q, w, x, z, å) found in some foreign words, the following simple rules are used to map the letters to phoneme labels: ch → `ts`, c → `k`, qu → `kv`, w → `v`, x → `ks`, z → `ts`, å → `o`.

For each phoneme, several context-dependent variants (also called *triphones*) are trained taking into account the neighbouring phonemes. For example, the triphone `t-a+n` corresponds to the case where `a` is preceded by `t` and followed by `n`. Each triphone is modelled with a 3-state hidden Markov model (HMM), and the observation densities in the 39-dimensional feature space are modelled with Gaussian mixture models with diagonal covariances.

2.3.1 Maximum likelihood training

The HMMs are trained by maximising an objective function that depends on the training data. In the conventional maximum likelihood (ML) training, the objective is to maximise the likelihood of the observations in the training data given the HMM of the reference transcription S_r:

$$F_{\mathrm{ML}}(\Lambda) = P(O|S_r, \Lambda). \tag{2.4}$$

where Λ denotes the HMM parameters.

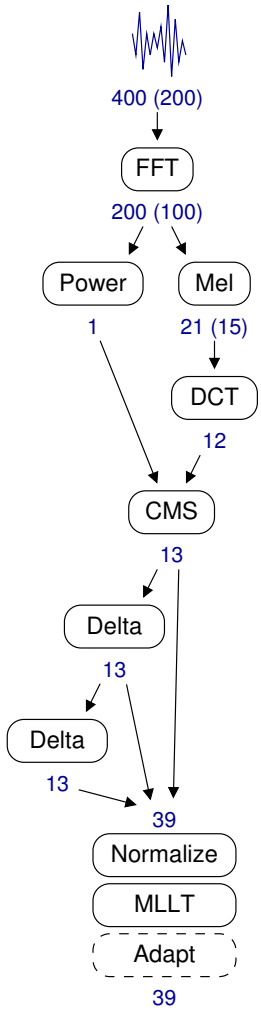

Figure 2.2: The feature extraction process of the recognition system and the dimensionality of the feature vector between the transforms. The parenthesised values apply if the sample rate of the input signal is 8000 Hz instead of the default 16 000 Hz.

2.3.2 Discriminative training

During the past decade, it has become popular to apply discriminative training methods to train the HMMs. Instead of finding the HMM parameters that maximise the likelihood of the training data, the aim is to maximise an objective function that measures the recognition accuracy on the training data. Various objective functions have been experimented in the literature (e.g., Zheng and Stolcke, 2005; Povey and Kingsbury, 2007), and when compared to plain ML training, discriminative training typically decreases error rates around 10 % relative.

The discriminative model used in this work is based on the minimum phone frame error (MPFE) criterion (Zheng and Stolcke, 2005). The aim is to maximise

$$F_{\mathrm{MPFE}}(\Lambda) = \sum_{S} P_{\kappa}(S|O,\Lambda)A(S,S_r) \tag{2.5}$$

Label	IPA	Example words
a	ɑ	talo
b	b	banjo
d	d	edes
e	e	etu
f	f	filmi
g	g, ŋ	mega, ongen
h	h	ohi
i	i	tai
j	j	ja
k	k	kai
l	l	oli
m	m	minä
n	n, ŋ	on, onki
o	o	on
p	p	pian
r	r	eri
s	s	sai
t	t	ote
u	u	tuo
v	ʋ	vai
y	y	ylös
ä	æ	tämä
ö	ø	ylös

Table 2.1: The phoneme labels used in the acoustic model.

where $A(S, S_r)$ measures the number of frames having a correct phone label in the recognition hypothesis S when compared to the reference hypothesis S_r. The subscript κ denotes the factor used to scale the probabilities in

$$P_\kappa(S|O,\Lambda) = \frac{P(O|S,\Lambda)^\kappa P(S)^\kappa}{\sum_{S'} P(O|S',\Lambda)^\kappa P(S')^\kappa}. \tag{2.6}$$

Assuming that normal LM scaling is already applied to $P(S)$, a good value for κ is the inverse of the LM scaling factor (Povey, 2003, page 17).

2.3.3 Speaker adaptation

Speaker adaptation is another technique for improving speaker-independent speech recognition. After observing a few sentences from a particular speaker, the general speaker-independent phoneme models can be transformed to match the voice of the speaker better. In this work, the adaptation is done by applying constrained maximum likelihood linear regression (CMLLR) introduced by Digalakis et al. (1995). Instead of adapting the models, the adaptation is performed by transforming the features (Gales, 1998):

$$\hat{o}(t) = Ao(t) + b \tag{2.7}$$

Assuming diagonal covariances in the original model $\mathcal{M}$, the parameters of the transformation can be found by maximising the following equation

$$Q(\mathcal{M}, A, b) = K - \frac{1}{2} \sum_{m=1}^{M} \sum_{t=1}^{T} \gamma_m(t) \Big[K_m + \log |\Sigma_m| - \log(|A|^2) + \big(\hat{o}(t) - \mu_m\big)^T \Sigma_m^{-1} \big(\hat{o}(t) - \mu_m\big) \Big] \quad (2.8)$$

where K is a constant dependent on the transition probabilities, M is the number of Gaussian components, μ_m and Σ_m are the mean and variance for m^{th} Gaussian component, K_m is a normalisation constant for the component. The posterior probability of the component at time t is denoted by $\gamma_m(t)$. An iterative solution to the maximisation problem is presented by Gales (1998).

In addition to using CMLLR during recognition, it can also be applied during training to reduce the inter-speaker variance in the model. This is called speaker adaptive training (SAT) and has been shown to improve the accuracy of speaker-adapted recognition (Anastasakos et al., 1996), even when combined with discriminative training (McDonough et al., 2002).

2.4 Language models

Language models have two purposes in speech recognition systems. Firstly, they define which words can be recognised by the system. This helps to restrict the search space of the recognition problem and makes the recognition more efficient, since the system can ignore phoneme sequences that do not form legal words. Secondly, the language models define a probability distribution $P(W)$ for legal word sequences, which is needed in order to maximise Eq. 2.3.

In this work, the word sequences are modelled using n-gram models either using whole words or word fragments (referred to as *morphs*) as modelling units. Chapter 3 presents the n-gram models in greater detail, and describes how they can be trained and represented during recognition. Morph-based language models are introduced in Chapter 4.

2.5 Decoding

The process of maximising Eq. 2.3, i.e., finding the most probable word sequence given the observations and the models, is often called *decoding*. In theory, we would like to compute the probability for every possible word sequence, and then select the most probable one. However, since the number of possible word sequences is astronomical, only a small fraction of all possible word sequences can be evaluated in practice. During the past two decades, decoding techniques were actively developed to allow fast decoding of continuous speech with large vocabularies. These techniques include token passing (Young et al., 1989), efficient decoding with context-dependent phoneme models (Odell, 1995; Kanthak et al., 2000; Achim Sixtus, 2002), and language model look-ahead (Ortmanns et al., 1996). The recognition system used in this work is based on these techniques, as described in the following sections.

Another approach, based on weighted finite-state transducers (WFSTs), has become popular since the development of efficient composition and optimisation

algorithms (Mohri et al., 2002). The idea is to define separate transducers for all models (HMMs, phoneme contexts, pronunciation dictionary, and language models) and compose them into a single transducer. Since the composition result can be extremely large, it is important that the size of the transducers can be optimised before and after composition. The composition process can be very memory intensive, but is shown to be possible even for large vocabularies (Mohri et al., 2002). Chen et al. (2006) describe recent techniques for making the composition and decoding more efficient while using very long phoneme contexts in acoustic modelling. The advantage of the WFST-based approach is that the decoding can be extremely fast since the language model is completely integrated and optimised in the search network off-line.

2.5.1 Partial hypotheses

The search is performed one frame at a time starting from the first time frame. At each time instant t, the system maintains a list of *partial hypotheses*. Each partial hypothesis corresponds to a HMM state sequence starting from the first time frame and ending at t^{th} frame. The idea is to prune the most improbable state sequences at each frame and extend only the best sequences to the next frame.

For each partial hypothesis h, the decoder maintains the cumulative partial probability $P(h)$ that is updated at time instant t:

$$P(h) \approx P_{\text{AM}}(o_1^t|S_h) \cdot P_{\text{LM}}(W_h)^{\lambda} \cdot P_{\text{LA}}(W_h)^{\lambda} \cdot P_{\text{DUR}}(S_h)^{\delta} \tag{2.9}$$

where o_1^t is the sequence of t first observations, S_h is the state sequence of the hypothesis, and W_h is the sequence of known words in the partial hypothesis. The sequence may end in the middle of the word, so the last word may be unknown at some point. In this case, the LM look-ahead model $P_{\text{LA}}(W_h)$ gives a 2-gram approximation for the probability of the next word. $P_{\text{DUR}}(S_h)$ is a state-duration model (Pylkkönen, 2004). The exponents λ and δ control the weight of the models and are set empirically.

2.5.2 Search network and tokens

During the search process, arbitrary state sequences are not allowed. Instead, the recognition is restricted to state sequences that correspond to valid word sequences present in the language model. If morph-based language models are used, words are formed by joining morphs together and very large vocabularies can be represented by using a small set of morphs (see Chapter 4).

In order to restrict the state sequences during recognition, it is common to form a cyclic state machine, or *search network*, that allows all the legal state sequences but nothing else[1]. Figure 2.3 illustrates a small morph-based network that contains two Finnish morphs: *JA* and *ON*, the word boundary *(W)*, and the end-of-sentence symbol *(S)*. The illustrative network contains 2-state triphone variants for phonemes *j, a, o*, and *n*. Additionally, there is a 2-state model for silence (*sil*), and a 1-state model for short inter-word pause (*sp*). Inside the triphone states, the superscripts indicate the triphone contexts. The numbered transitions at right

[1]For readers familiar with recognisers based on weighted finite-state transducers, the network roughly corresponds to the determinised composition of the HMMs H, triphone contexts C, and pronunciation lexicon L (see Mohri et al., 2002). Thus the n-gram language model is not incorporated in the network.

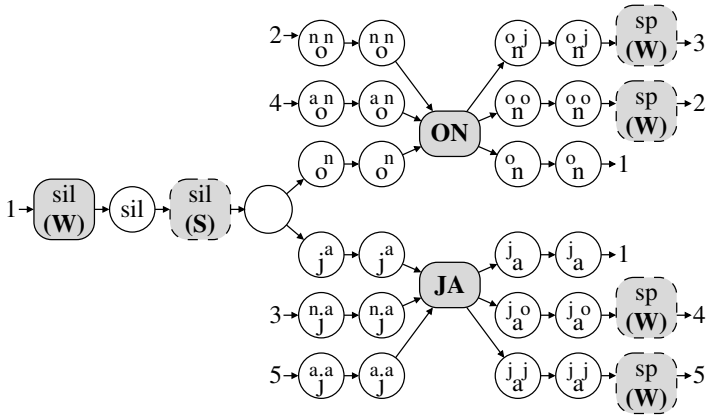

Figure 2.3: A simplified example of a search network that contains Finnish morphs *ON* and *JA*.

are connected to corresponding transitions at left making the network cyclic. The dashed states are optional and can be skipped. For simplicity, each state phoneme variant only contains two states and self transitions are not shown.

Each path through the network defines an HMM state sequence and a morph sequence, which naturally defines a word sequence too. Since the word boundaries in the morph network are optional, the network allows arbitrarily long words. For example, the morph sequence *(W) on (W) ja on (W) ja (W) ja on on ja (W) (S)* corresponds to the following 4-word sentence: *On jaon ja jaononja*. The first three words are actually real Finnish words, but the last one is not.

The search is implemented as a token passing algorithm. Each partial hypothesis is represented as a moveable token. Initially, the network only contains one empty token. At each time instant all tokens are propagated through the transitions and the probabilities of the tokens are updated. If all tokens were preserved, the network would contain tokens for all possible state sequences in the end. Thus, at each time instant only the most probable tokens are preserved and others are pruned. Because the language model is an n-gram model that depends only on a few preceding words, we know that the order of two tokens in the same node can never change in the future, if the word history of the tokens is similar enough.

2.6 Measuring recognition accuracy

The general recognition accuracy of a recognition system is measured by comparing the similarity between the recognition output and the correct reference transcription. The most common metric is the word error rate (WER). It is based on computing the smallest number of word insertions W_I, deletions W_D, and substitutions W_S needed to make the reference transcription identical to the recognition output:

$$\mathrm{WER} = \frac{W_I + W_D + W_S}{W_r} \cdot 100\%. \tag{2.10}$$

W_r is the number of words in the reference transcript. Note that WER may exceed 100 % if the recognition output contains many insertion errors.

Similarly, we can measure the letter error rate (LER) by computing the smallest number of letter insertions L_I, deletions L_D and substitutions L_S needed to make the reference transcription identical to the recognition output:

$$\mathrm{LER} = \frac{L_I + L_D + L_S}{L_r} \cdot 100\% \tag{2.11}$$

where L_r is the number of letters in the reference transcript, counting inter-word spaces as letters. LER may be a more suitable metric for morphologically rich languages. Consider, for example, the following 4-word sentence in Finnish and its English translation:

> Paperitehtaamme huoltokatko on huomenna.
> The service break of our paper mill is tomorrow.

If the system makes a small error in the first word and recognises *Paperitehtaanne* (your paper mill) instead of *Paperitehtaamme* (our paper mill), the word error rate will be 25 % in the Finnish case, but only 11 % in the English case. Rendering the whole Finnish word incorrect seems to be too strict. After all, the recogniser got the paper-mill part right and only made a mistake in the possessive suffix. In this case, the letter error rate (5.1 %) measures the severity of the error better.

Sometimes *word accuracy* or *letter accuracy* is used instead of the corresponding error rates. Accuracy is defined simply as: 100 % minus the error rate. This thesis reports mainly letter error rates, except for Section 4.4 where results from Publication V are reported.

2.7 Evaluation tasks

During the time the publications of this thesis were written, the research recognition system used in the laboratory was constantly developed. Thus, the experimental settings used in the publications have changed as new techniques have been implemented in the system and more training data have become available. The main experimental settings used in the thesis are listed in Table 2.2.

The first task (FinBook1), was used in Publication I, where the first morph-based experiments were reported. The speech corpus consisted of an audio book read by one female speaker, so the recognition system was speaker-dependent. Triphones were already used in this setup, but the method used for tying the triphone states was quite simple: both triphone contexts were modelled if there was enough data for the model, otherwise the dependency on the left or right contexts was dropped, and if the data was still insufficient, a monophone model was used instead. Also, the triphone contexts were modelled within words or morphs only.

The next tasks (FinBook2 and FinNews) were originally used in Publication II. The speech corpus of the FinBook2 was the same audio book that was used in FinBook1, but the partitions used for training, development and evaluation were changed slightly. The FinNews task consisted of a small news corpus read by a female speaker. The amount of LM training data increased slightly, and a global MLLT for the features was implemented in the system. Here, monophones were used instead of triphones in order to make comparisons between different morph segmentation approaches fair, since the old decoder did not support cross-morph triphones. The language models were trained using Kneser-Ney (KN) smoothing

Task	FinBook1	FinBook2	FinNews
Decoder	old	old	old
Text corpus	STT+CSC	CSC	CSC
Words	30 M	40 M	40 M
N-gram	3	2–5	2–5
Smooth.	GT	KN	KN
Vocab.	64 k	69 k, 410 k	69 k, 410 k
Morphs	65 k	26 k, 66 k	26 k, 66 k
Speech corpus	Audio book	Audio book	News
Training	12 h	11 h	3.5 h
Devel.	8 min	20 min	33 min
Test	28 min	27 min	49 min
Test speakers	1	1	1
Acoustic models	triphone	monophone	monophone
Duration	no	yes	yes
Features	13 MFCC+D	13 MFCC+D	13 MFCC+D
Other		MLLT	MLLT
Best LER	7.3	4.21	4.84
Task	FinSP	FinSD	EstSD
Decoder	new	new	new
Text corpus	CSC	CSC	Segakorpus
Words	150 M	150 M	127 M
N-gram	5, EP, KNG	3, KNG	3, KNG
Smooth.	KNG, KN, GT	KNG	KNG
Vocab.		500 k	500 k
Morphs	8.4 k	2 k, 10 k, 50 k	2 k, 10 k, 50 k
Speech corpus	Speecon	Speechdat	Speechdat
Training	26 h	39 h	110 h
Devel.	1 h	46 min	
Test	1.5 h	46 min	
Test speakers	31	79	50
Acoustic models	cw-triphone	cw-triphone	cw-triphone
Duration	yes	yes	yes
Features	13 MFCC+D+DD	13 MFCC+D+DD	13 MFCC+D+DD
Other	MLLT, CMS	MLLT, CMS	MLLT, CMS
Best LER	4.06	6.8[a], 4.8[b]	11.9

[a] Publication VI
[b] with speaker adaptive training and discriminative training (Publication VII)

Table 2.2: Summary of the recognition tasks used in the thesis.

instead of Good-Turing (GT) smoothing (see Chapter 3), and duration modelling was added in HMM states.

Publication III reported experiments on the speaker-independent FinSP task that used the phonetically rich sentences from the Finnish Speecon speech corpus. The amount of LM training data increased to 150 million words, and variable-length n-gram models were trained using growing and pruning algorithms. Tying of triphone states was improved using decision trees (Odell, 1995), and second time-derivatives were added in the features. Also, a new decoder (Pylkkönen, 2005) was used in this task, which enabled modelling full triphone contexts across

word and morph boundaries.

Publication VI and Publication VII reported experiments on the FinSD task. The task is also speaker-independent, but acoustically more difficult than the FinSP task, since the speech in the Finnish SpeechDat corpus is recorded over telephone. Otherwise, the setting is very similar to FinSP task. The corpus consists mostly of news-like sentences. In Publication VI, a similar task (EstSD) is used for Estonian experiments. The language models are trained on Estonian Reference Corpus[2] and acoustic models on a SpeechDat-like corpus (Meister et al., 2003, 2004).

Although not visible directly in Table 2.2, the effect of increasing the amount of LM training data from 40 million words to 150 million words and using the new decoder that properly applies the cross-morph triphones is large; the best LER reported by Kurimo et al. (2006a) for the audio book used in FinBook1 and FinBook2 is 0.95 %.

[2]At the time of writing the thesis, the corpus was available at `http://test.cl.ut.ee/korpused/segakorpus/`

CHAPTER 3
N-GRAM LANGUAGE MODELS

The recognition system described in the previous chapter uses statistical language models for two purposes. Firstly, the search space considered by the decoder can be restricted to phoneme sequences that correspond to words defined by the language model. Secondly, the language model defines probabilities for the word sequences, which helps the recogniser choose between hypotheses that are acoustically similar. This chapter introduces n-gram models, which are the most common statistical language models in speech recognition. After describing the standard training techniques, we focus on how high-order variable-length n-gram models can be trained with growing and pruning methods, and how the models can be efficiently represented in a speech recognition system.

3.1 Notation

In the later sections, a word is denoted by w or v, and a sequence of words by $W = w_1 \cdots w_n = w_1^n$, also called an *n-gram*. The start-of-sentence context is denoted by w_0, and w_* denotes the end of the sentence. When unambiguous, the words preceding w are loosely denoted by $\boldsymbol{h}$, called also the *word history*. By $\hat{\boldsymbol{h}}$ we denote the word history obtained by removing the first word from $\boldsymbol{h}$. For example, with the three-word history $\boldsymbol{h}$ = *out of the* and word w = *box*, we can form the n-grams $\boldsymbol{h}w$ = *out of the box* and $\hat{\boldsymbol{h}}w$ = *of the box*. The number of words in the n-gram $\boldsymbol{h}w$ is denoted by $|\boldsymbol{h}w|$, and $C(\boldsymbol{h}w)$ is the number of times $\boldsymbol{h}w$ occurs in the training data.

3.2 N-gram model

The n-gram models are the most widely used statistical language models in speech recognition. They are often used to model sentences as word sequences. The probability that the user utters a sentence $W = w_1 \cdots w_{|W|}$ can be factored as follows:

$$P(W) = \prod_{i=1}^{|W|} P(w_i | w_0^{i-1}) \cdot P(w_* | w_0^{|W|}) \tag{3.1}$$

where the start-of-sentence context is used to allow distinguishing between the probabilities $P(w|w_0)$ and $P(w)$. An n-gram model approximates the conditional probabilities $P(w_i|w_0^{i-1})$ so that the probability of the word w_i depends on the $(n-1)$ preceding words only:

$$P(w_i | w_0^{i-1}) \approx P(w_i | w_{i-n+1}^{i-1}). \tag{3.2}$$

Given a large text corpus for training the model, i.e. a large set of sentences $T = \{W_1, \ldots, W_{|T|}\}$, it is possible to find the model M that maximises the likelihood

of the training sentences:

$$P_M(T) = \prod_{i=1}^{|T|} P_M(W_i) \tag{3.3}$$

For an n-gram model, the maximum is obtained when

$$P_M(w|\boldsymbol{h}) = \begin{cases} 0, & \text{if } C(\boldsymbol{h}) = 0 \\ \dfrac{C(\boldsymbol{h}w)}{C(\boldsymbol{h})}, & \text{otherwise.} \end{cases} \tag{3.4}$$

This maximum likelihood estimate is the basis for training n-gram models. However, it is not used directly, because it over-learns the training sentences badly: $P(w_i|w_{i-n+1}^{i-1})$ will be zero for all n-grams w_{i-n+1}^{i} that do not appear in the training data. Thus, it is common to use smoothing methods as described in the next section.

3.3 Smoothing n-gram models

Smoothing methods (also called discounting methods) can be used for preventing n-gram models from over-learning the training data. Instead of using the maximum likelihood estimate shown in Eq. 3.4 directly, the model is changed slightly. Some probability mass is taken from the n-grams that have a positive ML estimate and distributed for the n-grams that had zero ML estimate. An extensive empirical study of many smoothing methods is presented by Chen and Goodman (1999). In this work, we consider Good-Turing smoothing, interpolated absolute discounting, and interpolated Kneser-Ney smoothing.

3.3.1 Good-Turing estimate with back-off

The Good-Turing (GT) estimate was originally presented by Good (1953) for estimating frequencies of species in a population from a sample. According to the estimate, for a species occurring r times in the sample, we should use the following modified count:

$$r^* = (r+1)\frac{N_{r+1}}{N_r} \tag{3.5}$$

where N_r is the number of species that occur exactly r times in the sample. The formula is derived by considering the frequency of a species in the sample as a binomially distributed random variable (because of the finite sample) and computing the expectation of the frequency. Later Nádas (1985) showed how to derive the same estimate as an empirical Bayes estimate in the context of linguistic data.

The GT estimate is usually used in a back-off model (Katz, 1987) where lower-order estimates are used whenever the higher-order estimates would give a zero probability. Also, the ML estimates for n-grams occurring more than k times in the training data are considered reliable and the smoothing is only applied for the n-grams occurring k or fewer times. The model can be written as

$$P_{\mathrm{GT}}(w|\boldsymbol{h}) = \begin{cases} d_{C(\boldsymbol{h}w)}\dfrac{C(\boldsymbol{h}w)}{C(\boldsymbol{h})}, & \text{if } C(\boldsymbol{h}w) > 0 \\ \beta(\boldsymbol{h})P_{\mathrm{GT}}(w|\hat{\boldsymbol{h}}), & \text{otherwise} \end{cases} \tag{3.6}$$

where

$$d_r = \begin{cases} 1, & \text{for } r > k \\ \frac{r^*}{r} - \frac{(k+1)N_{k+1}}{N_1}, & \text{for } 1 \leq r \leq k \\ \hline 1 - \frac{(k+1)N_{k+1}}{N_1} \end{cases} \tag{3.7}$$

and $\beta(\boldsymbol{h})$ is a normalisation constant that ensures that probability distributions sum to one.

3.3.2 Interpolated absolute discounting

Another standard technique for smoothing n-gram models is called interpolated absolute discounting (AD), originally proposed by Ney and Essen (1991). In contrast to back-off models, where the lower-order estimates are only considered when the higher-order estimates are zero, the estimates are always interpolated down to 1-grams. The ML estimates are smoothed by subtracting a constant D from the n-gram count:

$$P_{\text{AD}}(w|\boldsymbol{h}) = \frac{\max\left\{0, C(\boldsymbol{h}w) - D\right\}}{C(\boldsymbol{h})} + \gamma(D, \boldsymbol{h}) \cdot P_{\text{AD}}(w|\hat{\boldsymbol{h}}) \tag{3.8}$$

$$\gamma(D, \boldsymbol{h}) = \sum_v \frac{\min\left\{C(\boldsymbol{h}v), D\right\}}{C(\boldsymbol{h})} \tag{3.9}$$

The discounting term D determines how much probability mass is removed from the ML distribution, and the factor $\gamma(D, \boldsymbol{h})$ ensures that the corresponding amount of probability mass is interpolated from the lower-order estimate. The value of D can be chosen by maximising the likelihood of some held-out sentences, or by using the estimate provided by (Ney et al., 1994):

$$D = \frac{N_1}{N_1 + 2N_2}. \tag{3.10}$$

In this work, AD is not used directly as a smoothing method, but it forms the basis for Kneser-Ney smoothing presented in the next section.

3.3.3 Interpolated Kneser-Ney smoothing

The state-of-the-art smoothing method is Kneser-Ney (KN) smoothing or, more specifically, its interpolated version (Chen and Goodman, 1999). KN smoothing is otherwise identical to absolute discounting, but the lower-order estimates are modified. The idea is to try satisfying the marginal constraints

$$\sum_v P_M(v\boldsymbol{h}w) = P(\boldsymbol{h}w) \quad \text{for all } \boldsymbol{h} \text{ and } w \tag{3.11}$$

which are destroyed by absolute discounting. For the distribution $P(\cdot|\boldsymbol{h})$, the constraints result in the following formulas:

$$P_{\mathrm{KN}}(w|\boldsymbol{h}) = \begin{cases} P_{\mathrm{KN}}(w|\hat{\boldsymbol{h}}), & \text{if } \sum_v C'(\boldsymbol{h}v) = 0 \\ \dfrac{\max\{0, C'(\boldsymbol{h}w) - D\}}{\sum_v C'(\boldsymbol{h}v)} + \gamma(D, \boldsymbol{h}) \cdot P_{\mathrm{KN}}(w|\hat{\boldsymbol{h}}), & \text{otherwise} \end{cases} \tag{3.12}$$

$$\gamma(D, \boldsymbol{h}) = \sum_v \frac{\min\{C'(\boldsymbol{h}v), D\}}{\sum_v C'(\boldsymbol{h}v)} \tag{3.13}$$

where the modified counts $C'(\cdot)$ for an N-gram model are

$$C'(\boldsymbol{h}w) = \begin{cases} 0, & \text{if } |\boldsymbol{h}w| > N \\ C(\boldsymbol{h}w), & \text{if } |\boldsymbol{h}w| = N \\ |\{v : C(v\boldsymbol{h}w) > 0\}|, & \text{otherwise} \end{cases} \tag{3.14}$$

In other words, the actual count of n-gram $\boldsymbol{h}w$ is replaced by the number of distinct words v that have been observed before $\boldsymbol{h}w$. While the formula follows from the marginal constraints, it can also be justified intuitively as follows. For example, in some corpus, the words *Francisco* and *controls* might be equally frequent. Using the unmodified counts, their probability in a new context (after the word *atmosphere*, for example) would be equal. However, in the training data we may see that the word *controls* occurs in many different contexts (*it controls, doping controls, which controls, easily controls*), while the word *Francisco* mostly occurs in a single context (*San Francisco*). Thus, it is reasonable to assign a higher probability for *atmosphere controls* than *atmosphere Francisco* if we have not seen either of the 2-grams in the training data.

3.4 Variable-length N-gram models

The n-gram models easily become very large when large training corpora are used, since for each history-pair $\boldsymbol{h}w$ appearing in the training data, it is necessary to store at least the count $C(\boldsymbol{h}w)$ or a pre-computed probability. Also, the interpolation weights $\gamma(D, \boldsymbol{h})$ are typically pre-computed for all histories $\boldsymbol{h}$. In order to keep the models smaller, it is possible to prune the model. The idea is to remove the explicit estimates for $P(w|\boldsymbol{h})$ if the interpolated lower-order estimate $P(w|\hat{\boldsymbol{h}})$ provides a good approximation.

Several approaches have been proposed in the literature. Ristad and Thomas (1995) consider letter n-gram models and describe a greedy algorithm for growing the model. Riccardi et al. (1996) describe how n-gram models can be presented as stochastic automata and propose an algorithm for pruning states that correspond to infrequent n-gram contexts. Siu and Ostendorf (2000) present the n-gram model as a tree and propose ways to combine the nodes of the tree. In the following sections, we concentrate on entropy-based pruning (Stolcke, 1998), Kneser pruning (Kneser, 1996) and their extensions presented in Publication III.

3.4.1 Entropy-based pruning

Stolcke (1998) presented the entropy-based pruning (EP) algorithm for n-gram language models. Being implemented in the popular SRILM toolkit (Stolcke, 2002),

it is widely used. For each n-gram $\boldsymbol{h}w$ in the model M, the pruning cost $d(\boldsymbol{h}w)$ is computed as follows:

$$d(\boldsymbol{h}w) = \sum_v P_M(\boldsymbol{h}w) \log \frac{P_M(v|\boldsymbol{h})}{P_{M'}(v|\boldsymbol{h})} \tag{3.15}$$

where P_M is the original model and $P_{M'}$ is the model with the n-gram $\boldsymbol{h}w$ removed. First a pruning threshold is fixed, the cost is computed for each n-gram in the model, and then all n-grams whose cost fall below the threshold are removed.

3.4.2 Kneser pruning

Kneser (1996) also proposes a general algorithm for pruning back-off n-gram models. As in entropy-based pruning, the cost of pruning an n-gram is computed for each n-gram in advance. For an n-gram $\boldsymbol{h}w$ that is not a prefix of any higher-order n-gram in the model (a *leaf n-gram*), the cost is defined as:

$$d_1(\boldsymbol{h}w) = P_M(\boldsymbol{h}w) \log \frac{P_M(w|\boldsymbol{h})}{\gamma_M(\boldsymbol{h}) P_M(w|\hat{\boldsymbol{h}})} \tag{3.16}$$

where M is the full unpruned model. The cost for a *non-leaf n-gram*, on the other hand, is computed by averaging $d_1(\boldsymbol{h})$ for n-grams $\boldsymbol{h}$ that have $\boldsymbol{h}w$ as prefix, including $g = \boldsymbol{h}w$.

Kneser also gives a modified back-off distribution for satisfying the marginal constraints of KN smoothing after pruning:

$$\beta(w|\boldsymbol{h}) \approx \frac{C(\boldsymbol{h}w) - \sum_{v: v\boldsymbol{h}w \in S} \big[C(v\boldsymbol{h}w) - D\big]}{\sum_{w'} \left(C(\boldsymbol{h}, w') - \sum_{v: v\boldsymbol{h}w' \in S} \big[C(v\boldsymbol{h}w') - D\big] \right)} \tag{3.17}$$

where S is the set of n-grams included (i.e. not pruned) in the model. In later sections, this algorithm is referred to as Kneser pruning (KP).

3.4.3 Revised Kneser pruning

The above pruning algorithms have a few shortcomings when they are used with KN smoothing. The EP algorithm does not consider the marginal constraints at all, and Kneser pruning modifies the back-off distributions only after the model has been pruned. Publication III introduces a new pruning algorithm, called *revised Kneser pruning* (RKP), which aims to preserve the marginal constraints already during the pruning process. Starting from an absolute discounted model and taking the pruned n-grams and marginal constraints into account leads to the following model:

$$P_{\mathrm{RKP}}(w|\boldsymbol{h}) = \frac{\max\big\{0, C'(\boldsymbol{h}w) - D\big\}}{S(\boldsymbol{h}) + L(\boldsymbol{h})} + \gamma(D, \boldsymbol{h}) \cdot P_{\mathrm{KN}}(w|\hat{\boldsymbol{h}}) \tag{3.18}$$

$$\gamma(D, \boldsymbol{h}) = \frac{\sum_v \min\big\{C'(\boldsymbol{h}v), D\big\} + L(\boldsymbol{h})}{S(\boldsymbol{h}) + L(\boldsymbol{h})} \tag{3.19}$$

$$S(\boldsymbol{h}) = \sum_v C'(\boldsymbol{h}v) \tag{3.20}$$

where $L(\boldsymbol{h})$ is the sum of pruned counts and is initially set to zero for all $\boldsymbol{h}$. The detailed derivation of the formulas is given by Siivola (2007, page 62).

Figure 3.1 describes an algorithm for updating the parameters during pruning. PruneGram($\boldsymbol{h}w$) shows how C', L, and S are modified whenever an n-gram $\boldsymbol{h}w$ is removed from the model. PruneOrder(k, ϵ) shows how each n-gram order k is processed starting from the highest one. The algorithm tries to prune each n-gram of the given order at a time, and if the likelihood of the training data decreases more than the pruning threshold ϵ, the n-gram is put back to the model.

3.4.4 Kneser-Ney growing

One disadvantage of the above pruning algorithms is that a full unpruned n-gram model is required before n-grams can be pruned. If high-order n-grams are desired, the memory requirements can be too high for creating the full model. In order to attack this problem, Publication III also describes a growing algorithm that maintains the same properties as the revised Kneser pruning algorithm. The algorithm is called *Kneser-Ney growing* (KNG). Initially, the model is only a 1-gram model, and the algorithm starts adding 2-grams to the model. For each n-gram $\boldsymbol{h}$ present in the model, the algorithm tries adding all $(n+1)$-grams $\boldsymbol{h}w$ that occur in the training data. The cost of the change is evaluated and if the criterion is satisfied, the new n-grams are left in the model. Otherwise, they are all removed. After all 1-grams have been processed, the algorithm starts processing 2-grams $\boldsymbol{h}$, and tries adding 3-grams $\boldsymbol{h}w$. Figure 3.2 describes the algorithm in detail. After the model is grown as large as possible, it can be pruned to the desired size using the RKP algorithm.

3.4.5 Experimental evaluation

When developing language models for speech recognition, it is customary to measure the general modelling performance of the models by computing the probability of held-out text corpus for each model. The commonly used measures are the cross-entropy $H(T|M)$ and perplexity $\mathrm{Perp}(T|M)$ between the test corpus T and the model M. Since the size of the test corpus affects the probability given by the model, the measures are usually normalised by the number of words W_T in the corpus in order to make results on different corpora more comparable:

$$H(T|M) = -\frac{1}{W_T} \log_2 P(T|M) \tag{3.21}$$

$$\mathrm{Perp}(T|M) = P(T|M)^{-\frac{1}{W_T}} \tag{3.22}$$

The performance of the pruning and growing algorithms was evaluated in Publication III. First, a 150-million-word Finnish text corpus was used to train two full 5-gram models: one using Good-Turing smoothing, and another using Kneser-Ney smoothing. These models were then pruned using the EP, KP, and RKP algorithms. Additionally, one model was grown by using the KNG algorithm. The growing and pruning parameters were set so that four differently sized models were created for each algorithm: a small model (around 2 million n-grams), a medium model (around 11 million n-grams), a large model (around 70 million n-grams), and a full model (around 191 million n-grams).

Figure 3.3 shows the cross-entropies for the models. First, looking at the full 5-gram models, we can see that KN smoothing results in a better model than

PruneOrder(k, ϵ)

1 **for** $\{\boldsymbol{h}w : |\boldsymbol{h}w| = k \wedge C'(\boldsymbol{h}w) > 0\}$ **do**
2 $\quad$ $logprob_0 \leftarrow C(\boldsymbol{h}w) \log_2 P_{\mathrm{KN}}(w|\boldsymbol{h})$
3 $\quad$ PruneGram$(\boldsymbol{h}w)$
4 $\quad$ $logprob_1 \leftarrow C(\boldsymbol{h}w) \log_2 P_{\mathrm{KN}}(w|\boldsymbol{h})$
5 $\quad$ **if** $logprob_1 < logprob_0 - \epsilon$
6 $\quad\quad$ undo previous PruneGram

PruneGram$(\boldsymbol{h}w)$

1 $L(\boldsymbol{h}) \leftarrow L(\boldsymbol{h}) + C'(\boldsymbol{h}w)$
2 **if** $C'(\hat{\boldsymbol{h}}w) > 0$
3 $\quad$ $C'(\hat{\boldsymbol{h}}w) \leftarrow C'(\hat{\boldsymbol{h}}w) + C'(\boldsymbol{h}w) - 1$
4 $\quad$ $S(\hat{\boldsymbol{h}}) \leftarrow S(\hat{\boldsymbol{h}}) + C'(\boldsymbol{h}w) - 1$
5 $C'(\boldsymbol{h}w) \leftarrow 0$

Figure 3.1: The pruning algorithm. Note that the lines 3 and 6 in PruneOrder modify the counts $C'(\cdot)$, which also alters the estimate $P_{\mathrm{KN}}(w|\boldsymbol{h})$ used on the lines 2 and 4.

GrowOrder(k, δ)

1 **for** $\{\boldsymbol{h} : |\boldsymbol{h}| = k - 1 \wedge C'(\boldsymbol{h}) > 0\}$ **do**
2 $\quad$ $size_0 \leftarrow \left|\{g : C'(g) > 0\}\right|$
3 $\quad$ $logprob_0 \leftarrow 0$
4 $\quad$ **for** $w : C(\boldsymbol{h}w) > 0$ **do**
5 $\quad\quad$ $logprob_0 \leftarrow logprob_0 + C(\boldsymbol{h}w) \log_2 P_{\mathrm{KN}}(w|\boldsymbol{h})$
6 $\quad$ **for** $w : C(\boldsymbol{h}w) > 0$ **do**
7 $\quad\quad$ AddGram$(\boldsymbol{h}w)$
8 $\quad$ $size_1 \leftarrow |\{g : C'(g) > 0\}|$
9 $\quad$ $logprob_1 \leftarrow 0$
10 $\quad$ **for** $w : C(\boldsymbol{h}w) > 0$ **do**
11 $\quad\quad$ $logprob_1 \leftarrow logprob_1 + C(\boldsymbol{h}w) \log_2 P_{\mathrm{KN}}(w|\boldsymbol{h})$
12 $\quad$ $logscost = size_1 \log_2(size_1) - size_0 \log_2(size_0)$
13 $\quad$ $sizecost \leftarrow (size_1 - size_0)\alpha + logscost$
14 $\quad$ **if** $logprob_1 - logprob_0 - \delta \cdot sizecost \leq 0$
15 $\quad\quad$ undo previous AddGram$(\boldsymbol{h}w)$ for each w
16 re-estimate all discount parameters D_i

AddGram$(\boldsymbol{h}w)$

1 $C'(\boldsymbol{h}w) \leftarrow C(\boldsymbol{h}w)$
2 $S(\boldsymbol{h}) \leftarrow S(\boldsymbol{h}) + C(\boldsymbol{h}w)$
3 **if** $C'(\hat{\boldsymbol{h}}w) > 0$
4 $\quad$ $C'(\hat{\boldsymbol{h}}w) \leftarrow C'(\hat{\boldsymbol{h}}w) - C(\boldsymbol{h}w) + 1$
5 $\quad$ $S(\hat{\boldsymbol{h}}) \leftarrow S(\hat{\boldsymbol{h}}) - C(\boldsymbol{h}w) + 1$

Figure 3.2: The growing algorithm.

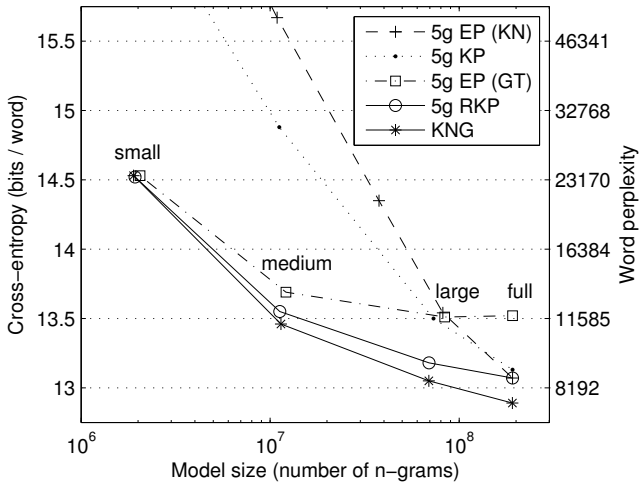

Figure 3.3: Comparing cross-entropies and perplexities of variable-length n-gram models on a Finnish text corpus. Abbreviations: entropy-based pruning (EP), Kneser pruning (KP), revised Kneser pruning (RKP), Kneser-Ney growing (KNG), Good-Turing smoothing (GT), and Kneser-Ney smoothing (KN).

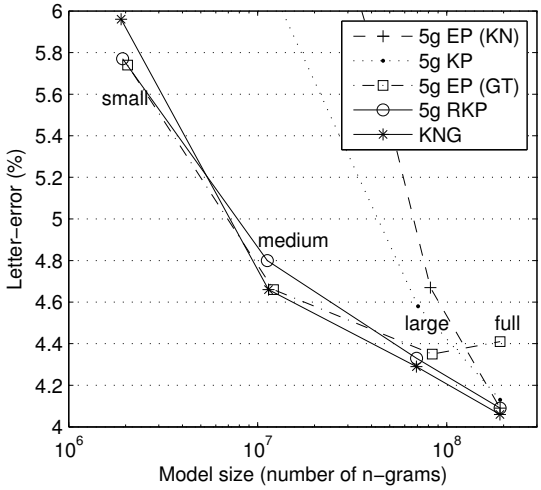

Figure 3.4: Evaluating variable-length n-gram models in a Finnish speech recognition task. Abbreviations: entropy-based pruning (EP), Kneser pruning (KP), revised Kneser pruning (RKP), Kneser-Ney growing (KNG), Good-Turing smoothing (GT), and Kneser-Ney smoothing (KN).

GT smoothing, as expected. However, if the KN smoothed model is pruned with with EP or KP, the performance degrades rapidly and the models become worse than the GT smoothed model pruned with EP. The RKP algorithm on the other hand maintains the performance difference to GT smoothed model. Only at the smallest model size, when only 1 % of the n-grams of the full model remain, the performance difference disappears. The best results are obtained with the KNG algorithm, where the order of the n-grams is not limited to 5-grams, as it was with the pruned models.

The models were also evaluated in speech recognition experiments using the FinSP task. Figure 3.4 shows the letter error rates. Here, again, we can see that the KN smoothing gives better results than GT smoothing for the full 5-gram models, and that pruning KN smoothed model with EP or KP degrades the results rapidly. At the small, medium and large model sizes, however, the models trained with RKP, KNG and EP (with GT smoothing) perform equally. The results suggest that, in contrast to the previous pruning methods, the proposed pruning and growing methods give good results regardless of the size of the model.

3.5 N-gram data structures

Since high-order n-gram models are often very large, it is not insignificant what kind of data structures are used to store them during recognition. As usual, the choice of the data structure is a trade-off between speed and memory consumption.

In this section, it is assumed that the language model is represented in the common back-off format (Katz, 1987). A back-off model M can be defined as a tuple (V, G, α, β'), where

- V is the set of units used by the model (usually words or subword units).
- G is the set of n-grams explicitly stored in the model.
- $\alpha : G \to \mathbb{R}$ is a table of the logarithmic probability estimates for the n-grams stored in the model: $\alpha(\boldsymbol{h}w) = P(w|\boldsymbol{h})$.
- $\beta' : G \to \mathbb{R}$ is a table of logarithmic back-off weights of the n-grams stored in the model, i.e. $\beta'(\boldsymbol{h}) = \log \beta(\boldsymbol{h})$ (see Eq. 3.6).

Recalling the Good-Turing smoothed back-off n-gram model (Section 3.3.1), the logarithm of the conditional probability $P(w|\boldsymbol{h})$ is computed recursively as follows:

$$\log P_M(w|\boldsymbol{h}) = \begin{cases} \alpha(\boldsymbol{h}w) & \text{if } \boldsymbol{h}w \in G, \\ \beta'(\boldsymbol{h}) + \log P_M(w|\hat{\boldsymbol{h}}) & \text{if } \boldsymbol{h}w \notin G \text{ and } \boldsymbol{h} \in G, \\ \log P_M(w|\hat{\boldsymbol{h}}) & \text{if } \boldsymbol{h}w \notin G \text{ and } \boldsymbol{h} \notin G. \end{cases} \tag{3.23}$$

Note that the above formulation differs from the interpolated formulation. Instead of interpolating all orders as in interpolated absolute discounting (Eq. 3.8) and interpolated Kneser-Ney smoothing (Eq. 3.12), the recursion is stopped on the first n-gram present in G. However, interpolated models can easily be pre-computed in the back-off format, which is more efficient than computing the sum on demand.

3.5.1 Compact tree structure

Whittaker and Raj (2001a) propose a compact data structure for storing back-off n-gram models (see also Whittaker and Raj, 2001b; Raj and Whittaker, 2003). The

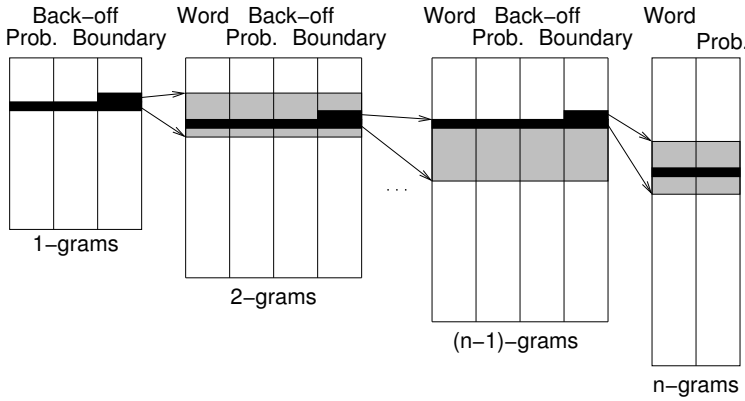

Figure 3.5: The compact tree structure.

structure utilises the observation that once the language model has been trained, it is usually not necessary to alter the structure afterwards. Thus, the structure can be compressed at the expense of making the modifying operations slower.

Figure 3.5 illustrates the structure. The parameters of the model are stored in tables that represent different n-gram orders, and each table contains 2–4 arrays. Each row in the n^{th} table correspond to an n-gram $w_1 \cdots w_n$ and contains the following fields:

1. The index of the word w_n. This field is optional for 1-grams, if we assume that the table contains a row for each word in the vocabulary.

2. The pre-computed log-probability $\alpha(w_1^n)$.

3. The logarithm of the back-off weight $\beta'(w_1^n)$. By the definition of the back-off model, this array is not required for the highest order.

4. A boundary index that forms a tree structure. It indicates the index of the row (in the next table) that follows the n-grams that have w_1^n as prefix.

Consider, for example, finding $\alpha(\textit{in, front, of, the})$ from the structure shown in the figure. We start from the first table, and move to the row r that corresponds to the index of the word *in* (the black row in the figure). Boundary values on the rows $r-1$ and r define the range that contains the 2-grams $(\textit{in}, \cdot)$ in the second table (the gray rows in the figure). From the range, we search the row r that corresponds to the word *front*. The ranges are sorted according to word indices, which allows efficient binary search. Again we look at the boundary values at rows $r-1$ and r to find the range in the next table. Proceeding further, the desired field $\alpha(\textit{in, front, of, the})$ is finally found in the fourth table.

The main benefit of the structure is that the tree structure is formed very compactly because the n-grams are sorted. Only one boundary value is needed for each n-gram, and still the ranges of the child n-grams in the next table can be defined.

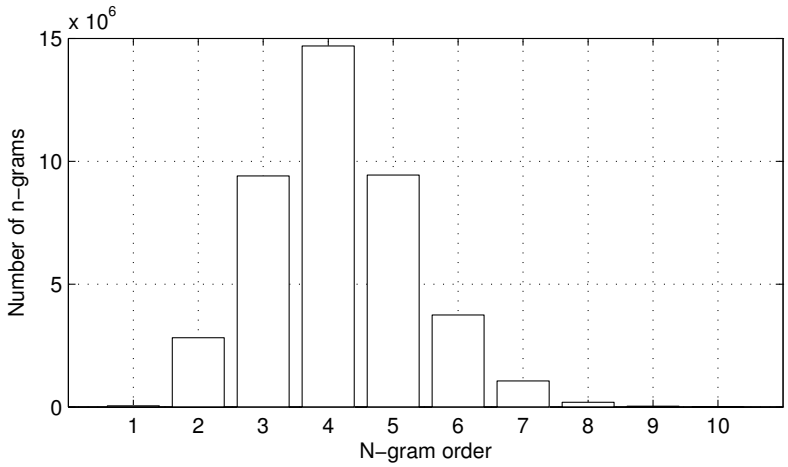

Figure 3.6: A typical n-gram distribution of a variable-length model. The model is the largest morph model used in Publication VI. The morph set contains 50 000 morphs.

3.5.2 Compressing the fields

The total size of the tables depends heavily on the number of bits used to represent the fields in the arrays. Within an array, the same number of bits must be used for each row in order to allow efficient random access to the array, but another array in the same table may have a different bit width. For the integer arrays (word index and boundary arrays), the most straightforward way is to select the bit width that is enough to store the largest value in the array. For example, if the largest value in a word index array is 8 428, and the largest value in the boundary array is 2 487 982, the width of the word index array will be 14 bits, and the width of the boundary array will be 22 bits ($\log_2 8428 \approx 13.04$ and $\log_2 2487982 \approx 21.2$).

Further compression can be obtained by noting that the values in a boundary array are always sorted in ascending order. Raj and Whittaker (2003) describe a general method for compressing sorted integer array. The computational cost of accessing a compressed array is not analysed in the original paper, but the access time should be roughly logarithmic with respect to the length of the array. The word index arrays are not sorted as a whole, so the compression can not be applied directly. However, each contiguous range corresponding to n-grams that share the same context are sorted, and it is possible to apply the compression to these ranges separately (Raj and Whittaker, 2003). In this thesis, the word index arrays are left uncompressed to keep the structure simpler.

Probabilities and back-off weights are usually represented as 32-bit floating point values in computer systems. Language models used in speech recognition, however, do not require that high accuracy. Whittaker and Raj (2001a) showed that the probabilities and back-off weights can be quantised even down to 8 bits before the quantisation degrades speech recognition results.

3.5.3 Extension for variable-length models

Recall from Section 3.4.2, that an n-gram $\boldsymbol{h}$ is called a leaf n-gram if the model does not contain a higher-order child n-gram $\boldsymbol{h}w$. From the definition of the back-

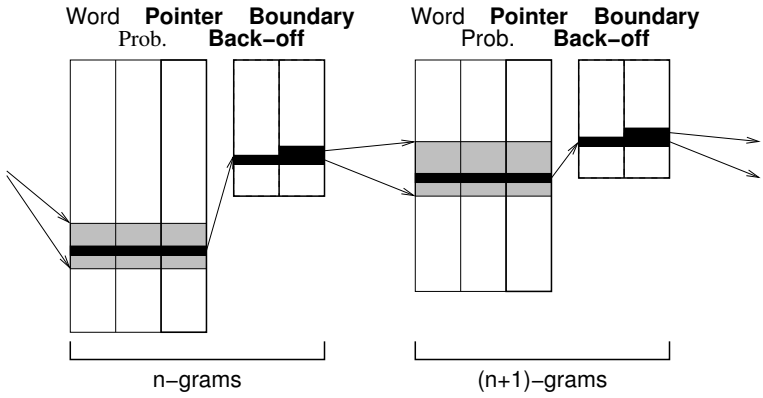

Figure 3.7: The extended tree structure.

off model (Eq. 3.23), it can be seen that the back-off weight is not needed for leaf n-grams. Also, the boundary values are, in a way, redundant since the leaf n-grams do not have any child n-grams in the next table. A full unpruned n-gram model typically contains leaf n-grams on the highest order only, because if the model has not been pruned, it contains all the n-grams that occurred in the training data. Thus, the compact data structure that omits the back-off weights and boundary values for the highest order, is well suited for full unpruned n-gram models.

When pruning and growing algorithms are used to obtain variable-length n-gram models, the distribution of leaf n-grams changes. Figure 3.6 shows a typical n-gram distribution for a variable-length morph model. The highest orders are heavily pruned and the n-grams are concentrated on the middle orders. This means that the leaf n-grams are also concentrated in the middle orders.

Publication IV extends the compact tree structure presented above so that the redundant back-off weights and boundary values can be omitted from all leaf n-grams. Figure 3.7 illustrates the idea. Instead of storing back-off weights and boundary values directly in the main table for every n-gram, they are stored in an auxiliary table, but only for the non-leaf n-grams. A new pointer array is introduced in the main table. For each non-leaf n-gram in the main table, the pointer field indicates the row that contains the back-off weight and the boundary value in the auxiliary table. For leaf n-grams the pointer field is copied from the previous row, which indicates that no auxiliary information is needed for the n-gram. Since the values in the pointer array are naturally in sorted order, the general integer compression described in Section 3.5.2 can be applied.

The usefulness of the extended structure depends on whether the saving obtained by omitting the back-off weights and boundary values in the auxiliary table is larger than the new pointer array introduced in the main table. That depends on the number of leaf n-grams, the bit-width of the fields, and whether integer compression is used. It may be that for some orders the baseline structure is more efficient, if the number of leaf n-grams is low on that particular order. Thus, the best structure can be chosen separately for each order.

Pruning threshold	Integer compression	Float bits	Baseline [MB]	Extended [MB]	Saving [%]
10^{-7}	yes	8	8.8	7.2	18.3
10^{-8}	yes	8	52.4	44.8	14.4
10^{-9}	yes	8	201.8	190.3	5.7
none	yes	8	407.3	407.0	0.1
10^{-7}	no	8	12.8	11.9	6.9
10^{-8}	no	8	78.8	76.2	3.3
10^{-9}	no	8	303.2	303.2	0.0
none	no	8	580.5	580.5	0.0
10^{-7}	yes	16	13.0	9.8	24.8
10^{-8}	yes	16	76.8	60.9	20.6
10^{-9}	yes	16	293.2	261.7	10.8
none	yes	16	582.4	581.5	0.1
10^{-7}	no	16	17.0	14.6	14.2
10^{-8}	no	16	103.2	94.2	8.7
10^{-9}	no	16	394.5	387.3	1.8
none	no	16	755.6	755.6	0.0
10^{-7}	yes	32	21.5	15.0	30.1
10^{-8}	yes	32	125.6	93.2	25.8
10^{-9}	yes	32	475.9	402.3	15.5
none	yes	32	932.6	904.7	3.0
10^{-7}	no	32	25.5	19.8	22.2
10^{-8}	no	32	152.0	126.8	16.6
10^{-9}	no	32	577.3	543.8	5.8
none	no	32	1105.8	1105.4	0.0

Table 3.1: Comparing the sizes of the baseline and extended structures applied to Finnish 6-gram models.

3.5.4 Evaluating the compact structures

In Publication IV, the efficiency of the baseline and extended data structures was evaluated on Finnish and English n-gram models. The Finnish 6-gram model was trained on a 145-million-word text corpus (same as in the FinSP task). Before training the model, the words in the corpus were segmented into morphs (see Chapter 4) so that the words could be formed by using 8428 distinct morphs. The English 4-gram model was trained using 927 million words from the New York Times partition of the Gigaword corpus. The vocabulary was limited to 50 000 words.

Entropy-based pruning was used to produce variable-length models for both languages (pruning thresholds $10^{-9}, 10^{-8}$ and 10^{-7}). The SRILM toolkit (Stolcke, 2002) was used for training and pruning all the models. The Finnish models were trained with the default cut-off values (all n-grams occurring only once were ignored for orders 3 and above), whereas, in the English models, 4-grams occurring only twice were also ignored.

Tables 3.1 and 3.2 show the memory footprint of the models in various conditions. First, it can be seen that the more entropy pruning is applied, the more the extended structure helps to compress the models. This is expected since pruning

Pruning threshold	Integer compression	Float bits	Baseline [MB]	Extended [MB]	Saving [%]
10^{-7}	yes	8	14.3	11.9	17.2
10^{-8}	yes	8	128.5	116.6	9.2
10^{-9}	yes	8	463.3	418.9	9.6
none	yes	8	637.8	582.0	8.7
10^{-7}	no	8	20.7	19.1	7.8
10^{-8}	no	8	183.4	183.4	0.0
10^{-9}	no	8	666.7	666.7	0.0
none	no	8	920.6	920.6	0.0
10^{-7}	yes	16	20.7	15.8	23.7
10^{-8}	yes	16	182.5	156.3	14.4
10^{-9}	yes	16	654.8	558.9	14.6
none	yes	16	899.5	776.5	13.7
10^{-7}	no	16	27.1	23.1	15.0
10^{-8}	no	16	237.5	229.5	3.4
10^{-9}	no	16	858.2	829.5	3.3
none	no	16	1182.4	1150.8	2.7
10^{-7}	yes	32	33.5	23.7	29.2
10^{-8}	yes	32	290.7	235.5	19.0
10^{-9}	yes	32	1037.7	838.8	19.2
none	yes	32	1423.0	1165.5	18.1
10^{-7}	no	32	39.9	31.0	22.4
10^{-8}	no	32	345.7	310.9	10.1
10^{-9}	no	32	1241.2	1109.4	10.6
none	no	32	1705.8	1539.8	9.7

Table 3.2: Comparing the sizes of the baseline and extended structures applied to English 4-gram models.

removes leaf n-grams from the highest orders and creates new leaf n-grams on the middle orders, and the extended structure was specifically designed to compress the leaf n-grams. The reader may wonder why the extended structure manages to compress the full English models, while, in the Finnish case, only negligible savings are obtained. The reason is the higher cut-off values used for training the English models, which makes even the full models somewhat pruned. The effect of cut-offing can be seen in the Finnish case: the full model benefits from the extended structure slightly (3 % saving) if 32-bit floats and integer compression is used.

Another trend in the results seems to be that, the fewer bits are used to store the floating point values (probabilities and back-off weights), the smaller relative savings are obtained with the extended structure. The effect of the floating point quantisation is twofold. If the back-off weights were heavily quantised in the first place, no big savings can be expected by getting rid of the redundant back-off weights. On the other hand, if the explicit probability estimates are also heavily quantised, the overall size of the model is originally smaller, and the possible savings are relatively larger.

Finally, the use of integer compression seems to benefit the extended struc-

Pruning	Baseline [s]	Extended [s]	Overhead [%]	Compression
10^{-7}	85.37	102.62	20.21	yes
10^{-8}	103.48	143.88	39.04	yes
10^{-9}	133.69	185.29	38.60	yes
none	171.55	248.49	44.85	yes
10^{-7}	28.41	30.16	6.16	no
10^{-8}	36.55	38.45	5.20	no
10^{-9}	44.08	44.47	0.88	no
none	47.15	47.81	1.40	no

Table 3.3: Computational cost of the structures when computing perplexity of a Finnish 149-thousand-word corpus using the models with 32 bits per float and repeating the computation 10 times.

ture considerably. This can be explained by the fact that the pointer arrays are naturally in sorted order and the values grow slowly. It seems that the integer compression algorithm is very efficient on this kind of arrays.

As expected, the savings are not completely free. Table 3.3 shows the results on using the structures to compute the perplexity of a Finnish 149-thousand-word corpus ten times. If integer compression is not used, the extended structure does not create much overhead, but with compression the overhead is larger. It should be noted that the algorithms were quite straightforwardly implemented and the run-time performance was not carefully optimised in this work. Thus, the results shown in Table 3.3 are only suggestive. In speech recognition the overhead will be smaller, since computing the language model probabilities is only a small part of all computation needed.

3.5.5 Finite-state structure

Back-off n-gram models can also be represented as a finite-state automaton as illustrated on the left side of Figure 3.8. The model is a variable-length 3-gram model containing the following 11 n-grams:

- 1-grams: *a, c, E*
- 2-grams: *ab, ac, cc, Sa, Sb*
- 3-grams: *abE, acc, SaE.*

S denotes the start-of-sentence context and E denotes the end-of-sentence symbol. In the figure, the initial state corresponding to S is drawn bold. All arcs are weighted, the solid transitions corresponding to n-gram probabilities and dashed transitions to back-off weights. The right side of the figure shows a compacted version of the automaton. The dotted states can be removed since there were only back-off transitions leaving them originally. The incoming n-gram transitions can be pointed to the back-off target directly.

The advantage of the finite-state representation lies in decoding. Each token can maintain a pointer to the state that corresponds to the token's LM context. Every time a token reaches a word identity, the pointer can be updated by following the transitions in the LM automaton, using the back-off transition when necessary,

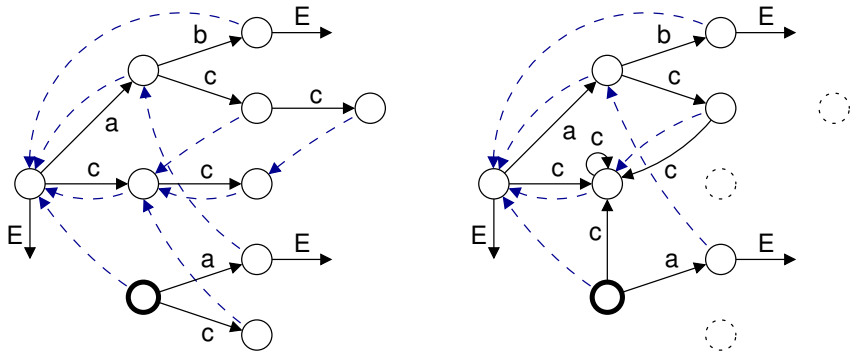

Figure 3.8: Representing an n-gram model as a state machine .

and the token's LM score can be updated efficiently. The pointers make it also efficient to find out when two tokens have similar LM history. If they are in the same LM state, it is known that the future LM scores will never change the order of the tokens, so only the better one needs to be preserved. This is especially good for morph-based variable-length models, since the context lengths used by the model can differ greatly depending on the context. This has been the default structure in the recognition system since Publication VI.

CHAPTER 4
MORPH-BASED LANGUAGE MODELS

4.1 The OOV problem in morphologically rich languages

The usual way to model languages in speech recognition, especially for English, is based on words. Using large training text corpora, the frequencies of words are computed, and the most common words of the language are selected to be included in the model vocabulary. Typically, the vocabulary contains 20 000–60 000 words, which is enough to cover the commonly used words in English. The size of the vocabulary is limited for two reasons: The n-gram model trained on the corpus will be smaller, if the rare words can be omitted from the model, and the overall search space will be smaller leading to more efficient recognition. The main disadvantage of limiting the vocabulary is that the words outside the vocabulary can not be recognised at all. The effect of these out-of-vocabulary (OOV) words is twofold. Firstly, if a word can not be recognised correctly, it will be missed or replaced by another word or words present in the vocabulary, and recognition errors will occur. Secondly, since the LM considers dependencies between the words, the erroneous word may cause other words to be recognised incorrectly.

The number of words needed to cover the common words adequately depends greatly on the language. A typical vocabulary used in English large-vocabulary speech recognition contains around 20 000–65 000 words depending on the task, but in Finnish, for example, a larger vocabulary is needed. The coverage of a vocabulary is often measured by computing so called OOV rate, which measures how often we encounter an OOV word in an independent text corpus that has not been used to select the vocabulary originally. For example, if the OOV rate is 25 %, every fourth word of the corpus, in average, is not present in the vocabulary. Table 4.1 lists OOV rates that have been reported in the literature for different languages. We can see that even a 20 000-word vocabulary in English gives a lower OOV rate than a 500 000-word vocabulary in Finnish or Estonian.

The OOV problem seen in Finnish and Estonian, also applies to other morphologically rich languages, where long words can be formed by inflecting, compounding and deriving. Each word may consist of several morphemes that individually bear a meaning[1]. Many expressions that consist of several words in English can be expressed in a single word, as shown in Table 4.2. This results in longer words and a larger vocabulary.

Figure 4.1 illustrates the issue by showing how the size of the vocabulary of a text corpus grows in four languages when the size of the corpus is grown. For example, the English 40-million-word corpus contains about 190 000 words, while the corresponding Finnish corpus contains about 1.9 million unique words.

[1]In linguistics, morphemes are defined as the smallest linguistic units that have a semantic meaning. For example, the English word *unsuccessful* consists of three morphemes: the prefix *un-*, the stem *success*, and the suffix *-ful*.

Language	Vocabulary	OOV rate [%]	Reference
English	20 000	2.4–2.7	(Woodland et al., 1995)
	65 464	0.3–0.7	(Woodland et al., 1995)
Finnish	69 000	13.1–19.9	(Hirsimäki et al., 2006)
	410 000	5.0–7.3	(Hirsimäki et al., 2006)
	500 000	5.4	(Hirsimäki et al., 2009)
Estonian	20 000	28	(Alumäe, 2006, page 75)
	60 000	18	(Alumäe, 2006, page 75)
	500 000	5.6	(Hirsimäki et al., 2009)

Table 4.1: Out-of-vocabulary rates for different languages.

English	Finnish	Estonian	Turkish
to run	juosta	jooksma	koşmak
I run	(minä) juoksen	(mina) jooksen	koşarım
you run	(sinä) juokset	(sina) jooksed	koşarsın
he runs	(hän) juoksee	(tema) jookseb	koşar
by running	juoksemalla	joostes	koşarak
without running	juoksematta	jooksmata	koşmadan
in order to run (I)	juostakseni	jooksmiseks	koşmam için
in order to run (you)	juostaksesi	jooksmiseks	koşman için

Table 4.2: Examples of inflection in Finnish, Estonian and Turkish.

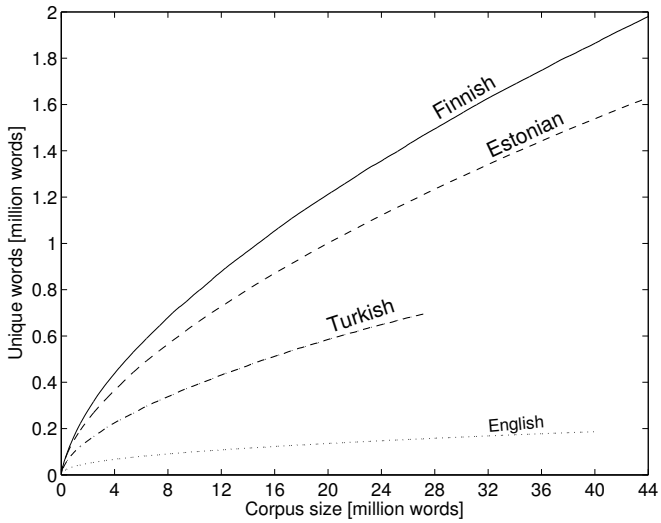

Figure 4.1: Vocabulary growth curves for four languages.

In Estonian and Turkish, the vocabularies also grow considerably faster than in English.

4.2 Segmenting words into morphs

If the words of the language are essentially formed by joining morphemes, it is inefficient to try modelling every word as a whole. Instead, given a text corpus for training an n-gram model, it makes sense to segment the words in the corpus into smaller units and allow the n-gram model to learn the word-internal structure too. In this work, the word segments are referred to as *morphs* regardless of the segmentation approach. Strictly speaking, in linguistics, a morph is defined as a phonetic realisation of a morpheme. However, since the usual aim of the automatic segmentation methods used in speech recognition is to find the most common realisations of the morphemes, and not the underlying morphemes, the term is practical here.

The most important advantage of the morph-based approach is that the vocabulary of the model does not need to be explicitly limited to the most common words of the language. If the words in the training corpus are segmented enough, it is possible to include all the morphs in the model, even if the corpus contained millions of different word forms. Then, when a smoothed n-gram model is trained on the corpus, even the most infrequent words appearing in the training data are modelled to some extent. How accurately they are modelled depends on how long n-grams the model applies in different contexts. By joining the morphs, the smoothed n-gram model is also able to give positive probability to words that did not appear in the training data at all. Thus, the OOV problem is, in theory, avoided altogether.

How should the words then be segmented into morphs? At one extreme is the word-based model where all words are unsegmented and the number of morphs (i.e., the words in this case) is very large resulting in the OOV problem. The other extreme is to segment the words into individual letters leading to a very small morph set, and by joining the morphs, all the words in the language can be formed. However, the morphs will be very short, and as we will see later in this chapter, segmenting words into too short morphs may degrade the recognition accuracy. The optimum lies somewhere between these two extremes.

One might also think that it would be important that the subword units used in speech recognition should correspond to linguistic morphemes of the language as closely as possible. This might be useful if the semantics was somehow taken into account in the model. However, as long as plain n-gram models are used, it does not matter as such whether an individual morph has a real meaning in the language or not. It is more important that the morphs cover the language well and allow efficient n-gram modelling.

4.2.1 Morfessor

In this work, the words are segmented using the Morfessor Baseline algorithm (Creutz and Lagus, 2005). Given a text corpus, the algorithm tries to find a compact set of morphs that enable forming the words of the corpus compactly. Formally, the algorithm maximises the following posterior probability:

$$\arg\max_{\mathcal{M}} P(\mathcal{M}|corpus) = \arg\max_{\mathcal{M}} P(corpus|\mathcal{M}) \cdot P(\mathcal{M}) \tag{4.1}$$

where $\mathcal{M}$ is the segmentation model. The model can also be formulated in a minimum description length framework, as was done in Publication II, but here we follow the formulation given by Creutz and Lagus (2005). Every word in the corpus is modelled as a sequence of morphs, and a simple 1-gram model is used for the morphs:

$$P(corpus|\mathcal{M}) = \prod_{j=1}^{N_w} \prod_{k=1}^{n_j} P(m_{jk}) \tag{4.2}$$

where N_w is the number of the words in the corpus and n_j is the number of morphs in the j^{th} word. The k^{th} morph in the j^{th} word is m_{jk} and its probability $P(m_{jk})$ is computed from the segmented corpus.

The segmentation model $\mathcal{M}$ is defined as the string representations of the morphs $\{s_{m_1}, \ldots, s_{m_M}\}$, and their frequencies $\{f_{m_1}, \ldots, f_{m_M}\}$. When defining the prior probability $P(\mathcal{M})$, the Morfessor Baseline model assumes that the frequency values and the strings are independent of each other, and that the strings are independent of each other:

$$P(\mathcal{M}) = M! \cdot P(s_{m_1}, \ldots, s_{m_M}) \cdot P(f_{m_1}, \ldots, f_{m_M}) \tag{4.3}$$

$$= M! \cdot P(s_{m_1}) \cdots P(s_{m_M}) \cdot P(f_{m_1}, \ldots, f_{m_M}) \tag{4.4}$$

where M is the number of distinct morphs in model. The $M!$ factor is explained by the fact that the order of the morphs in the model does not matter. The probability of the morph string, $P(s_{m_i})$, is defined simply as the product of the individual character probabilities computed from the training data. For the morph frequencies, the following uniform prior distribution is assumed:

$$P(f_{m_1}, \ldots, f_{m_M}) = \binom{N-1}{M-1}^{-1} = \frac{(M-1)!(N-M)!}{(N-1)!} \tag{4.5}$$

where N is the number of morph tokens in the segmented corpus. The detailed derivation is given by Creutz and Lagus (2005, page 25). They also give a few other alternative definitions for the prior probabilities.

The model is optimised using a greedy search algorithm. Initially, all words are unsegmented. Then the words are processed in a random order. For each word, the algorithm evaluates all possible segmentations into two parts. If none of the segmentations improves the unsegmented solution, the word is left unsegmented and the algorithm moves to the next word. Otherwise, the best segmentation is selected and the resulting two morphs are recursively segmented into shorter morphs until no more gains are obtained. The algorithm maintains a binary segmentation tree for each word so that whenever a morph is recursively split, the segmentation is changed in all words containing that morph. The pseudo code of the search algorithm is given in Publication II (page 525).

Morphs obtained with the Morfessor algorithm have been evaluated in several speech recognition experiments. In Finnish recognition tasks the Morfessor morph models have been compared to syllable and word models (Publication I), models based on automatic morphological analysis (Publication II), and word models with huge vocabularies (Publication VI). In all cases, lowest error rates have been obtained with the morph models. Good results have also been reported for Estonian (Kurimo et al., 2006b; Puurula and Kurimo, 2007), Hungarian (Mihajlik et al., 2007), and Turkish (Hacioglu et al., 2003; Kurimo et al., 2006b; Arısoy et al., 2007).

4.2.2 Other segmentation methods

There are a few other data-driven word segmentation algorithms that have been evaluated in speech recognition experiments. Whittaker and Woodland (2000) described an algorithm for finding a word segmentation that maximises the likelihood of the training corpus given by a 2-gram model. In contrast to Morfessor, the complexity of the model was restricted by limiting the number of the morphs. Interpolating the morph model with a word model decreased the error rates slightly in an English broadcast news task. Kneissler and Klakow (2001) used a similar approach but using a 1-gram model for optimising the segmentation. Four different segmentation strategies were evaluated in Finnish and German tasks, but comparison to word-based approaches was not done.

For many languages, there exist automatic morphological analysers that can be used to segment word into morphs. This approach has been used in many speech recognition studies on different languages: Arabic (Sarikaya et al., 2007), Czech (Byrne et al., 2000, 2001), Estonian (Alumäe, 2006), Finnish (Hirsimäki et al., 2006), Hungarian (Mihajlik et al., 2007; Szarvas and Furui, 2003), and Turkish (Hacioglu et al., 2003; Arısoy et al., 2006, 2007). While the analysers may produce morphs that correspond more closely to the linguistic morphemes, their vocabulary is often limited, so all words can not be analysed properly. Also, they usually can not handle foreign names, which are frequent in news material. The data-driven methods, on the other hand, are mostly language independent and are capable for segmenting foreign words, too.

4.3 Interaction between morphs and n-gram models

Because segmenting words into morphs makes the modelling units shorter, the context length of the n-gram model is affected directly. Consider, for example, training a word-based 3-gram model and a morph-based 3-gram model using the same text corpus. If the length of the n-gram contexts are measured in letters, the word model is allowed to use longer contexts than the morph model. Also, the number of n-grams in the word model is most likely greater than in the morph model. Thus, comparing the models directly is not fair.

A better way to compare models with differing word segmentations is to fix the number of n-grams allowed in the model, and to create the best possible n-gram model for each segmentation taking the size constraint into account. After all, the size of the n-gram model is usually the limiting factor in speech recognition, since the computational cost for using the model in recognition does not increase along with the size. In this respect, the pruning and growing algorithms provide an elegant way to create n-gram models for speech recognition. If full models were used, altering the order of the model would change the model size in huge steps, while with variable-length models, the size of the models can be controlled in fine resolution.

The interaction between the segmentation and the context length of the n-gram model may partly explain why the speech recognition experiments reported in the literature have given varying results when morph-based models have been compared with word-based models. Table 4.3 lists recent work on morph-based speech recognition for morphologically rich languages. The languages, of course, have differences that make the morph-based approach more suitable for some languages than others, but the trend seems to be that improvements have not been

Language	N-gram	Vocabularies		Corpus	Comment	Reference
		Words	Morphs	[M words]		
Arabic (ECA)	4	18k	6k	0.16	Words were better.	Publication V
Arabic (ECA)	3	54k?	49k	0.15	Morphs were slightly better.	Kirchhoff et al., 2006
Arabic (IRA)	3	98k	58k	2.8	Morphs were better; combining with words improved further.	Sarikaya et al., 2007
Arabic (MSA)	3, 7	64k, 800k	64k, 800k	14 (utt.)	64k: morphs were better; 800k: no difference.	Choueiter et al., 2006
Arabic (MSA)	3	64k–300k	64k	400	Morphs were better than 64k words, but equal to 300k words.	Xiang et al., 2006
Czech	2	20k	10k	33	Morphs gave better results.	Byrne et al., 2000
Czech	2, 3	60k	25k	33	No difference.	Byrne et al., 2001
Czech	2	64k–300k	—	360	Large vocabularies improve results.	Nouza et al., 2005
Dutch	3	40k	28k	35	Automatic splitting of compound words better than whole words.	Laureys et al., 2002
Estonian	3	60k	60k	76	Morphs were better.	Alumäe, 2006
Estonian	Var.	60k	37k	55	Morphs were better.	Kurimo et al., 2006b
Estonian	Var.	5k, 60k	5k, 60k	43	Morphs were better.	Puurula and Kurimo, 2007
Finnish	3	64k	64k	30	Morphs were better.	Publication I
Finnish	3–5	69k, 410k	26k, 65k	40	Morphs were better.	Publication II
Finnish	Var.	400k	25k	150	Morphs were better.	Kurimo et al., 2006b
German	2, 3	3k	2k	0.12	No difference.	Geutner, 1995
German	4	20k–500k	—	315	Large vocabularies improve results.	McTait and Adda-Decker, 2003
Hungarian	3	20k	5k–8k	0.2	Morphs were slightly better.	Mihajlik et al., 2007
Hungarian	2, 3	—	25k	39	Morph-based approach only.	Szarvas and Furui, 2003
Slovenian	2, 3	20k, 60k	20k, 60k	105	20k: Morphs were better, 60k: morphs slightly better.	Rotovnik et al., 2007
Slovenian	2–4	20k	8k	59	Morphs were better.	Maučec et al., 2003
Turkish	3	60k	15k–48k	2	Morphs were better, some recombination improved further.	Hacioglu et al., 2003
Turkish	2, 3	30k	2k–23k	81	Half-words were better. Authors mention that the stems from the test set were added to all models.	Erdoğan et al., 2005
Turkish	6	50k	34k	27	Morphs were better.	Kurimo et al., 2006b
Turkish	2	60k	60k	10	Words were better.	Arısoy et al., 2006
Turkish	3-?	50k	34k	96	Morphs were better (highest n-gram order not mentioned).	Arısoy et al., 2007

Table 4.3: Related work on morph-based speech recognition

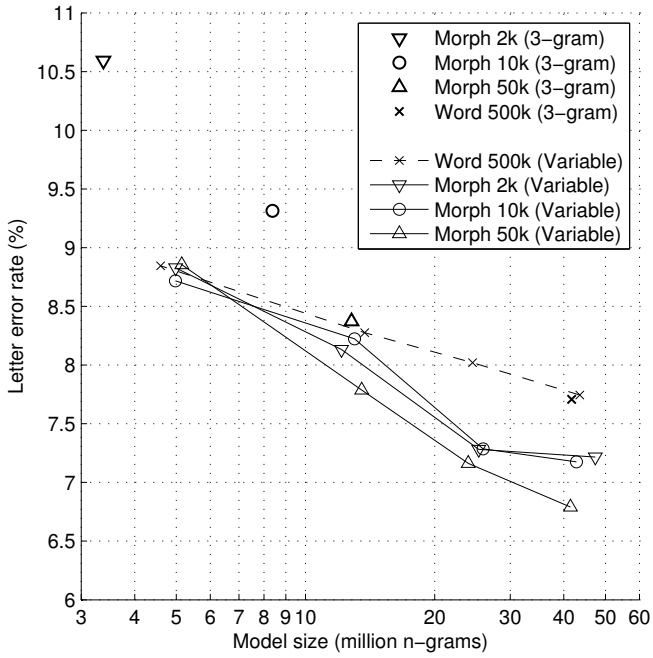

Figure 4.2: Finnish recognition results.

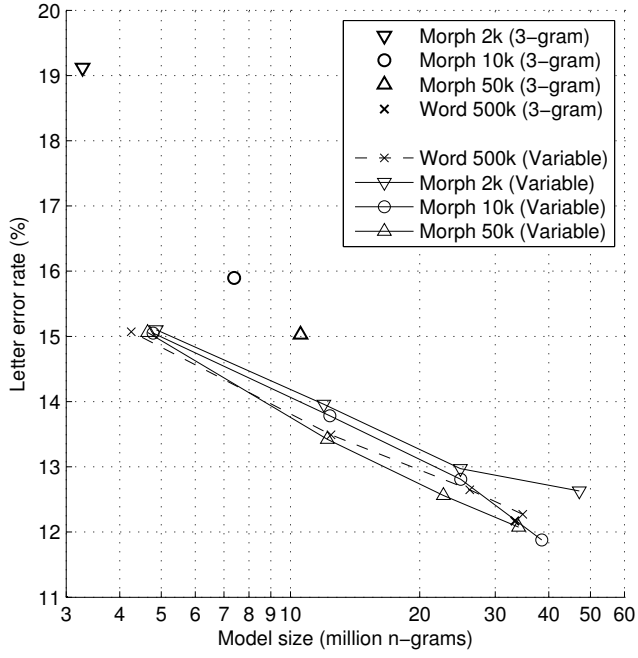

Figure 4.3: Estonian recognition results.

obtained with morph-based models if the order of the n-gram model has been low. Another important factor is the size of the vocabulary in the word model. Small vocabularies lead to high OOV rates in recognition, in which case switching to morph-based models improve the results more easily.

In Publication VI, the interaction was studied further. Figure 4.2 shows the effect in a Finnish speech recognition task. The speech recognition experiments were repeated with three different morph models and a huge word model. If a fixed 3-gram model is used, the recognition results and the size of the models are heavily affected by the length of the morphs. On the other hand, if the model size is fixed and variable-length models are compared, the morph length has much smaller effect on the recognition result. Figure 4.3 shows similar results in an Estonian recognition task.

4.4 Morphs versus words

Since training n-gram models on morphs obtained with the Morfessor algorithm has given good results in speech recognition, it is interesting to examine where the recognition errors are reduced. It is expected that the morph-based recognisers are better at recognising words that are missing from the vocabulary of the word-based systems, but are the results better for the in-vocabulary words too? The issue was first studied in Publication V, where earlier recognition results from Finnish, Estonian, Turkish, and Arabic experiments were gathered. This thesis concentrates on the Finnish and Estonian tasks, but the Turkish and Arabic results are also shown below for completeness. See Chapter 3 in Publication V for details of the recognition settings used in the experiments reported below.

For each experiment, the words in the reference transcription of the test set were divided in three groups:

1. Words present in the vocabulary of the word-based model

2. Words excluded from the vocabulary of the word-based model

3. Words not occurring in the training data at all.

Then, the letter accuracies were computed separately for these groups. Figure 4.4 shows the accuracies of the first group for morph-based systems (shown in black) and word-based systems (shown in white). The grey bars show the overall accuracies computed for the whole test set for reference. As can be seen, there is no consistent difference in the performance between the morph and word-based systems.

Figure 4.5 shows the accuracies for the second group, the OOV words. As expected, the morph-based models give considerably better results in this group, since the word-based models simply can not recognise these words correctly. The only exception is the Arabic task.

Finally, the accuracies for completely new words are shown in Figure 4.6. This group is the most difficult group for all the models. Still, also in this group, the morph-based models give considerably better results when compared to word models.

Since the Finnish systems considered above were all speaker-dependent, a similar analysis was repeated in Publication VI for a more recent speaker-independent

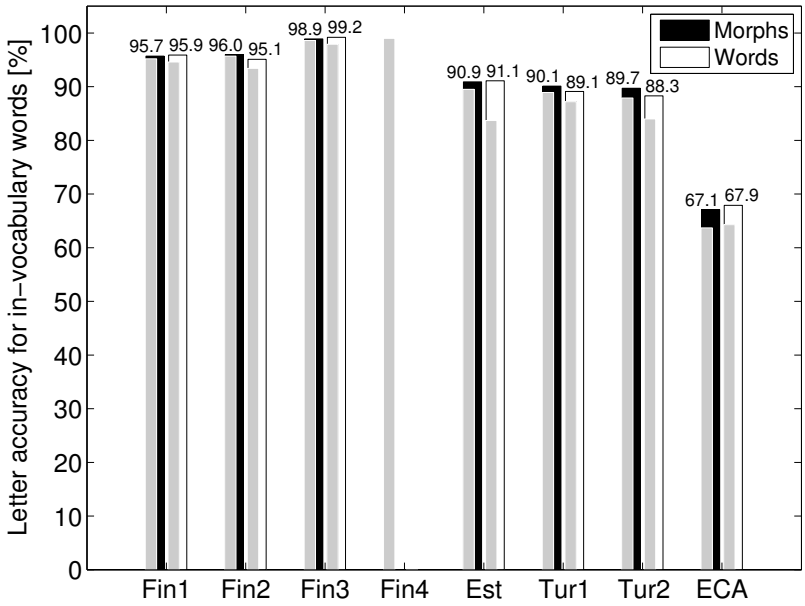

Figure 4.4: Letter accuracies for in-vocabulary words.

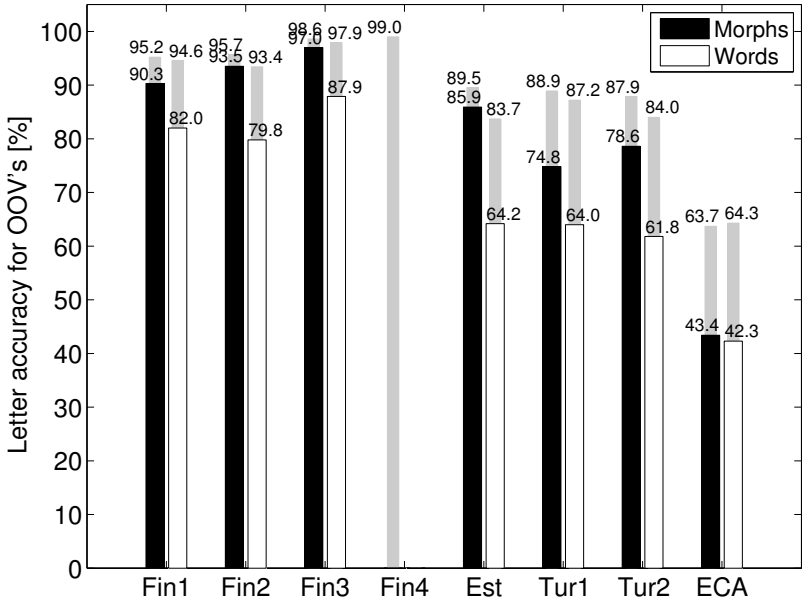

Figure 4.5: Letter accuracies for OOV words.

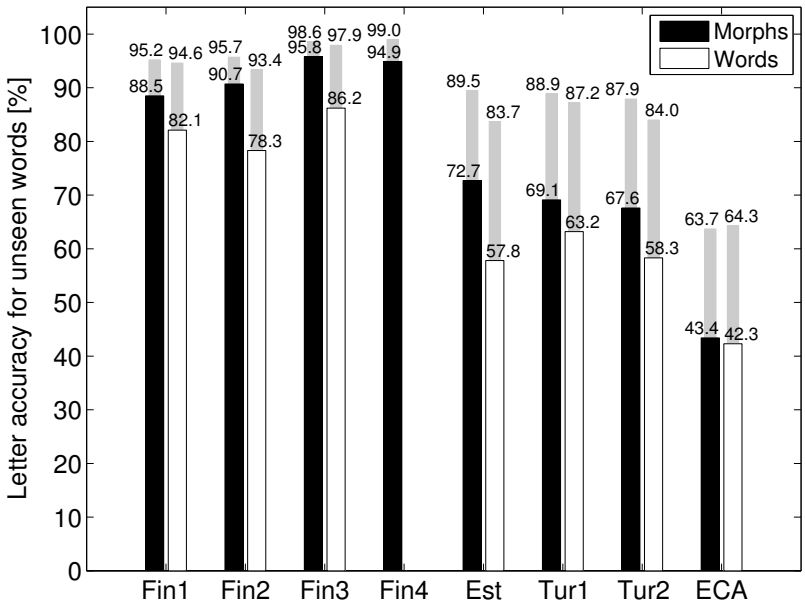

Figure 4.6: Letter accuracies for new words.

system (the FinSD task), and an Estonian task (EstSD). Furthermore, three different Morfessor-based morph segmentations were experimented in order to study how the length of the morphs affect the errors in different word groups, and the vocabulary of the word model was extended up to 500 000 words. Finnish results are shown in Figure 4.7 and Estonian results in Figure 4.8.

These results suggest that morph-based models improve the recognition accuracy for the words that are outside the vocabularies of the word models. The improvement is largest for the words that are present in the training data, but previously unseen words are also recognised more accurately. The results do not seem to be affected greatly by the size of the morph set, even if the average morph length in the smallest set is only 2.5 letters (3.4 letters in the 50k model and 7.9 letters in the word model).

4.5 Word models with very large vocabularies

There are a few studies in which the size of the vocabulary used in a conventional word model is increased to hundreds of thousands of words. McTait and Adda-Decker (2003) report that the word error rate in a German task drops from 20.4 % to 18.5 % when a 60 000-word vocabulary is replaced by a 300 000-word vocabulary. At the same time, the OOV rate drops from 3.5 % to 1.4 %. Similar behaviour is reported also for Czech (Nouza et al., 2005): increasing the vocabulary size from 64 000 words to 310 000 words, decreases the error rate from 31.5 % to 24.5 % and reduces the OOV rate from 5.2 % to 0.8 %. In these studies, however, the performance of the word model is not compared to morph-based approaches.

So far, there have been only a few studies where word models with very large vocabularies are compared to morph-based approaches. In Figure 4.2, we saw that

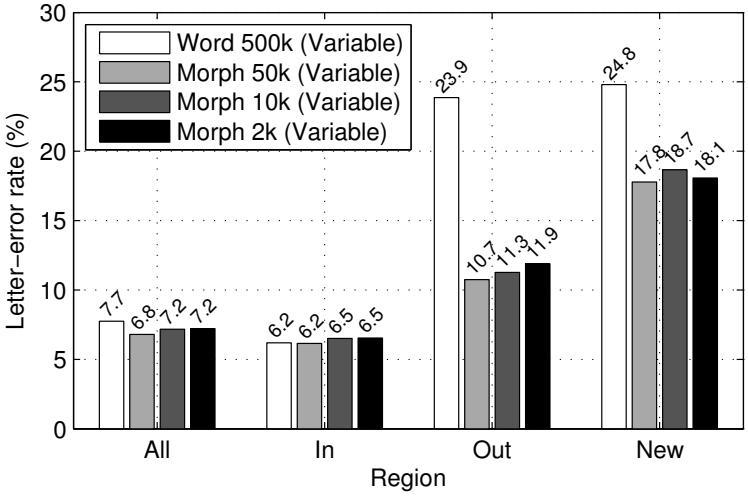

Figure 4.7: Letter error rates in the FinSD task.

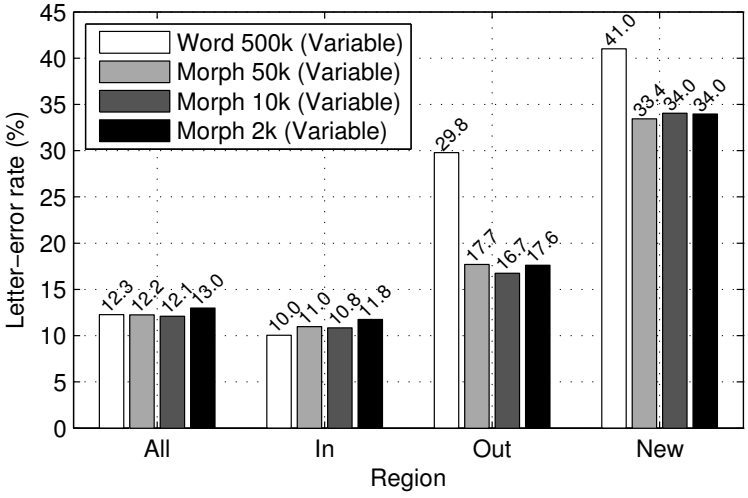

Figure 4.8: Letter error rates in the EstSD task.

even a 500 000-word model does not reach the performance of the morph model in a Finnish task, but in an Estonian task it does. Also, in Arabic experiments (Choueiter et al., 2006; Xiang et al., 2006), very large word vocabularies gave equal results with morph models.

However, using very large vocabularies may be cumbersome in some recognition systems. In the experiments of Publication VI, the run-time memory consumption of the word-based recognition systems was much higher when compared to the morph models. This was mainly due to the search network and LM look-ahead caches. The look-ahead algorithm has to maintain a cache for LM probabilities that have been computed recently, and if the vocabulary of the LM contains half a million words, storing the probability distributions takes a lot of space. It might be possible to optimise the look-ahead algorithm for very large vocabularies, but that has not been pursued in this work.

CHAPTER 5
ANALYSING RECOGNITION ERRORS

When a recognition system makes an error, we would always want to know why the system has selected an erroneous hypothesis instead of the correct transcription. If we knew the cause of the error, we might be able to improve the system and avoid the error in future. State-of-the-art recognition systems are, however, very complex, and there are numerous possible causes for errors: some of the phoneme models or the language model may be poor, training data may have been insufficient, or perhaps the correct transcription is the most probable hypothesis according to the models, but the decoder has failed to find it. It is also possible that there is just too much noise in the signal, or the speaker just stammers or pronounces the words wrongly.

Even if it might be difficult to point out the reason for every single error, it is useful to find out if the recogniser consistently makes errors in certain situations. In this chapter, automatic and manual methods are used to analyse the errors made by a Finnish morph-based recognition system.

5.1 Error Region Analysis

The analysis techniques used in this chapter are based on a method called error region analysis (ERA) introduced by Chase (1997). ERA can be used to automatically analyse the role of the acoustic model and the language model at erroneous regions. The main idea is to segment the recognition output into independent error regions, and find out which of the models prefers the erroneous hypothesis over the reference transcription.

First, the recognition hypothesis and the reference transcription are aligned in time as shown in Figure 5.1. In order to get the time alignment of the reference

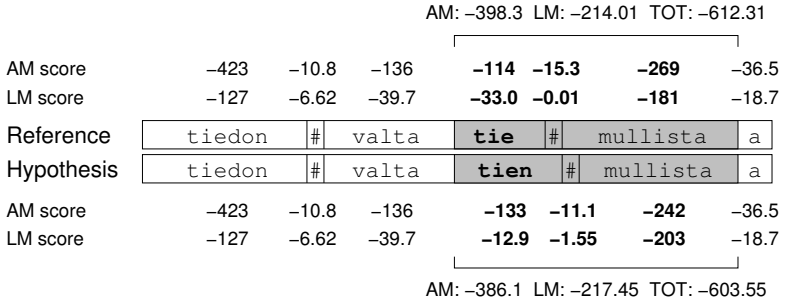

Figure 5.1: An example of a HYP-AM region. The hypothesis has a better total score, the acoustic model prefers the hypothesis, and the language model prefers the reference.

Category	Conditions		
	P_{total}	P_{AM}	P_{LM}
REF-TOT	REF $>$ HYP		
HYP-AM	HYP $>$ REF	HYP $>$ REF	HYP $\leq$ REF
HYP-LM	HYP $>$ REF	HYP $\leq$ REF	HYP $>$ REF
HYP-BOTH	HYP $>$ REF	HYP $>$ REF	HYP $>$ REF

Table 5.1: Categories of the error regions.

transcription, the decoder can be forced to recognise the reference transcription. The hypothesis is then segmented into *correct regions* and *error regions*. Correct regions (white boxes in the figure) only contain morphs that are identical to the corresponding reference morph: identical morph labels, start and end times, model scores, and language model states. Other morphs belong to error regions (dark boxes in the figure). This ensures that the error regions are independent so that errors in one region do not affect errors in the other regions.

Next, the error regions are divided into categories according to Table 5.1. The first category, REF-TOT, contains regions in which the total score of the reference transcription is higher than the score of the hypothesis selected by the recogniser. These errors are so called search errors. The recogniser would prefer the complete reference transcription over the selected hypothesis, but, for some reason, the decoder has pruned the partial hypothesis of the reference transcription at some stage. The number of REF-TOT errors typically increases if the recognition is speeded up by lowering pruning thresholds. In this work, however, we are more interested in other errors, so the pruning thresholds are kept high in order to keep the number of REF-TOT errors low.

The other three error categories mentioned in Table 5.1 correspond to errors where the models (when combined together) actually prefer the erroneous hypothesis over the reference transcription. Thus, in a way, either the AM, the LM, or both of the models are wrong. In HYP-AM errors, the AM prefers the erroneous hypothesis so strongly, that the erroneous hypothesis is selected, even if the LM prefers the reference transcript. In HYP-LM errors, the situation is reversed: the LM prefers the erroneous hypothesis strongly, while the AM prefers the reference transcript. Finally, HYP-BOTH contains errors where both the LM and AM prefer the erroneous hypothesis over the reference transcript.

5.2 Comparison of training methods

It has been shown that the basic acoustic models trained with the conventional maximum likelihood criteria can be improved by using speaker adaptive training or discriminative training. It has also been shown that combining these techniques improve the results further (McDonough et al., 2002). In this work, the ERA method is used to study what kind of effect the different training procedures have on the recognition error types.

Publication VII presented recognition experiments on the Finnish SpeechDat database using three kinds of models. The first model was trained using the traditional maximum likelihood (ML) training without any speaker adaptation. The next model (ML+SAT) added three iterations of speaker adaptive training

Region	Letter errors		
	ML	ML+SAT	ML+SAT+MPFE
HYP-BOTH	962	909	783
HYP-AM	1059	709	727
HYP-LM	623	597	425
REF-TOT	82	60	15
Total	2726	2275	1950
LER (%)	6.8	5.6	4.8

Table 5.2: Letter errors for different training methods and error regions.

and unsupervised speaker adaptation was used during recognition. In the third model (ML+SAT+MPFE), the ML+SAT model was trained further using four iterations of discriminative training. The evaluation set was recognised using these three models, the hypotheses were segmented into error regions using ERA, and the letter errors were computed separately for the error region categories. The results are shown in Table 5.2.

First, looking at the letter error rates, we can see that speaker adaptive training and the discriminative training improve the recognition accuracy as expected. For the ML model, most of the letter errors occur in the HYP-AM regions, almost twice as many as in the HYP-LM regions. Using speaker adaptation in training and recognition reduces the letter errors considerably in the HYP-AM regions, but less in other regions. This is perhaps expected. Since the evaluation set only contains new speakers, which were not included in the training set, the mismatch between the training and evaluation data likely causes errors for some speakers. Adapting the models towards the evaluation set is expected to increase the acoustic score of the reference transcripts, which is likely to correct errors especially in the HYP-AM regions, in which the language model already prefers the reference transcript.

Discriminative training, on the other hand, seems to decrease the number of letter errors in HYP-BOTH and HYP-LM regions, and actually the number of errors in HYP-AM region increases. The reason for this is not clear.

5.3 Manual error classification

In order to analyse the errors of the best model (ML+SAT+MPFE) further, the error regions were manually inspected. In order to minimise the manual work, a simple program was used to show the erroneous sentences, the error regions with recognition hypothesis and the reference aligned in time, and to play the audio. Then the letter errors in each error region were assigned manually to different error categories. Each error region could contain many error categories, but each letter error was assigned to one category only.

Table 5.3 shows, for each cause, the total number of letter errors, and the number of letter errors in HYP-BOTH, HYP-AM, HYP-LM and REF-TOT regions.

Foreign: This is the largest specific error class. It contains recognition errors on foreign names or words, which usually do not follow the normal Finnish pronunciation rules. Since the current recogniser does not have a special pronunciation modelling for foreign words, it is natural that almost all errors are in HYP-BOTH and HYP-AM regions, where the acoustic model prefers the erroneous hypothesis

Cause	Letter errors				
	Total	HYP-BOTH	HYP-AM	HYP-LM	REF-TOT
Foreign	156	89	**61**	6	
Inflect	143	74	26	**43**	
Pronounce	131	37	**84**		10
Noise	124	21	**97**	6	
Name	81	29	**29**	23	
Delete	65	29	9	**27**	
Acronym	53	44	**6**	3	
Compound	42	11	8	**23**	
Correct	37	15	**19**	3	
Rare	27	11	3	**13**	
Insert	9	3	**6**		
Other	1082	421	**379**	277	5

Table 5.3: Manual error classes and the number of letter errors for the ML+SAT+MPFE system. Dominance of AM or LM shown in bold.

over the reference transcription. In order to alleviate the problem, new pronunciation rules for the most common foreign letter sequences would be necessary. However, since the words are currently segmented into morphs before any pronunciation is considered, the rules may not be applicable after the morphs are taken apart. Therefore, one may also need a way to prevent segmentation in the middle of a letter sequence that has a special pronunciation rule.

Inflect: In these regions, the root of the word has been recognised correctly, but the inflected form is slightly wrong. Quite often, there is only a one-letter difference between the hypothesis and the reference, which corresponds to one-phoneme difference in Finnish and makes the difference between the AM scores small too. Thus, the LM usually gets to decide which inflected form is used. However, correcting these errors by modifying the language model may be difficult, since getting the inflected forms right often requires quite deep understanding of the language and long contexts have to be considered.

Pronounce: Sometimes the recogniser makes errors because the speaker pronounces a word very carelessly, stammers or hesitates in the middle of the word, or even utters a non-word. As expected, the acoustic model never prefers the reference transcription over the hypothesis in these cases. Also, in this case, it is very difficult to figure out ways to improve the accuracy. In some cases, it is not even clear what the correct transcription should be.

Noise: In the regions that have some noise on the background, the majority of the errors are also in HYP-AM regions. Dealing with noise is difficult, but developing noise-robust recognition systems is an active research topic.

Name: The next region contains misrecognised Finnish proper names. In contrast to foreign proper names, Finnish names follow the usual pronunciation rules, so the pronunciation modelling should not be a problem. The errors are evenly distributed between HYP-BOTH, HYP-AM, and HYP-LM regions. There is no obvious way to correct these errors.

Delete and **Insert**: A short word is deleted or inserted incorrectly. It seems that the system more often deletes than inserts short words incorrectly. Some of the errors might be avoided by introducing an artificial language model cost or

score for inserting words or morphs in the hypothesis. This kind of parameter is often used in English recognition systems and called *insertion penalty*.

Acronym: As in the case of foreign words, the recogniser does not have a separate pronunciation dictionary for acronyms, so the acoustic model seldom prefers the reference transcript. In order to correct these errors, the acronyms should be first detected from the language model training data. The main difficulty is that the text material is originally from various sources and the writing conventions may be different. For example, often upper case letters are used (*TV*, *SAK*) for acronyms, but sometimes not. Other words may also be written in upper case. An acronym may be written in a kind of phonetic form (*teevee* instead of *TV*). The inflected forms may be written with or without a colon (*SAK:n* or *SAKin*). Then the most probable pronunciation variants should be defined for each acronym, and the word segmentation algorithm should be altered to avoid splitting in the middle of acronyms.

Compound: In these regions, the parts of the compound word are recognised correctly, but a word boundary is inserted or deleted incorrectly. In fluent speech, the word boundaries often have little acoustic evidence unless the speaker keeps an explicit pause between the words. Thus, the word boundaries in compound words are practically decided by the language model, which can also be seen in Table 5.3. Here, the quality of the LM training data plays a significant role, since sometimes the compound words are already written incorrectly in the training corpus.

Correct: In some cases, the recognition hypothesis differs from the reference transcription, but can still be considered correct. For example, there may be two ways to write a word correctly. Sometimes, the recognition hypothesis may be acoustically so close to the reference that it may be hard for even humans to distinguish which one is actually said in the audio.

Rare: The misrecognised word is clearly very rare. The errors based on the frequency of the word are analysed in greater detail in the next section.

Other: All other errors that did not have an obvious reason. Often the quality of the audio and the language seemed to be normal, but still the sentence had been recognised incorrectly. Since the majority of the errors are in HYP-AM regions, one could suggest that most of the errors are caused by difficult acoustic conditions. It would interesting to see, how the number of errors would change if more acoustic training data was available.

Since the manual classification of the errors is always subjective, it is not possible to make very strong conclusions from the analysis, especially for the categories that are infrequent. Still, the analysis suggests that there are some common error types that could be attacked by improving the pronunciation model of the system.

5.4 Frequency analysis

In word-based recognition, some of the words in the training data are considered out-of-vocabulary words. The OOV words are mapped to a unique OOV symbol, so the n-gram language model does not even try to model the OOV words. In morph-based language modelling, the aim is to model all words in the training data. If enough splitting is done, all morphs of the training data are included in the language model. Then it is left to the variable-length n-gram model to decide how long contexts are used in different situations. Generally, frequent words and

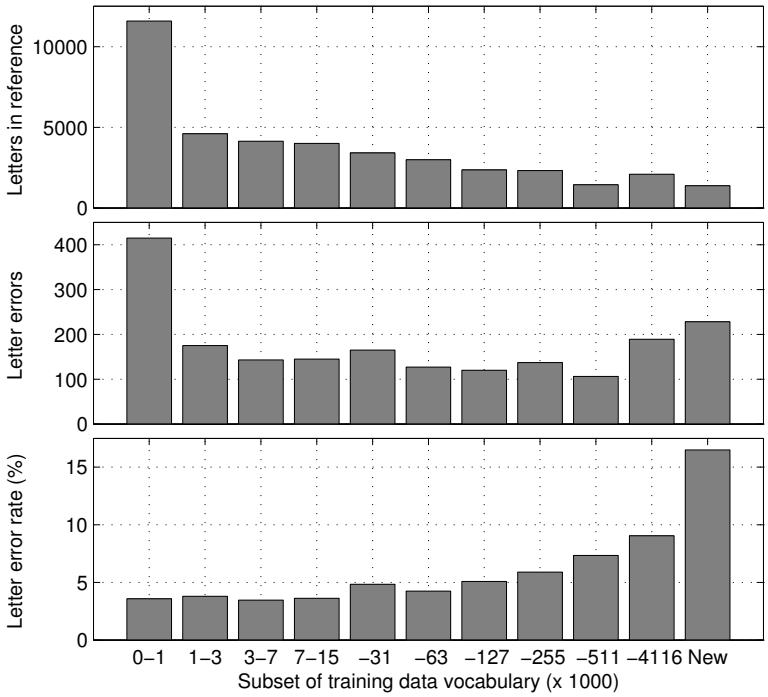

Figure 5.2: Frequency analysis.

phrases are modelled with long contexts, and rare words and phrases with short contexts. Since frequent words usually get higher LM scores from the language model, the natural hypothesis is that the recognition accuracy is better for frequent words that for rare words.

In Publication VII, the effect of the word frequencies on the recognition accuracy was studied. The vocabulary of the language model training corpus was partitioned as follows. The first partition contained the 1000 most common words, the next partition contained the next 2000 words, then next 4000 words, and so on, until the last partition contained the words that were not among the most common 512 000 words. Then the reference transcriptions used in the speech recognition experiments were segmented according to these partitions and letter error rates were computed separately for each partition. Additionally, there was a partition for words that did not occur in the LM training corpus, but were present in the reference transcription.

Figure 5.2 shows the error statistics for the ML+SAT+MPFE model. The number of letters in each partition is shown at top, the absolute number of letter errors at middle, and the letter error rates at bottom. First, looking at the error rates, we see that the error rate increases steadily towards the rare words, and is highest for the words that did not appear in the LM training data at all. On the other hand, the absolute number of letter errors shows that a large portion of the letter errors occur in the words that are among the 1000 most common words. These statistics suggest that even if the recognition of previously unseen words is considerably more difficult than the words in the training data, they are not a major problem in this task, because they are so rare.

CHAPTER 6
CONCLUSIONS

Even if the languages spoken around the world differ from each other phonetically and grammatically, the speech recognition methods developed during recent decades have been surprisingly language independent. Especially, the feature extraction methods and the HMMs as phoneme models have shown to work well across languages. Even if the n-gram language models, as such, are also language independent, the methodology used for modelling the words needs to take the target language into account. In morphologically rich languages where words are formed through inflection, derivation, and compounding, the growth of vocabulary makes it difficult to cover the language adequately with a fixed-size vocabulary.

This thesis covers methods that can be used to build efficient speech recognition systems for morphologically rich languages. The main approach is to train the statistical n-gram models on a text corpus, the words of which have automatically been segmented into shorter fragments called morphs. Basing the n-gram model on morphs instead of whole words allows the model, in theory, to capture unlimited vocabularies using a reasonably small-sized morph set. This enables training the model on all the words of the training text corpus without limiting to the most frequent words only. Furthermore, the model is able to capture completely new words that are not present in the training data but can be formed by joining available morphs together. By comparing the recognition errors made by the word-based and morph-based systems, it is found out that the morph-based approach reduces recognition errors especially on the words that are infrequent or not present in the language model training corpus.

It is also shown how variable-length n-gram models, which are trained using growing and pruning algorithms, can be used to improve the accuracy of morph-based recognition systems. The use of high-order variable-length models seems to be especially useful when the language model contains many short morphs. Restricting the order of the n-gram model degrades the recognition accuracy considerably if short morphs are used. On the other hand, when variable-length models are used, the length of the morphs used in the language model does not affect the recognition accuracy so much. This may partly explain some of the previous results reported in the literature where morph-based language models, used with 2-gram or 3-gram models, have not improved word-based models.

In order to use variable-length n-gram models in a one-pass speech recognition system, the decoding algorithm should be able to take long n-gram contexts into account. The recognition system described in this thesis represents the allowed morph sequences as a finite-state search network, and the recognition hypotheses as tokens that are propagated in the network. By representing the language model as a separate finite-state automaton, and using the states of the automaton to mark the n-gram context in the tokens, recognition hypotheses can be pruned efficiently when it is known that another token in the same n-gram context will

be more probable in the end. Alternatively, the presented compact n-gram data structure can be used if memory-efficient recognition is desired.

Analysing the errors made by the Finnish recognition system sheds light on how the recognition errors are affected by the training criteria used in acoustic modelling. Using speaker adaptation in training and recognition reduces letter errors mostly in the regions where the acoustic model erroneously preferred hypothesis even if the language model would have preferred the reference transcription. Discriminative training, on the other hand, seems to reduce errors mostly in the regions where the language model, alone or together with the acoustic model, prefers the erroneous hypothesis.

All in all, even with the best acoustic model, there are more errors in the regions where the acoustic model dominated when compared to the regions dominated by the language model. A manual classification of the error regions suggests that a part of these errors might be avoided, if the pronunciation models for acronyms and foreign proper names were improved. In future, it would also be interesting to analyse how much increasing the amount of data used for training the acoustic and language models would reduce the errors in different regions.

BIBLIOGRAPHY

Achim Sixtus, H. N. (2002). From within-word model search to across-word model search in large vocabulary continuous speech recognition. *Computer Speech and Language*, 16(2):245–271.

Alumäe, T. (2006). *Methods for Estonian Large Vocabulary Speech Recognition*. PhD thesis, Tallinn University of Technology.

Anastasakos, T., McDonough, J., Schwartz, R., and Makhoul, J. (1996). A compact model for speaker-adaptive training. In *Proceedings of the 4th International Conference on Spoken Language Processing (ICSLP)*, pages 1137–1140.

Arısoy, E., Dutağaci, H., and Arslan, L. M. (2006). A unified language model for large vocabulary continuous speech recognition of Turkish. *Signal Processing*, 86(10):2844–2862.

Arısoy, E., Sak, H., and Saraçlar, M. (2007). Language modeling for automatic Turkish broadcast news transcription. In *Proceedings of the Annual Conference of the International Speech Communication Association (Interspeech)*, pages 2381–2384.

Byrne, W., Hajič, J., Ircing, P., Jelinek, F., Khudanpur, S., Krbec, P., and Psutka, J. (2001). On large vocabulary continuous speech recognition of highly inflectional language — Czech. In *Proceedings of the 7th European Conference on Speech Communication and Technology (Eurospeech)*, pages 487–489.

Byrne, W. J., Hajič, J., Krbec, P., Ircing, P., and Psutka, J. (2000). Morpheme based language models for speech recognition of Czech. In *Proceedings of the Third International Workshop on Text, Speech, Dialogue (TSD)*, pages 211–216.

Chase, L. L. (1997). *Error-Responsive Feedback Mechanisms for Speech Recognizers*. PhD thesis, Carnegie Mellon.

Chen, S., Kingsbury, B., Mangu, L., Povey, D., Saon, G., Soltau, H., and Zweig, G. (2006). Advances in speech transcription at IBM under the DARPA EARS program. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(5):1596–1608.

Chen, S. F. and Goodman, J. (1999). An empirical study of smoothing techniques for language modeling. *Computer Speech and Language*, 13(4):359–393.

Choueiter, G., Povey, D., Chen, S. F., and Zweig, G. (2006). Morpheme-based language modeling for Arabic LVCSR. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages I–1053–I–1056.

Creutz, M. and Lagus, K. (2005). Unsupervised morpheme segmentation and morphology induction from text corpora using Morfessor 1.0. Publications in Computer and Information Science A81, Helsinki University of Technology.

Digalakis, V. V., Rtischev, D., and Neumeyer, L. G. (1995). Speaker adaptation using constrained estimation of Gaussian mixtures. *TSAP*, 3(5):357–366.

Erdoğan, H., Büyük, O., and Oflazer, K. (2005). Incorporating language constraints in sub-word based speech recognition. In *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 98–103.

Gales, M. J. F. (1998). Maximum likelihood linear transformations for HMM-based speech recognition. *Computer Speech and Language*, 12(2):75–98.

Gales, M. J. F. (1999). Semi-tied covariance matrices for hidden Markov models. *TSAP*, 7(3):272–281.

Geutner, P. (1995). Using morphology towards better large-vocabulary speech recognition systems. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 445–448.

Good, I. J. (1953). The population frequencies of species and the estimation of population parameters. *Biometrika*, 40:237–264.

Hacioglu, K., Pellom, B., Ciloglu, T., Ozturk, O., Kurimo, M., and Creutz, M. (2003). On lexicon creation for Turkish LVCSR. In *Proceedings of the 8th European Conference on Speech Communication and Technology (Eurospeech)*, pages 1165–1168.

Hansen, J., Huang, R., Zhou, B., Seadle, M., Deller, J., Gurijala, A., Kurimo, M., and Angkititrakul, P. (2005). SpeechFind: Advances in spoken document retrieval for a National Gallery of the Spoken Word. *IEEE Transactions on Speech and Audio Processing*, 13(5):712–730.

Hirsimäki, T., Creutz, M., Siivola, V., Kurimo, M., Virpioja, S., and Pylkkönen, J. (2006). Unlimited vocabulary speech recognition with morph language models applied to Finnish. *Computer Speech and Language*, 20(4):515–541.

Hirsimäki, T., Pylkkönen, J., and Kurimo, M. (2009). Importance of high-order n-gram models in morph-based speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing*. Accepted for publication.

Juang, B. and Rabiner, L. R. (2005). Automatic speech recognition—a brief history of the technology. In *Elsevier Encyclopedia of Language and Linguistics*. 2nd edition.

Kanthak, S., Sixtus, A., Molau, S., and Ney, H. (2000). Within-word vs. across-word decoding for online speech recognition. In *Proceedings of Automatic Speech Recognition Workshop*, pages 40–45.

Katz, S. M. (1987). Estimation of probabilities from sparse data for the language model component of a speech regonizer. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 35(3):400–401.

Kirchhoff, K., Vergyri, D., Bilmes, J., Duh, K., and Stolcke, A. (2006). Morphology-based language modeling for conversational Arabic speech recognition. *Computer Speech and Language*, 20(4):589–608.

Kneissler, J. and Klakow, D. (2001). Speech recognition for huge vocabularies by using optimized sub-word units. In *Proceedings of the 7th European Conference on Speech Communication and Technology (Eurospeech)*, pages 69–72.

Kneser, R. (1996). Statistical language modeling using a variable context length. In *Proceedings of the 4th International Conference on Spoken Language Processing (ICSLP)*, pages 494–497.

Kurimo, M., Puurula, A., Arısoy, E., Siivola, V., Hirsimäki, T., Pylkkönen, J., Alumäe, T., and Saraçlar, M. (2006a). Unlimited vocabulary speech recognition for agglutinative languages. In *Human Language Technology, Conference of the North American Chapter of the Association for Computational Linguistics, HLT-NAACL*, pages 487–494.

Kurimo, M., Puurula, A., Arısoy, E., Siivola, V., Hirsimäki, T., Pylkkönen, J., Alumäe, T., and Saraçlar, M. (2006b). Unlimited vocabulary speech recognition for agglutinative languages. In *Human Language Technology, Conference of the North American Chapter of the Association for Computational Linguistics, HLT-NAACL*, pages 487–494.

Laureys, T., Vandeghinste, V., and Duchateau, J. (2002). A hybrid approach to compounds in LVCSR. In *Proceedings of the 4th International Conference on Spoken Language Processing (ICSLP)*, pages 697–700.

Maučec, M. S., Rotovnik, T., and Zemljak, M. (2003). Modelling highly inflected Slovenian language. *International Journal of Speech Technology*, 6(3):245–257.

McDonough, J., Schaaf, T., and Waibel, A. (2002). On maximum mutual information speaker-adapted training. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 1, pages I–601–I–604.

McTait, K. and Adda-Decker, M. (2003). The 300k LIMSI German broadcast news transcription system. In *Proceedings of the 8th European Conference on Speech Communication and Technology (Eurospeech)*, pages 213–216.

Meister, E., Lasn, J., and Meister, L. (2003). Development of the estonian speechdat-like database. In *Proceedings of the 8th European Conference on Speech Communication and Technology (Eurospeech)*, pages 1601–1604.

Meister, E., Lasn, J., and Meister, L. (2004). Estonian SpeechDat completed. In *Foneetikan päivät*, pages 61–64.

Mihajlik, P., Fegyó, T., Tüske, Zoltá, and Ircing, P. (2007). A morpho-graphemic approach for the recognition of spontaneous speech in agglutinative languages - like Hungarian. In *Proceedings of the Annual Conference of the International Speech Communication Association (Interspeech)*, pages 1497–1500.

Mohri, M., Pereira, F., and Riley, M. (2002). Weighted finite-state transducers in speech recognition. *Computer Speech and Language*, 16(1):69–88.

Nádas, A. (1985). On Turing's formula for word probabilities. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 33(6):1414–1416.

Ney, H. and Essen, U. (1991). On smoothing techniques for bigram-based natural language modelling. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 825–828.

Ney, H., Essen, U., and Kneser, R. (1994). On structuring probabilistic dependences in stochastic language modelling. *Computer Speech and Language*, 8(1):1–38.

Nouza, J., Ždánský, J., David, P., Červa, P., Kolorenč, J., and Nejedlová, D. (2005). Fully automated system for Czech spoken broadcast transcription with very large (300k+) lexicon. In *Proceedings of the 9th European Conference on Speech Communication and Technology (Interspeech)*, pages 1681–1684.

Odell, J. J. (1995). *The Use of Context in Large Vocabulary Speech Recognition*. PhD thesis, University of Cambridge.

Ortmanns, S., Ney, H., and Eiden, A. (1996). Language-model look-ahead for large vocabulary speech recognition. In *Proceedings of the 4th International Conference on Spoken Language Processing (ICSLP)*, pages 2095–2098.

Picone, J. (1993). Signal modeling techniques in speech recognition. *Proceedings of the IEEE*, 81(9):1215–1247.

Povey, D. (2003). *Discriminative Training for Large Vocabulary Speech Recognition*. PhD thesis, University of Cambridge.

Povey, D. and Kingsbury, B. (2007). Evaluation of proposed modifications to MPE for large scale discriminative training. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages IV–321–IV–324.

Puurula, A. and Kurimo, M. (2007). Vocabulary decomposition for Estonian open vocabulary speech recognition. In *45th Annual Meeting of the Association for Computational Linguistics*, pages 89–95.

Pylkkönen, J. (2005). An efficient one-pass decoder for Finnish large vocabulary continuous speech recognition. In *Proceedings of 2nd Baltic conference on human language technologies*, pages 167–172.

Pylkkönen, J. (2004). Using phone durations in finnish large vocabulary continuous speech recognition. In *Proceedings of the 6th Nordic Signal Processing Symposium (Norsig*, pages 324–327.

Raj, B. and Whittaker, E. (2003). Lossless compression of language model structure and word identifiers. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 388–391.

Renals, S., Abberley, D., Kirby, D., and Robinson, T. (2000). Indexing and retrieval of broadcast news. *Speech Communication*, 32(1–2):5–20.

Renals, S., Hain, T., and Bourlard, H. (2007). Recognition and understanding of meetings the AMI and AMIDA projects. In *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 238–247.

Riccardi, G., Pieraccini, R., and Bocchieri, E. (1996). Stochastic automata for language modeling. *Computer Speech and Language*, 10(4):265–293.

Ristad, E. and Thomas, R. (1995). New techniques for context modeling. In *Preceedings of the Meeting of the Association for Computational Linguistics*, pages 220–227.

Rosenberg, A. E., Lee, C.-H., and Soong, F. K. (1994). Cepstral channel normalization techniques for HMM-based speaker verification. In *Proceedings of the 4th International Conference on Spoken Language Processing (ICSLP)*, pages 1835–1838.

Rotovnik, T., Maučec, M. S., and Kačič, Z. (2007). Large vocabulary continuous speech recognition of an inflected language using stems and endings. *Speech Communication*, 49(6):437–452.

Sarikaya, R., Afify, M., and Gao, Y. (2007). Joint morphological-lexical language modeling (JMLLM) for Arabic. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages IV–181–IV–184.

Siivola, V. (2007). *Language models for automatic speech recognition: construction and complexity control*. PhD thesis, Helsinki University of Technology.

Siu, M. and Ostendorf, M. (2000). Variable n-grams and extensions for conversational speech language modeling. *IEEE Transactions on Speech and Audio Processing*, 8(1):63–75.

Stolcke, A. (1998). Entropy-based pruning of backoff language models. In *Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop*, pages 270–274.

Stolcke, A. (2002). SRILM – an extensible language modeling toolkit. In *Proceedings of the 4th International Conference on Spoken Language Processing (ICSLP)*, pages 901–904.

Szarvas, M. and Furui, S. (2003). Evaluation of the stochastic morphosyntactic language model on a one million word Hungarian task. In *Proceedings of the 8th European Conference on Speech Communication and Technology (Eurospeech)*, pages 2297–2300.

Tur, G., Stolcke, A., Voss, L., Dowding, J., Favre, B., Fernandez, R., Frampton, M., Frandsen, M., Frederickson, C., Graciarena, M., Hakkani-Tur, D., Kintzing, D., Leveque, K., Mason, S., Niekrasz, J., Peters, S., Purver, M., Riedhammer, K., Shriberg, E., Tien, J., Vergyri, D., and Yang, F. (2008). The CALO meeting speech recognition and understanding system. In *Proceedings of the IEEE Workshop on Spoken Language Technology*, pages 69–72.

Whittaker, E. and Raj, B. (2001a). Comparison of width-wise and length-wise language model compression. In *Proceedings of the 7th European Conference on Speech Communication and Technology (Eurospeech)*, pages 733–736.

Whittaker, E. and Raj, B. (2001b). Quantization-based language model compression. In *Proceedings of the 7th European Conference on Speech Communication and Technology (Eurospeech)*, pages 33–36.

Whittaker, E. and Woodland, P. (2000). Particle-based language modelling. In *Proceedings of the 4th International Conference on Spoken Language Processing (ICSLP)*, pages 170–173.

Williams, J. D. and Young, S. (2007). Partially observable Markov decision processes for spoken dialog systems. *Computer Speech and Language*, 21(2):231–422.

Woodland, P., Leggetter, C., Odell, J., Valtchev, V., and Young, S. (1995). The 1994 HTK large vocabulary speech recognition system. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 1, pages 73–76.

Xiang, B., Nguyen, K., Nguyen, L., Schwartz, R., and Makhoul, J. (2006). Morphological decomposition for Arabic broadcast news transcription. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages I–1089–I–1092.

Young, S. J., Russell, N. H., and Thornton, J. H. S. (1989). Token passing: a simple conceptual model for connected speech recognition systems. Technical report TR.38, Cambridge University Engineering Department.

Zheng, J. and Stolcke, A. (2005). Improved discriminative training using phone lattices. In *Proceedings of the Annual Conference of the International Speech Communication Association (Interspeech)*, pages 2125–2128.