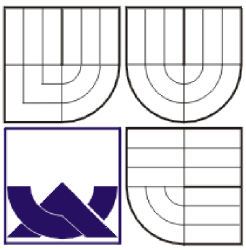
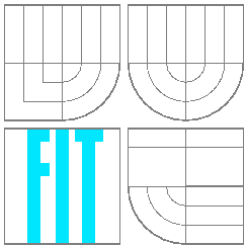


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

Hybridní slovní a pod-slovní detekce klíčových frází

Hybrid word-subword spoken term detection

DISERTAČNÍ PRÁCE
DOCTORAL THESIS

AUTOR PRÁCE
AUTHOR
Ing. Igor Szöke

VEDOUCÍ PRÁCE
SUPERVISOR
Doc. Dr. Ing. Jan Černocký

BRNO 2010

Abstract

The thesis investigates into keyword spotting and spoken term detection (STD), that are considered as sub-sets of spoken document retrieval. It deals with two-phase approaches where speech is first processed by speech recognizer, and the search for queries is performed in the output of this recognizer. Standard large vocabulary continuous speech recognizer (LVCSR) with fixed vocabulary is tested first, and its main drawback – incapability of detecting out-of-vocabulary words (OOV) is discussed. Subword systems that rely on phones or other subword units are also investigated, with the accent on subword units automatically inferred using constrained phone multigram approach. The next step is the creation of a hybrid spoken term detection system combining both word and subword parts in one recognition network. Extensive experiments investigating into different variants of this approach are performed, and the results (in terms of spoken term detection precision, speed, and necessary computing resources) are reported on standard data from NIST STD 2006 evaluation. The accuracy of the hybrid system was found marginally inferior to the combination of individual word and subword systems but this drawback is largely compensated by the simplicity and efficiency of the proposed system. The final tests were performed in combination with real indexing and search engine.

Keywords

keyword spotting, spoken term detection, confidence measures, large vocabulary continuous speech recognition, combined word-subword system, out-of-vocabulary

Bibliographic citation

Igor Szöke: *Hybrid word-subword spoken term detection*, Doctoral thesis, Brno, Brno University of Technology, Faculty of Information Technology, 2010

Abstrakt

Tato disertační práce se zabývá detekcí klíčových slov (keyword spotting) a frází (spoken term detection – STD), které jsou považovány za podmnožinu vyhledávání v řečových dokumentech (spoken document retrieval). Týká se dvoufázových přístupů, ve kterých je řeč nejprve přepsána rozpoznávačem, vyhledávání dotazů pak probíhá ve výstupu tohoto rozpoznávače. Na začátku testujeme standardní rozpoznávač spojité řeči s velkým a fixním slovníkem (large vocabulary continuous speech recognizer – LVCSR), a diskutujeme jeho hlavní nevýhodu – neschopnost detekovat slova mimo slovník (out-of-vocabulary words – OOV). Zkoumáme rovněž systémy založené na pod-slovních jednotkách (subword), důraz klademe na jednotky, které jsou automaticky určeny pomocí fonémových multigramů s omezujícími podmínkami. Dalším krokem je tvorba hybridního systému pro vyhledávání dotazů – ten kombinuje obě části (slovní i pod-slovní) v jedné rozpoznávací síti. V experimentech testujeme různé varianty tohoto přístupu, výsledky (přesnost detekce frází, rychlost, spotřeba výpočetních prostředků) uvádíme na standardních datech z NIST STD 2006 evaluace. Přesnost hybridního systému je o něco menší než u kombinace samostatného slovního a pod-slovního systému, tato nevýhoda je však převážena jednoduchostí a efektivitou námi navrženého přístupu. Konečné testování je provedeno v kombinaci se skutečným enginem pro indexování a vyhledávání v řeči.

Klíčová slova

detekce klíčových slov, detekce frází v řeči, míry konfidence, rozpoznávání spojité řeči s velkým slovníkem, kombinovaný slovní-podslovní systém, slova mimo slovník

Bibliografická citace

Igor Szöke: *Hybridní slovní a pod-slovní detekce klíčových frází*, Disertační práce, Brno, Vysoké Učení Technické v Brně, Fakulta informačních technologií, 2010

Prohlášení

Prohlašuji, že jsem tuto disertační práci vypracoval samostatně pod vedením Doc. Dr. Ing. Jana Černockého. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal. Některé systémy použité v práci byly vytvořeny členy výzkumné skupiny Speech@FIT, a nebo v rámci Evropských projektů. Toto je vždy explicitně uvedeno.

V Brně dne 7.6.2010

Contents

Abbreviations

ALL	ALL (means in-vocabulary and out-of-vocabulary terms)
ASR	Automatic Speech Recognition
ATWV	Actual Term-Weighted Value
CTS	Conversation Telephone Speech
DET	Detection Error Trade-off
FA	False Alarm (term detection)
FOM	Figure-Of-Merit
HIT	HIT (term detection)
IV	In-Vocabulary (word/term)
KWS	KeyWord Spotting
LM	Language Model
LP	Link Posterior
LSE	Lattice Search Engine
LVCSR	Large Vocabulary Continuous Speech Recognizer
MAM	Maximum Active Models (decoder parameter)
MTWV	Maximum Term-Weighted Value
NIST	National Institute of Standards and Technology (United States of America)
OOV	Out-Of-Vocabulary (word/term)
PAC	Phone ACcuracy
PER	Phone ERror
PHN	Phone
PLAC	Phone Lattice ACcuracy
PLER	Phone Lattice ERror
PR	PRuning (decoder parameter)
RT	Real-Time (factor)
SC	Subword Cost
SCOLP	Sum of Center Overlapped Link Posteriors
SLMSF	Subword Language Model Scaling Factor
SOLP	Sum of Overlapped Link Posteriors
STD	Spoken Term Detection
STK	Toolkit for speech processing developed at FIT BUT
SWIP	Subword Word Insertion Penalty
TWV	Term-Weighted Value (equal to MTWV)
TWV-IV	Term-Weighted Value of In-Vocabulary terms
TWV-OOV	Term-Weighted Value of Out-Of-Vocabulary terms
UBTWV	Upper Bound Term Weighted Value
UBTWV-IV	Upper Bound Term Weighted Value of In-Vocabulary terms
UBTWV-OOV	Upper Bound Term Weighted Value of Out-Of-Vocabulary terms
WAC	Word ACcuracy
WER	Word ERror
WFSA	Weighted Finite State Acceptor
WFST	Weighted Finite State Transducer
WLAC	Word Lattice ACcuracy
WLER	Word Lattice ERror

Chapter 1

Introduction

Speech is natural and the most wide spread way of information exchange among people. Many technologies were invented to allow better, faster and longer distance communication using speech, as telephones, radio and television. All of these techniques allow us to communicate (in one or both directions), but usually cannot store the speech communication for later processing. If the speech contained an important information, we usually cannot search for it after the speech communication finished. And if the speech communication was recorded, one has to listen to the whole recording to find the requested information. This is very expensive in terms of time and human resources. Our goal is to build a system for efficient and precise retrieval and fast access to the information in the stored "speech communication". On contrary to the speech search, textual data is accessible and search-able very quickly.

Firstly we need to understand what speech is. Speech is a way to transmit an information from one person to others. Speech contains a lot of information which can be divided to the following categories:

What was said Usually the most important information in speech. People speak for information exchange. This information can be split to two parts. *The first one* (the major and more important) can be transcribed to a sequence of words. *The second one* is hidden in speaker's prosody. The meaning of an utterance can be changed by the prosody sometimes.

Who said it Speaker's voice characteristics. We can identify person's age, gender, education or place of childhood just by his/her voice.

This thesis will target the "**What was said**" type of information without prosody information. The information is of discrete value (e.g. "*green* – color of an object") and is encoded into a sequence of (discrete) units (words and sentences) in speaker's brain. These units are then realized by muscle movements in our vocal tract. The result are continuous, very small and fast changes of air pressure going out of our mouth (speech). These air-pressure changes are transmitted in our environment until they arrive to listener's ear. They are converted to electrical impulses there. These are interpreted by the listener's brain as a sequence of units building up a (discrete) information. Information in listener's mind is not necessarily the same as in speaker's mind. If a difference appears, it is misunderstanding of information. For more details about speech production, language, listening and psychoacoustics see for example [DP73] and [HAH00].

We conclude that information exchange among people consists of:

Information is discrete and is what we want to "say".

Words are discrete units. Information is encoded into sequences which consist of words.

Phones are discrete units. Words can be converted to sequences of phones. Phone is elementary unit[1] which can be mapped to a certain muscle movements in vocal tract.

Speech are continuous air-pressure changes which are produced by muscle movements in vocal tract. Speech is transmitted from speaker to listener(s) through air.

Our goal is to efficiently store the information for later search. The only way how to get the information is from speech, because only speech is available "outside" the speaker. But, the information can be realized by many different sequences of words. Each of these sequences can be realized by infinite number of ways (speech air-pressure changes). There is other information incorporated in speech. Some of it has no direct link to speech like characteristics of speaker's voice or environment noise. The other has direct link to the information in speech as voice stress or prosody and can change the meaning of (information in) speech.

The process of information retrieval from speech can be split to several steps:

Recording and signal processing Contains recording of speech, storing of speech signal and basic signal processing operations as sampling, quantization and filtering. The input is spoken speech and the output is filtered digital signal.

Automatic speech recognition Once we have recorded speech, we can apply speech recognition to "discretization/tokenization" of speech to sequence of phones, words and their scores. Other information such as speed, loudness, stress, pitch and dialect can be also estimated. The output of speech recognizer is a data structure containing one or more hypothesis in parallel (lattice).

Indexing Techniques for space-optimal storing and fast access to the recognizer's output. Allows for search over large set of spoken documents in very short time.

Keyword spotting/Spoken term detection Algorithms which search recognizer's output for a given word or term. The output is a sequence of putative occurrences and their confidences. Position of the systems in the information retrieval chain is illustrated in figure 1.1.

Spoken document retrieval The output of spoken document retrieval system is a set of relevant documents according to given query. Spoken document retrieval is similar to Internet textual searchers (for example Google). The system should have very fast response and should handle large amounts of spoken data (documents).

Information retrieval Receives a "question" and tries to find an "answer" from information stored in a pool of documents. It can use output of speech recognizer, keyword spotter/term detector and other sources (speed, stress, pitch, etc.). It can also combine other modalities as textual search, video search (face detection, object detection and classification, etc.).

[1] In fact, an allophone is elementary acoustic unit of speech. Several allophones are mapped to one phone. The phone is logical unit and if a phone changes into another phone within the scope of a word, the meaning of the word changes too (or is meaning-less). If an allophone is changed to another within the scope of the same phone, then the word sounds different but has the same meaning.

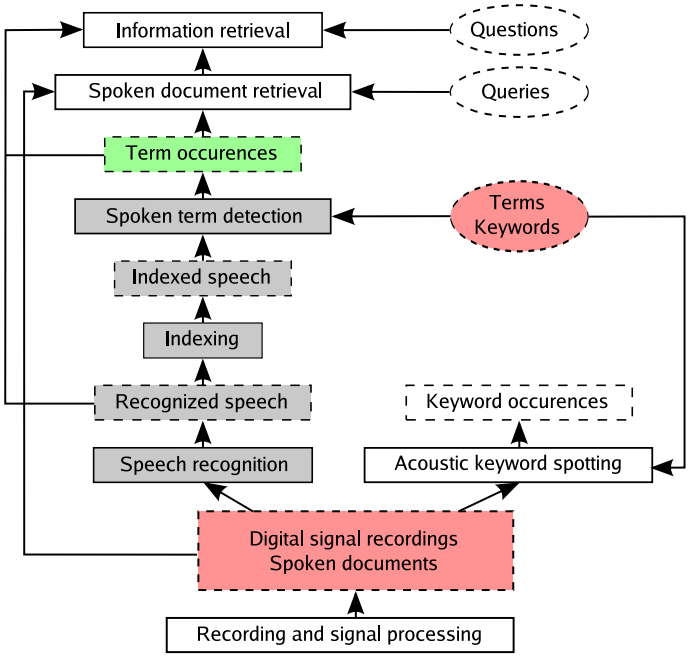


Figure 1.1: Role of Keyword spotting/Spoken term detection in Information retrieval from speech. The input (red) is processed by the chain of algorithms (gray) which produces occurrences of terms (green).

The research field of this thesis is spoken term detection. The corner stone of this thesis is search of out-of-vocabulary terms which are not present in dictionary of word-based speech recognizer. Also, topics as term confidence measures, weighted finite state transducers, indexing of spoken documents and phone multigram units are touched.

The motivation of this thesis is practical solution of search of out-of-vocabulary terms in spoken data. One important occasion was NIST spoken term detection evaluation[2] in 2006. Another occasions were our participation several large European projects: *M4*[3], *AMI*[4] and *AMIDA*[5]. A fast, accurate and robust approach of term search containing possibly out-of-vocabulary words was needed. Our previous research was aimed at combination of independent in-vocabulary (word) and out-of-vocabulary (subword) search. This led to the hybrid approach, which is investigated later in this thesis.

1.1 Scope of chapters

This work is organized as follows:

Chapter 2 describes **bases of spoken term detection**. Terminology and methods are described in this chapter. This chapter also contains spoken term detection survey which deals with recently published approaches. It concludes with the definition of terms as *Term posterior probability*, *Lattice*, *Term score estimation* and *Term confidence estimation*.

Chapter 3 contains description of **evaluation data and metrics**. Brief description of used tools is in the end of this chapter.

[2] `http://www.itl.nist.gov/iad/mig/tests/std/2006/index.html`
[3] `http://www.dcs.shef.ac.uk/spandh/projects/m4/index.html`
[4] `http://www.amiproject.org/`
[5] `http://www.amiproject.org/ami-scientific-portal`

Chapter 4 is related to description of used **large vocabulary continuous speech recognizer**. Training data and used techniques are listed there. Also, baseline experiment results are presented.

Chapter 5 deals with **phone**-based subword recognition. Baseline spoken term detection experiments are stated there.

Chapter 6 focuses on **phone multigrams**-based subword recognition. Firstly, the definition of phone multigram is provided. Section describing experiments for estimation of the best parameters of multigrams follows. Also, two new constrained multigram training techniques are proposed.

Chapter 7 investigates into **hybrid word-subword recognition**. The hybrid word-subword recognition network is described at the beginning of this chapter. Large set of experiments aiming at search for optimal hybrid combination of words and subwords is presented.

Chapter 8 describes **indexing and search** experiments with the hybrid word-subword recognizer. Several impacts of using hybrid word-subword lattices in the indexing and search engine for spoken term detection are outlined.

Chapter 9 concludes and discusses the results of this thesis.

1.2 Original claims of the thesis

The goal of this thesis is to investigate into combination of word and subword approaches to get the best search accuracy (especially for out-of-vocabulary words) having the highest search speed and the lowest memory consumptions. The original claims are as follows:

- **Proposal and evaluation of subword-based spoken term detector.** Improved phone multigram units are used. Constraints applied during training of multigrams improved the accuracy.

- **Subword system evaluation**. Dependency of the accuracy on computational resources and size of indices is studied.

- Proposal of **hybrid word-subword recognizer** with higher order language model over phone multigram subword units.

- Using the output of hybrid word-subword recognizer for **open-vocabulary spoken term detection task**. However, this is not completely new approach in general. Akbacak et al. [AVS08] converted subword units back to words and indexed only words. On contrary, we let the subword units in the word index, converted out-of-vocabulary terms to subword units and searched them in the index.

- Evaluation of **hybrid word-subword based spoken term detection accuracy** as function of computational resources and size of indices.

- Evaluation of **real speech indexing and search application** using the hybrid word-subword approach.

- **Upper-bound scoring method for OOV terms** based on NIST's Term Weighted Value

1.3 Acknowledgments

First, I would like to thank my supervisor Honza Černocký for his endless patience, support and guidance. I am grateful to him for guiding me during my doctoral studies.

I would like to thank all of my colleagues in Speech Group at Faculty of Information Technology in Brno especially: Lukáš Burget for great support and help, Martin Karafiát for the baseline word recognizer and Mišo Fapšo for his indexing and search system.

Finally, I would like to express thanks to my parents for supporting me during my early studies and to my wife Jana for support in writing this thesis.

My research has been supported by Faculty of Information Technology of Brno University of Technology, in part by EC projects Multi-modal meeting manager (M4), No. IST-2001-34485, Augmented Multi-party Interaction (AMI), No. 506811, AMIDA (FP6-033812), by Grant Agency of Czech Republic projects No. 102/02/0124, No. 102/08/0707 and No. 102/05/0278, Czech Ministry of Interior project No. VD20072010B16, and by Czech Ministry of Defense.

The hardware used in this work was partially provided by CESNET under project No. 201/2006.

Chapter 2

Bases of spoken term detection

Short definition of important terms is placed in the following paragraph to avoid confusion of the reader of this thesis. We define the differences between `keyword`, `term`, `query`, `keyword spotting` and `spoken term detection`.

Keyword is understood as a single word within the scope of this thesis (e.g. `"SOMETHING"`, `"DETECTION"` or `"IGOR"`). It is used within **acoustic keyword spotting** context. In fact, the keyword can be also sequence of consecutive words `"IGOR SZÖKE"` in context of *acoustic keyword spotting*. It is why these consecutive words can be processed as one keyword `"IGORSZÖKE"`.

Term is defined as one or multiple words in sequence like `"KEYWORD"`, `"KEYWORD DETECTION"` or `"THE PRESIDENT GEORGE BUSH"`. It is used within **spoken term detection** context. If the term consists of one word, there is no difference between *term* and *keyword*. For terms containing multiple words, the exact logic of how the words can be connected needs to be defined by the spoken term detector. For example, the `"KEYWORD DETECTION"` term can mean words `"KEYWORD"` and `"DETECTION"` in sequence where silence between them is shorter than 1s. Another words can be allowed between these two words. These conditions are defined in the spoken term detection system.

Query is defined as one or multiple words consisting of terms and operators `"('IGOR SZÖKE' near THESIS) and 'KEYWORD SPOTTING' not BIOLOGY"`. The operators should define the semantic information. The query is usually used in context of spoken document retrieval or information retrieval.

Keyword spotting system is a system for spotting (searching) given keywords in speech data. It understands the keyword as one object despite the number of words the keyword list might consist of. Keyword spotting system can be based on speech recognizer but it can be also "standalone" system which spots only given keywords and does not "understand" surrounding speech.

Spoken term detection system is also a system for spotting (searching) given terms in given speech data. On contrary to the *keyword spotter*, spoken term detector somehow parses and splits multiple word terms and searches for term candidates according to defined criteria (distance for example). The spoken term detection system is usually built-up on speech recognizer (and depends on it).

The topic of this thesis is aimed to **Spoken Term Detection** – STD. The STD system takes a set of terms and output of a speech recognizer and produces a list of putative hits

of given term. The term is understood as sequence of one or more consecutive words. Only short silence is allowed between these particular words. Term definition is discussed more thoroughly in section 3.1. Our spoken term detector is based on a **large vocabulary continuous speech recognizer** – LVCSR. It takes the output of speech recognizer and provides search of terms. The speech recognizer is mainly taken "as is" and is described in chapter 4. The output list of putative hits of given term can be viewed by human or processed by a system (information retrieval or spoken document retrieval) allowing for search for more complex queries.

The complexity of spoken term detector depends on the output of speech recognizer. Such output can be a 1-*best* output (simple text string, spoken term detection is then simple text search), an *N-best* output or a graph of parallel hypothesis so called **lattice** (for definition see section 2.3). The recognizer can recognize *word units* or *subword units* (syllables, phones, etc.).

Out-of-vocabulary (OOV) words handling is also important in case of word-recognition. Words which are not present in word recognizer dictionary should be detected. Normalization is useful for scaling and shifting of term confidences. Each term should have the **confidence** normalized, so that one global threshold can be used for decision of acceptance/rejection of terms. Speed and computational requirements are also important from practical point of view.

Search accuracy depends on recognition accuracy of used speech recognizer. We need only 1-best (single string) output in a case of 100% reliable speech recognizer. Nowadays, the state of the art word recognizers achieve about 10% – 20% word error rate (WER) and about 5% – 10% *lattice word error rate* (section 3.2) on broadcast news and *conversation telephone speech* (CTS) [Le04]. This gives very good search results in combination with lattice search [FAGD07]. But the language is an evolving thing and each day many new words appear. There can be hardly a speech recognizer having all words in the dictionary. Information theory also states that the least frequent words carry most of the information. That is why we aim at out-of-vocabulary words.

The problem of OOVs can be solved by recognizing subword units (syllables or phones). The drawback of this approach is absence of strong word n-gram language model and strong acoustic model of words which are both included in large vocabulary continuous speech recognizer (*LVCSR*). That is why subword recognition does not achieve so good accuracies. Phone recognition is quite sensitive to pronunciation errors for example. These possible errors should be taken into account in the search. On the other hand, LVCSR contains only a close set of words to be recognized and word language model prefers likely word sequences off the "exotic" ones (probably carrying higher information). Also it is shown that if an OOV appears, it usually causes no 1 word error, but approximately 2 – 4 word errors [BN05a]. This is a justification of an investigation into subword recognition.

2.1 Keyword spotting techniques

The following sections deal with brief overview of spoken term detection / keyword spotting methods. Keyword spotting approaches can be classified into two main categories. The first one is acoustic ("direct") keyword spotting and the second is spoken term detection and ("indirect") keyword spotting which is based on the output of a speech recognizer.

2.1.1 Acoustic keyword spotting

Before we start formal and mathematical description of spoken term detection, simple informal principle of keyword spotting will be demonstrated on acoustic keyword spotter. The *acoustic keyword spotter* was proposed in 1989 [RRRG89]. The principle is in the detection of defined set of keywords in utterances and in attaching a confidence score to each detection. The confidence score should be independent on the keyword and data.

Keyword spotting system contains several parts (figure 2.1). **Speech signal is parametrized** to stream of features. Parametrization is out of the scope of this thesis (see reference books [You99] or [HAH00] for details).

Stream of features is then used as the input of **models**. One model is modeling keyword(s) and the other is modeling the general speech (so-called *background model*). The outputs of the keyword and background models are likelihoods.

Then the likelihoods are divided and a **score** (likelihood ratio) of a particular occurrence of the keyword is obtained. Why the score is calculated as likelihood ratio is discussed in section 2.2.1.

A set of putative detections of keywords are obtained by thresholding of the score. A trade-off between more or less detections is obtained by setting of the **threshold**.

The **confidence score estimation** can be inserted between "Likelihood ratio" and "Filtering" box. The confidence score should be normalized and take into account the length of keyword, phones of which keyword consists, prior probabilities and other factors which make scores of different keywords from different parts of the utterance comparable. This is important for real applications and evaluation metrics (DET curve (section 3.3.2) or TWV (section 3.3.3)) where one global threshold is set. Normalization is also important for evaluation metrics or applications where each keyword has its own threshold (FOM (section 3.3.1)).

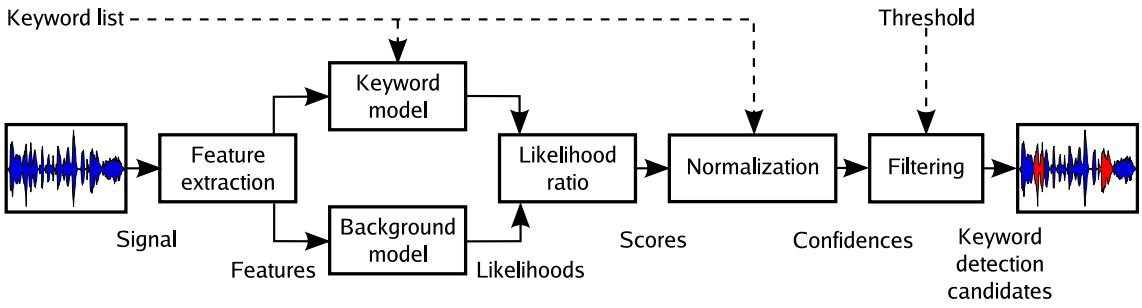


Figure 2.1: General scheme of acoustic keyword spotting system.

2.1.2 Spoken term detection

The generic scheme of a spoken term detection system (figure 2.2) is similar to that of acoustic keyword spotter (figure 2.1). The spoken term detection system is built on speech recognizer, which usually encapsulates also the feature extraction. The speech recognizer produces textual strings or so-called lattices (section 2.2.2) which contain transcribed speech in words labels. The lattices are searched for the given terms or keywords.

The estimation of the term score is done in different way compared to the acoustic keyword spotting, but the theoretical background is the same. The goal is to calculate the score as the proportion of scores of the term and background. Mathematically it is described in section 2.2.1.

The following steps are the same as are in the acoustic keyword spotting: confidence score estimation and the filtering of hypothesis of terms according to given threshold.

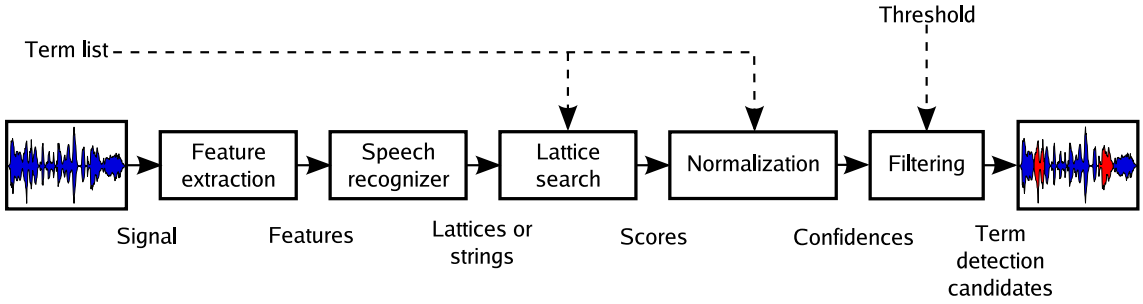


Figure 2.2: General scheme of spoken term detector.

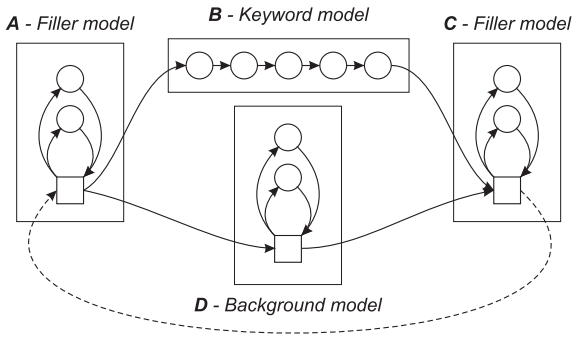


Figure 2.3: General model of acoustic keyword spotting system.

2.1.3 Keyword spotting score estimation

Simplified explanation of the keyword score estimation is done on acoustic keyword spotter. Figure 2.3 shows a general recognition network for keyword detection based on phone models without any language model. Parts denoted `A` and `C` are filler models (phone loops) which model non-keyword parts of utterance. Part `B` is linear model for given keyword created by concatenation of keyword phone models. Part `D` is the background model (also phone loop) which models the same part of utterance as the keyword model. The loop-back from model `D` to model `A` is added for ability to detect more than one keyword per utterance.

The utterance is modeled using model `A-B-C` concatenated of models `A`, `B` and `C`, and using model `A-D-C` concatenated of models `A`, `D` and `C`. We have to do an assumption, that models `B` and `D` will recognize exactly the same part of utterance. So, the final likelihood of model `A-B-C` (L_{ABC}) and `A-D-C` (L_{ADC}) should differ only because of models `B` and `D`. If the part of utterance beneath model `B` is not a keyword, the likelihood of the model `B` should be low (bad match to the keyword model), but the likelihood of the model `D` will be high (good match to the phone loop). In the opposite case, when the part of utterance beneath model `B` is the keyword, the likelihood will be high and so will the likelihood of model `D` (the likelihoods will be the same in ideal case). Now, we compute the ratio of likelihoods `A-D-C` and `A-B-C`: $L_{Ratio} = L_{ADC}/L_{ABC}$. If there is a keyword beneath model `B`, L_{Ratio} will approach 1 and will be lower for non-keywords. If a noise appears in

the speech, both likelihoods L_{ABC} and L_{ADC} will be lower, but because of the likelihood ratio, the influence of noise should be limited.

2.1.4 Comparison of keyword spotting techniques

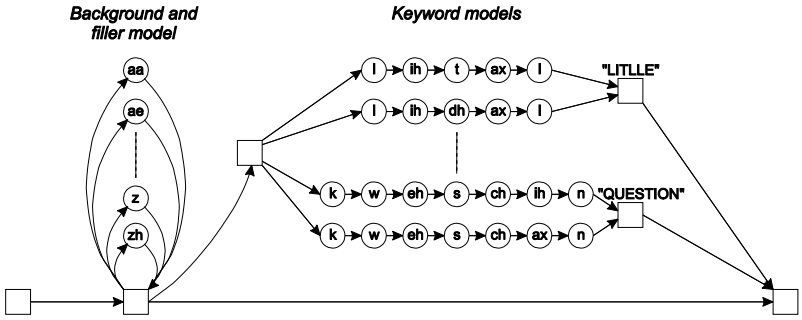


Figure 2.4: An example of acoustic keyword spotting network.

Acoustic keyword spotting is a search for a keyword in parametrized spoken data. Searched word(s) in textual form is converted to sequence of units. These units correspond to acoustic models (monophones, triphones, etc.). Specialized recognition network (keyword spotting network) is then created and speech is recognized. The output is not a classical word or phone string or lattice, but a list of putative keyword candidates. Acoustic keyword spotting can be very simple, fast and inaccurate (monophones models without any language model) but also more complex, slower and more accurate (triphones, at least simple word language model). An example of an acoustic keyword spotting network is in figure 2.4. Advantages of this method are its speed and on-line functionality. Simple keyword spotter (having hundreds of words) can be easily attached to a telephone line and can rise alarms with no more than 0.2-2s delay. Drawbacks are absence (or simplicity) of word language model and "hard-wired" set of keywords. Simpler language model leads to lots of false alarms (especially for short keywords). The detector can detect only the keywords which are built in the recognition network. Once the set of keywords is changed, the network has to be rebuild. If a new keyword is to be searched in already recorded data, the keyword spotting process must be rerun over the stored data. This is considerably time consuming even if the acoustic keyword spotter is faster than real-time. That is why this approach is not suitable for fast STD. On the other hand, acoustic keyword spotting does not suffer from the out-of-vocabulary problem. The qualities of acoustic keyword spotting approach are summarized in table 2.1.

Spoken term detection (indirect keyword spotting) is based on the output of a speech recognizer. It is a two step method where the first step consists of the time consuming speech recognition and the second one consists of a fast spoken term/keyword search. The method inherits main characteristics of the recognizer used. Input term/keyword must be converted to a sequence of units similar to recognizer's output units (e.g. words, syllables, phones, etc.). Then the sequence is searched in the output of the recognizer. The recognizer (usually the slowest step of whole STD) is run only once. The STD or keyword spotter is run each time a term or keyword has to be found. In comparison to the acoustic keyword spotting, the

search is very fast because it is done over "textual data" (output of speech recognizer). Advantages of STD are the speed of search and detection accuracy (depends on recognizer's accuracy). Searching speed can be optimized by techniques known in information retrieval, such as inverted indices, caching etc. to achieve searching times less than $10^{-3}s/hr/term$. The disadvantages are off-line processing (especially LVCSR is complex and time consuming) and closed unit vocabulary. The recognizer has finite and closed vocabulary of units it can recognize. Once the recognition is done, the spoken term detector will "find" only units which were recognized by the recognizer. This is a drawback if a word recognizer is used. STD approach can be split according to used recognizer to word-based and subword-based. The word-based STD has very high accuracy (having phone models "organized" in words and strong word language model) but limited vocabulary. The subword-based STD approach has unlimited vocabulary (search word must be converted to a sequence of subword units) but has lower accuracy (missing word acoustic models and word language model). The qualities of STD approaches (word and subword) are summarized in table 2.1.

Property	Direct	Indirect	
	Acoustic keyword spotting	Word-based STD	Subword-based STD
Recognition speed	Slow	Slow	Slow
Search speed		Fast	Fast
Search in	Raw speech	Words (text)	Subwords (text)
Vocabulary	Open[1]	Closed	Open
Accuracy	Low – Average	Average – High	Low – Average
Mode	On-line or Off-line	Off-line	Off-line

Table 2.1: Advantages and drawbacks of different keyword spotting techniques.

2.2 Bayesian probability

This section deals with the principles of spoken term detection and term score and confidence estimation from mathematical point of view. Correct and reliable estimation of the term / keyword score is crucial part of spoken term detection (or keyword spotting). Two main confidence measures were proposed in the literature (see section 2.4.6). It is the *Hypothesis Testing* which is the key technique of frequentist statistical inference and the *Posterior Probability* which is based on the Bayesian view of the probability.

Hypothesis testing approach is based on the test of *null hypothesis* H_0 against *alternative hypothesis* H_1. For example, the null hypothesis means that the keyword exists and is correctly recognized in a portion of speech. The alternative hypothesis means that there is no keyword or the keyword is incorrectly recognized. According to the Neyman-Pearson Lemma, the optimal solution of hypothesis testing is the likelihood-ratio test.

The Bayesian view of probability is considered as more general, Bayesians describe probabilities in terms of beliefs and degrees of uncertainty. In the field of pattern recognition (speech recognition), it is helpful to have a more general notion of probability such as the Bayesian is [Bis06].

[1]Open – Recognition step must be run again for a keyword which is not in the keyword list.

In general, speech recognition is based on *Bayesian* view of probability. Speech recognizer, spoken term detector, speaker recognition system and other systems work in probabilistic domain. Recognized hypothesis (word, term, speaker) have always probabilities assigned. Several following paragraphs are adopted from Bishop's Pattern Recognition and Machine Learning book [Bis06].

Let us define $\mathbf{w}$ as recognized sequence of M units (words or phones) $\mathbf{w} = \{w_1, w_2, \ldots, w_M\}$ in given utterance u. The utterance u is represented by observed data. This data is processed into sequence of N feature vectors $\mathcal{D} = \{\mathbf{o}_1, \ldots, \mathbf{o}_N\}$. The observed data $\mathcal{D}$ is expressed through the conditional probability $p(\mathcal{D}|\mathbf{w})$. Bayes' theorem, which takes the form

$$p(\mathbf{w}|\mathcal{D}) = \frac{p(\mathcal{D}|\mathbf{w})p(\mathbf{w})}{p(\mathcal{D})} \tag{2.1}$$

then allows us to evaluate the uncertainty in $\mathbf{w}$ after we have observed $\mathcal{D}$ in the form of the *posterior probability* $p(\mathbf{w}|\mathcal{D})$.

The quantity $p(\mathcal{D}|\mathbf{w})$ on the right-hand side of Bayes' theorem is evaluated for the observed data set $\mathcal{D}$ and can be viewed as a function of the sequence of units $\mathbf{w}$, in which case it is called the *likelihood function*. It expresses how likely the observed data set is for different recognized sequences of units $\mathbf{w}$. Note that the likelihood is not a probability distribution over $\mathbf{w}$, and its integral with respect to $\mathbf{w}$ does not equal one.

Given this definition of likelihood, we can state Bayes' theorem in words

$$posterior \propto likelihood \times prior, \tag{2.2}$$

where all of these quantities are viewed as functions of $\mathbf{w}$. The denominator in 2.1 is the normalization constant, which ensures that the posterior distribution on the left-hand side is a valid probability density and integrates to one. Indeed, integrating both sides of 2.1 with respect to $\mathbf{w}$, we can express the denominator in Bayes' theorem in terms of the prior distribution and the likelihood function

$$p(\mathcal{D}) = \int p(\mathcal{D}|\mathbf{w})p(\mathbf{w})d\mathbf{w}. \tag{2.3}$$

2.2.1 Speech recognition and posterior probability

Conventional automatic speech recognition algorithm uses the maximum a posteriori (MAP) decision rule to find the most likely sequence of units $\hat{\mathbf{w}}$ which achieves the maximum posterior probability $p(\mathbf{w}|\mathcal{D})$ given acoustic observation $\mathcal{D}$

$$\hat{\mathbf{w}} = \arg\max_{\mathbf{w}\in\mathcal{W}} p(\mathbf{w}|\mathcal{D}), \tag{2.4}$$

where $\mathcal{W}$ is the set of all permissible unit sequences. This equation can be rewritten using *Bayes formula* (equation 2.1) to form:

$$\hat{\mathbf{w}} = \arg\max_{\mathbf{w}\in\mathcal{W}} \frac{p(\mathcal{D}|\mathbf{w})p(\mathbf{w})}{p(\mathcal{D})}, \tag{2.5}$$

where $p(\mathcal{D}|\mathbf{w})$ is the likelihood of observing $\mathcal{D}$ by assuming that $\mathbf{w}$ is the underlaying sequence of units. The likelihood $p(\mathcal{D}|\mathbf{w})$ can modeled by Gaussian mixtures hidden Markov model for example. $p(\mathbf{w})$ is the prior probability of $\mathbf{w}$ and is modeled by language model (LM). And finally, $p(\mathcal{D})$ is the probability of observing the acoustic data $\mathcal{D}$. Because $p(\mathcal{D})$ is constant across different unit sequences, most automatic speech recognition systems

simply ignore it. The formula for simplified likelihood of unit sequence (it is not probability yet, $p(\mathbf{w}|\mathcal{D}) \approx L(\mathbf{w})$) follows:

$$L(\mathbf{w}) = p(\mathcal{D}|\mathbf{w})p(\mathbf{w}) \tag{2.6}$$

The non-normalized raw score (likelihood $L(\mathbf{w})$) does not represent the absolute quantitative measure of the match between $\mathcal{D}$ and $\mathbf{w}$. Reliable confidence measure can be accomplished by normalization of the score by $p(\mathcal{D})$. Theoretically, $p(\mathcal{D})$ should be computed as follows:

$$p(\mathcal{D}) = \sum_{\mathbf{w}' \in \mathcal{W}} p(\mathcal{D}|\mathbf{w}')p(\mathbf{w}'), \tag{2.7}$$

where $\mathbf{w}'$ denotes any hypothesis of all possible hypothesis of unit sequences $\mathcal{W}$ (combinations of words, phones, silences etc.), without any other constraint. It is impossible to enumerate all these hypothesis. Some approximations must be done to estimate $p(\mathcal{D})$.

One possibility is to use so-called *background model* for estimation of $p(\mathcal{D})$. The background model can be realized for example by all-phone recognition or "catch-all" model. This approach is mainly used in acoustic keyword spotting. It is the `A-D-C` model stated in previous section 2.1.

In case of 1-best output of the LVCSR system, it is not straightforward to calculate the posterior probability (normalized scores) attached to hypothesized sequence of units. This has to be done internally in the decoder by using anti-models, background models or other techniques. However, with multi-hypothesis decoders the estimation of $p(\mathcal{D})$ is more straightforward:

Decoders producing lattices are nowadays standard. Detailed description of a lattice is in the following section 2.3. The lattice is a compact representation of N most likely hypothesis $\mathcal{W}'$. If the N is sufficiently large, the lattice approximates space of all possible hypothesis of unit sequences $\mathcal{W}$. The approximated form of equation 2.7 is simply

$$p(\mathcal{D}) = \sum_{\mathbf{w}' \in \mathcal{W}'} p(\mathcal{D}|\mathbf{w}')p(\mathbf{w}'). \tag{2.8}$$

In practice, the calculation of $p(\mathcal{D})$ (equation 2.7) is well approximated by so-called *forward-backward algorithm* for given set of hypothesis $\mathcal{W}'$ defined by the lattice. Description of the forward-backward algorithm is provided in following section 2.3 dealing with STD in lattices.

2.2.2 Spoken term detection

In the previous section, speech recognition task was described using the Bayesian framework. In STD, we are not "asking" for the most likely sequence of units $\hat{\mathbf{w}}$ according to the acoustic observation $\mathcal{D}$, but we "ask" for the posterior probability $p(term_{t_b}^{t_e})$ of occurrence of the term $term$ from time t_b to time t_e. The sequence of units $\mathbf{w}$ is constrained to $\mathbf{w}(term_{t_b}^{t_e})$ which contain the term in given time. By application of the MAP decision rule, the equation 2.4 is rewritten to

$$\hat{\mathbf{w}}(term_{t_b}^{t_e}) = \underset{\mathbf{w}(term_{t_b}^{t_e}) \in \mathcal{W}(term_{t_b}^{t_e})}{\arg\max} p(\mathbf{w}(term_{t_b}^{t_e})|\mathcal{D}), \tag{2.9}$$

where $\mathcal{W}(term_{t_b}^{t_e})$ is the set of all permissible sentences having the term in defined time. Applying the Bayes formula 2.1 and equation 2.7, we get

$$\hat{\mathbf{w}}(term_{t_b}^{t_e}) = \underset{\mathbf{w}(term_{t_b}^{t_e}) \in \mathcal{W}(term_{t_b}^{t_e})}{\arg\max} \frac{p(\mathcal{D}|\mathbf{w}(term_{t_b}^{t_e}))p(\mathbf{w}(term_{t_b}^{t_e}))}{\sum_{\mathbf{w}' \in \mathcal{W}} p(\mathcal{D}|\mathbf{w}')p(\mathbf{w}')}. \tag{2.10}$$

In practice, direct implementation of formula 2.10 is difficult. We do not know the time of occurrence t_b and t_e of the term *term*. Again, an approximation must be used to hypothesize t_b and t_e. The time of the term can be suggested from $\mathcal{W}$. To avoid having size of $\mathcal{W}$ infinite, $\mathcal{W}$ is approximated by lattice similarly as at the end of previous section 2.2.1.

So, the real spoken detection task has two steps. The set of the most likely hypothesis $\mathcal{W}'$ is generated using formula 2.5. Then occurrences of searched terms are found in $\mathcal{W}'$ and estimation of term posterior probability $p(term_{t_b}^{t_e})$ is:

$$p(term_{t_b}^{t_e}) = \frac{p(\mathcal{D}|\mathbf{w}(term_{t_b}^{t_e}))p(\mathbf{w}(term_{t_b}^{t_e}))}{\sum_{\mathbf{w}' \in \mathcal{W}'} p(\mathcal{D}|\mathbf{w}')p(\mathbf{w}')}. \tag{2.11}$$

2.3 Search in lattice

This section presents the "implementation" of the calculation of term posterior probability stated in equation 2.11 in the previous section. Lattice (figure 2.5) are nowadays used as the multiple hypothesis output of speech recognizer.

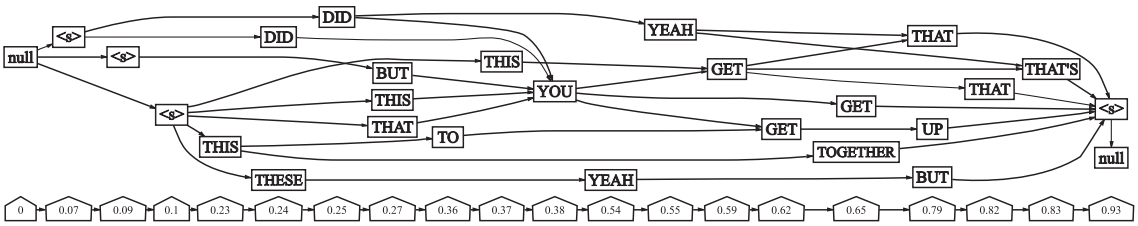


Figure 2.5: An example of word lattice. X-axis represents time.

The lattice is an acyclic oriented graph. Each node n represents a time. An arc a connects two nodes n_1, n_2 and represents a speech unit[2] $u = U(a)$ and set of two likelihoods $L(a)$ (acoustic $L_{Ac}(a)$ and language $L_{LM}(a)$). Start time $t_b(a)$ and end time $t_e(a)$ of arc a representing unit $U(a)$ correspond to the time of start node $t(n_b(a))$ and end node $t(n_e(a))$ of the arc a:

$$\begin{aligned} t(n_b(a)) &= t_b(a) \\ t(n_e(a)) &= t_e(a). \end{aligned}$$

The $L_{Ac}(a) \propto p(\mathcal{D}|\mathbf{w}(a_{t_e}^{t_b}))$ and $L_{LM}(a) \propto p(\mathbf{w}(a_{t_e}^{t_b}))$. According to equation 2.6, we can write $L(a) = L_{Ac}(a)\ L_{LM}(a)$.

The best hypothesis (the most likely path) can be derived from lattice. The best path through the lattice is also known as 1-*best* or *string* output. N most likely paths through the lattice are known as N-*best* output. Lattice can be understood as compact representation of the N-best output where the N is a large number.

Searching for a term in the string output (1-best) is straightforward. An algorithm goes through the string of units and compares each term to a sequence of units. If the comparison is successful, time boundaries and likelihood of units are stored to a list of term detections. The drawback of this approach is the absence of normalization. Term scores, which are derived from likelihoods of units ($L(u)$) according to equation 2.6, are sensitive to background noises. The term detector is not robust in this case.

[2]Another possibility is to represent speech unit as end node of the arc, then the arc represents only time information and likelihoods.

On the other hand, searching for the term in the lattice is more robust. Having the lattice, we have the $\mathcal{W}'$ and we can estimate the posterior probability of term according to equation 2.11. The posterior probability gives confidence of term for particular occurrence of term (represented by arc a) in time $t_b(a),\ t_e(a)$.

However, one more problem should be solved. Assume, that we also hypothesized occurrence a' of the term, which is slightly shifted but still overlapped with the original one. The problem is: is the probability of the original occurrence affected by the fact that several overlapped occurrences of the same term exist? This leads to "alternative" formula estimating the posterior probability of the term in time t: $t(term) = term_t$. The occurrence of term in time $t(term)$ is defined by condition $t_b(term) \leq t(term) \leq t_e(term)$.

These two points of view are defined in this thesis in the following way:

1. The **term score**. The term score is the posterior probability $p(term_{t_b}^{t_e})$ of particular term hypothesis in the lattice from time t_b to time t_e (figure 2.6). It does not consider other overlapped occurrences of the term in the time. Section 2.3.1 deals with the computation of the term score.

2. The **term confidence**. On the other hand, term confidence is the posterior probability $c(term_t) = p(term_t)$ of existence of the term in the lattice at given time t (figure 2.6). It takes into account several overlapped particular term hypothesis in the lattice. Section 2.3.2 deals with estimation of the term confidences.

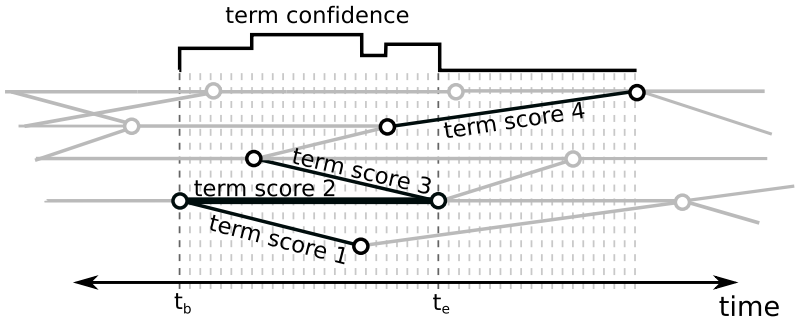


Figure 2.6: Example of a term occurrences in a lattice. "Term scores" denote different values of the posterior probability $p(term_{t_b}^{t_e})$ for particular term occurrence. "Term confidence" denotes evolution of posterior probability $p(term_t)$ of existence of the term in the lattice at given time t.

2.3.1 Term score estimation

Let us define likelihood $L^\alpha(n)$ from the beginning b of a lattice to node n and let it be called *forward likelihood*. The likelihood from node n to the end e of the lattice is denoted $L^\beta(n)$. We can also imagine this as the likelihood from the end of a lattice e to node n and call it *backward likelihood*. These two likelihoods (L^α and L^β) mean, how likely is to go from the beginning of the lattice to a node, or from a node to the end of the lattice. All possible hypothesis (paths) should be taken into account in evaluation of forward and

backward likelihoods. The mathematical definition is the following:

$$L^{\alpha}(n) = \sum_{\mathbf{w}\in\mathcal{W}'^{n}_{b}} \prod_{i=1}^{N} L(w_i) \tag{2.12}$$

$$L^{\beta}(n) = \sum_{\mathbf{w}\in\mathcal{W}'^{e}_{n}} \prod_{i=1}^{N} L(w_i) \tag{2.13}$$

where $\mathcal{W}'^{n}_{b}$ is a set of all possible paths from beginning of the lattice to node n. $\mathcal{W}'^{e}_{n}$ is a set of all possible paths from node n to the lattice end. w_i is the i-th arc of path $\mathbf{w}$ having N arcs (units). In practice, equation 2.12 is approximated by so-called forward-backward algorithm, which can be easily implemented and is not computationally expensive.

Let us have a term which consists of only one unit (for example one word term). This term is represented by one arc in the lattice. The likelihood of all paths in the lattice *latt* containing (going through) this term *term* is defined:

$$L^{latt}(term) = L^{\alpha}(n_b(term))L(term)L^{\beta}(n_e(term)). \tag{2.14}$$

More generally, the likelihood of all paths in the lattice *latt* containing (going through) a certain term **term** consisting of more consecutive units is defined:

$$L^{latt}(\mathbf{term}) = L^{\alpha}(n_b(term_1)) \left[\prod_{i=1}^{M} L(term_i)\right] L^{\beta}(n_e(term_M)) \tag{2.15}$$

and holds for $n_e(term_i) = n_b(term_{i+1})$, and where the term **term** consists of M units $\{term_1,\ term_2,\ \ldots\ ,\ term_M\}$ which can be either words or phones.

If we look back to the section 2.2 aimed to spoken term detection from Bayesian view, the $L^{latt}(term)$ can be understood as likelihood of term *term* in lattice *latt* which is equal to $p(\mathcal{D}|\mathbf{w}(term^{t_e}_{t_b}))p(\mathbf{w}(term^{t_e}_{t_b}))$ (eq. 2.10).

Actually $p(\mathcal{D})$ must be evaluated (equations 2.1 and 2.8) to get correct posterior probability (equation 2.11 of the unit sequence $w(term^{t_e}_{t_b})$. Equation 2.8:

$$p(\mathcal{D}) = \sum_{\mathbf{w}'\in\mathcal{W}'} p(\mathcal{D}|\mathbf{w}')p(\mathbf{w}') \tag{2.16}$$

can be easily "implemented" in the lattice by:

$$L^{\alpha}(n_e(latt)) = \sum_{\mathbf{w}\in\mathcal{W}'^{e}_{b}} \prod_{i=1}^{N} L(w_i), \tag{2.17}$$

which means: sum likelihood of all possible paths through the lattice.

Finally, the complete form of the term posterior probability $p(\mathbf{term}^{t_e}_{t_b})$ estimated over the lattice *latt* is:

$$p^{latt}(\mathbf{term}^{t_e}_{t_b}) = \frac{L^{\alpha}(n_b(term_1)) \left[\prod_{i=1}^{M} L(term_i)\right] L^{\beta}(n_e(term_M))}{L^{\alpha}(n_e(latt))} \tag{2.18}$$

The posterior probability $p^{latt}(\mathbf{term}^{t_e}_{t_b})$ means ratio of the likelihood of going through the term **term** to the likelihood of going through all possible paths in the lattice *latt*. The following notation of the term *term* will be used further in this section for simplicity. But it can represent both one word long term (*term*) and a sequence of words (**term**).

2.3.2 Term confidence estimation

Previous section 2.3.1 deals with estimation of posterior probability $p^{latt}(term_{t_b}^{t_e})$ of a particular occurrence of the term $term_{t_b}^{t_e}$ in lattice *latt*. We decided to denote, that $p^{latt}(term_{t_b}^{t_e})$ is the *term score*. In this section, we aim at posterior probability $p^{latt}(term_t)$ of the term in the lattice at given time t, that we denote as *term confidence*. The notation $c(term_t)$ is define for better distinction between term score $p(term)$ and term confidence $c(term)$. We assume that all operations are performed over one given lattice. That is why, the index *latt* is dropped in the following formulae.

Wessel [WSMN01] proposed 3 methods of evaluation of term confidence:

1. Sum of scores of all particular term occurrences which are overlapped with selected term occurrence in time.
2. Sum of scores of all particular term occurrences which are overlapped with selected term occurrence in mid-time (time of the center of the term occurrence).
3. Sum of scores of all particular term occurrences which are overlapped with selected term occurrence in particular time. The final score is maximum of all particular scores during the time of the selected term. This term confidence was denoted and is known as C_{max}.

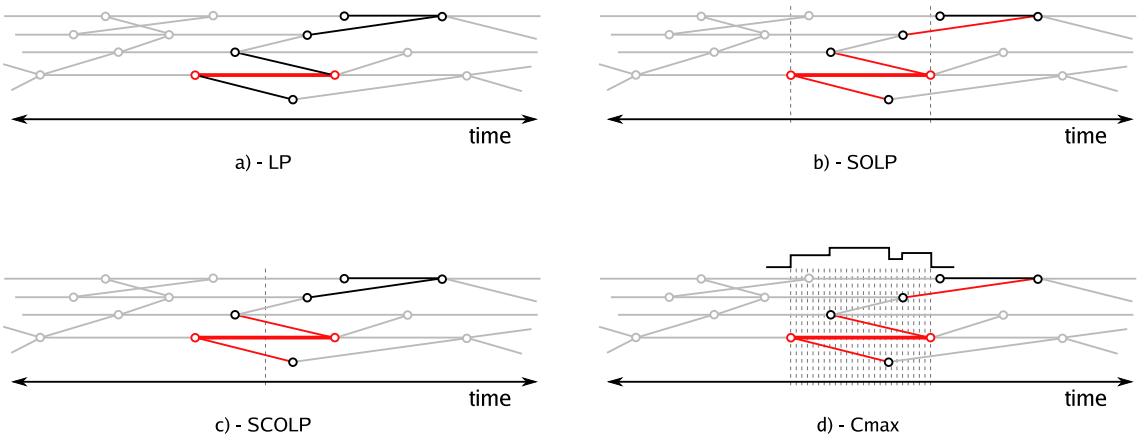


Figure 2.7: Different confidence estimation approaches of a term in a lattice. Given term detection is noted by thick red line and red nodes. Another candidates of the term are in black. If the candidate is considered for confidence estimation of the term, then it is in red.

In this thesis, these three approaches and a baseline approach are evaluated. These methods are denoted:

LP – Link Posterior Posterior probability of term candidate link (sequence of links) is assigned as the confidence of term candidate. The same term detections overlapping with the candidate have no effect on the candidate confidence (figure 2.7a). This is the baseline method.

$$c_{LP}(term_t) = p(term_{t_b}^{t_e}), \quad t_b \leq t \leq t_e \tag{2.19}$$

SOLP – Sum of Overlapped Link Posteriors Sum of posterior probabilities of all term detections which are overlapped with the term candidate in time (figure 2.7b).

$$c_{SOLP}(term_t) = \sum_{term' \in \mathcal{T}} p(term'^{t_e}_{t_b}), \tag{2.20}$$

where $\mathcal{T}$ is set of terms $term'$ containing all terms from the lattice satisfying conditions:

$$t_b(term') < t_e(term) \quad and \tag{2.21}$$
$$t_e(term') > t_b(term). \tag{2.22}$$

SCOLP – Sum of Center Overlapped Link Posteriors Sum of posterior probabilities of all term detections which are overlapped time with the center of the term candidate in time (figure 2.7c).

$$c_{SCOLP}(term_t) = \sum_{term' \in \mathcal{T}} p(term'^{t_e}_{t_b}), \tag{2.23}$$

where $\mathcal{T}$ is set of terms $term'$ containing all terms from the lattice satisfying conditions:

$$t_b(term') < t_c(term) \quad and \tag{2.24}$$
$$t_e(term') > t_c(term), \tag{2.25}$$

$t_c = t_b + (t_e - t_b)/2$ is the center time of the term.

C_{max} **– Maximum of frame-by-frame overlapped link posteriors sum** Sum of posterior probabilities of all term detections which are overlapped with the x-th frame of term candidate in time. This calculation is evaluated for all frames of the candidate and then the maximum is chosen and set as the term candidate confidence. This approach is widely denoted by C_{max} [WSMN01] (figure 2.7d).

$$c_{max}(term_t) = \underset{t \in (t_b(term), t_e(term))}{\arg\max} \sum_{term' \in \mathcal{T}} p(term'^{t_e}_{t_b}), \tag{2.26}$$

where $\mathcal{T}$ is set of terms $term'$ containing all terms from the lattice satisfying conditions:

$$t_b(term') < t_e(term) \quad and \tag{2.27}$$
$$t_e(term') > t_b(term) \tag{2.28}$$

After the confidence estimation is applied, only the candidate with the best confidence in the overlapped group of term detections is reported.

Comparison of these confidence estimation techniques is given in sections 4.4 and 6.5.1. If not stated else, the c_{LP} confidence will be used further in this thesis.

2.4 Survey of spoken term detection

Overviews of spoken term detection techniques and close domains are presented in this section. Brief history of the keyword spotting and spoken term detection – STD is described first. This section is followed by a section on spoken document retrieval – SDR. Although, the out-of-vocabulary words and recognition errors were not found "dangerous" from accuracy point of view of the SDR task, there are still several areas where the out-of-vocabulary words can cause serious deterioration of accuracy. Section *Do we need OOV handling?* concludes these facts.

Basic methods of handling out-of-vocabulary words in queries and terms are briefly outlined in section *How to handle OOV*. Part of these methods are aimed to spoken document retrieval task but they can be also applied in spoken term detection. Special set of methods proposing the use of hybrid word-subword language models is discussed in section *Hybrid language model*.

The final part of this section (*Confidence measures*) describes and concludes on approaches for confidence estimation of words and terms.

Note: the terms "term score" and "term confidence" are taken from the cited paper in this survey. The meaning can differ from the definition in the previous section.

2.4.1 History and present of keyword spotting

The problem of detecting a keyword in running speech has been approached in several different ways. Bridle [Bri73] originally introduced dynamic programming techniques for whole-word template matching. Every keyword template is matched against every portion of input speech and a score is computed using dynamic programming in these systems. Keyword detections are established by thresholding of the score. Higgins and Wohlford [HW85] proposed template-based dynamic time warping speech recognition system. They defined *filler template* to represent out-of-vocabulary speech. One of the first hidden Markov model based keyword spotters was proposed by Rohlicek et. al. [RRRG89] in 1989. The keyword spotter consists of a keyword model and a filler model. Computation of keyword score is based on maximum likelihood approach presented by Bahl [BJM83]. That inspired Rose and Paul [RP90] for keyword spotting system based on continuous speech recognition model. Filler model contains word models in this case.

Keyword spotters based on Rohlicek et. al. [RRRG89] are usually called **acoustic keyword spotters**. This approach can be used either as on-line keyword spotter or as off-line one. Possible improvement of the output of acoustic keyword spotting can be done by post-processing of detected keywords. One possible approach of such post-processing is acoustic modeling of keywords [GNR92], estimation of keyword confidence or using anti-models.

Another approach to keyword spotting is to search output of a speech recognizer. Two types of recognizers have been used: word recognizers and subword (phone or syllable) recognizers. A **large vocabulary continuous speech recognizer** (*LVCSR*) falls into the word recognizers category. Keyword spotting on single string output of LVCSR or subword recognizer is similar to a textual search. The difference is only in the likelihood (or confidence) attached to each word recognized by the LVCSR. Considering the likelihood, keyword spotter can estimate the confidence of the keyword. Keyword spotting based on LVCSR has a critical drawback in dropping keywords which are not in the LVCSR vocabulary, so called **out-of-vocabulary words** – *OOV* words. Phone lattice search overcomes OOV problem, but the accuracy deteriorates. The accuracy deterioration can be reduced by implementation of modeling of phone insertion, substitution and deletion errors [PSPH08]. A mapping of phones to phonetic groups was proposed in [AES01] to overcome the errors in phone string search. Another drawback of subword based keyword spotting is in production of more false alarms compared to LVCSR based keyword spotting.

Important improvement in accuracy of keyword spotting systems was brought by a multi-hypothesis decoder. First multi-hypothesis decoders generated N-best hypothesis. A keyword which was not in the 1-best output of the recognizer can appear in the parallel hypothesis. Another advantage of parallel hypothesis is modeling of the search space which can be used for *confidence estimation*. Keyword spotting utilizing the N-best output of

LVCSR system was proposed by Weintraub [Wei95]. It was based on a log-likelihood ratio approach. The next step to improve the output of recognizer for better keyword spotting and better modeling of the background model is to generate *lattices*. Keyword spotting based on phone lattice search was proposed by James [Jam95].

Keyword spotters based on LVCSR output give actually better accuracy than acoustic keyword spotters. However, the problem of out-of-vocabulary words leads to proposing *system combinations*. A paper written by Saraclar et al. [SS04] deals with different combinations of results obtained with word and subword lattice searching. Word, phone and syllable lattice searcher combinations were proposed in [LMD02]. Experiments in [JFJY96] showed improvement of combined word and phone lattice search system. More references on spoken term detectors based on the output of recognizer and their combination are given below in the spoken document retrieval section.

Another way to increase spoken term detection accuracy is the use of discriminatively trained models or anti-models. These approaches are usually used for on-line keyword spotting applied in dialog systems. For example, Rose in [Ros92] proposed discriminative training (using MMI – Maximum Mutual Information) of keyword models to achieve more precise keyword models. Choice of a good and reliable confidence measure can also significantly improve spoken term detection. This topic is described below in section 2.4.6 dealing with confidence measures.

A fraction of spoken term detection papers also deal with speed-up and optimization of STD. Knill et al. in [KY96] proposed a technique for optimization of acoustic keyword spotting. They used pre-calculation of background model (by Viterbi best path), reducing the number of features, Gaussian selection and clustering. We have also proposed [SSB+05] a speed-up method for off-line acoustic keyword spotting. This approach uses a matrix of monophone posterior probabilities and speed-up was based on masking this matrix by phone lattice. This leads to speed-up of about 30%.

Interesting way to detect keywords is using weighted automata (such as finite state automata and transducers). Lattices can be also seen as automata. Spoken term detections can be found by a composition of the term-automata and lattice-automata. Automata can be also indexed. Allauzen et al. presented a general algorithm for the indexation of weighted automata in [AMS04]. The index is represented by a deterministic transducer which should be optimal for search. The search algorithm should achieve linear time complexity (depending on size of indices). Parlak et al. [PS08] proposed spoken term detection system using indexing and search based on weighted finite state machines.

2.4.2 Spoken document retrieval

As computers have stored information and people have used computers for searching in it, a discipline called *Information Retrieval* (IR) was developed. A specialized branch of the information retrieval is so-called *document retrieval* aiming to search in documents. The answer to given textual query must be found in textual documents and the most relevant documents should be returned. As spoken documents started to be accessible, a branch dealing with spoken documents called *Spoken Document Retrieval* (*SDR*) developed from the document retrieval task.

One of the first spoken document retrieval application was Video Mail Retrieval system presented by Young at Cambridge University [BFJ+96]. It was based on searching spoken e-mail for approximately 20 keywords. Later, it was extended to unlimited search vocabulary. Another application was proposed by Witbrock in [WH97]: spoken document retrieval used to digital video library access using word and phonetic search. An

automatic processing of lectures for information retrieval was presented by Park [PHG05]. Another example is SpeechFind, the SDR system for a National Gallery of the Spoken Word developed by Hansen [HHZ+05].

Because the search in spoken data started to be applicable, NIST decided to include spoken document retrieval to TREC evaluations[3] as a new track. The goal of TREC evaluations is annual evaluation of search systems in large text collections. The TREC-SDR aimed to examine the behavior of information retrieval approaches on erroneous output of speech recognizers. Four TREC-SDR NIST evaluations were organized in years 1997–2000. The TREC-SDR evaluation data was broadcast news split to a set of documents. Participants processed the data by their speech recognizers to produce a textual output. Then the SDR systems had to find relevant documents to given queries in the recognizers output. The TREC-SDR task was declared as solved problem at TREC-9 in the 2000 [VH01]. The accuracies of SDR on recognizers' output were nearly the same as on reference human transcripts.

Several Ph.D. thesis deal with SDR. One of the first implementations of a spoken document retrieval system based on the output of conventional speech recognizer was proposed by James [Jam95]. James used keyword spotting technique based on word lattices with incorporation of language model. Another thesis written by Siegler [Sie99] deals with incorporating continuous speech recognizer uncertainty into information retrieval system. Spoken document retrieval using search in phone strings using probabilistic string matching was described in Wechsler's thesis [Wec98]. Subword based (phones, broad phonemic classes and syllables) approach for spoken document retrieval was investigated by Ng [Ng00b].

Finally, a recent Ph.D. thesis by Dong Wang [Wan10] is aimed at modeling of uncertain pronunciation of OOV terms. These terms are searched in pruned phone lattices. The pronunciation model was based on stochastic joint-multigrams. Next, Dong Wang proposed "term-dependent discriminative decision" which integrates multiple informative decision factors into a classification posterior probability. Finally, a direct posterior confidence based on multi-layer perceptron is addressed to compute the acoustic confidence of an OOV detection.

2.4.3 Do we need out-of-vocabulary handling?

Many people have been trying to overcome the problem of out-of-vocabulary words in LVCSR. The investigated techniques include hybrid word-subword decoders, hybrid language models, OOV detectors, background models or combination of word and subword outputs. Bisani and Ney [BN05a] claimed that an OOV word will never be recognized, but will be substituted by in-vocabulary word. This will also harm neighboring words and they will be also miss-recognized. OOV words are often content words so we loose a lot of information by miss-recognizing them. Also, they claimed that later processing stages (translation, understanding, SDR) cannot recover from OOV errors.

However, several experiments [VH98, GAV00], showed that for example SDR can handle non 100% accurate ASR. The results of the TREC-9 2000 SDR evaluation [GLV00] showed that retrieval performance for sites on their own recognizer transcripts was virtually the same as their performance on the human reference transcripts. Therefore, retrieval of excerpts from broadcast news using automatic speech recognition for transcription was deemed to be a solved problem – even with word error rates of 30%. This result was cited and disputed by Yu and Seide in [YS04b]. They claimed that most TREC-SDR benchmark

[3] http://trec.nist.gov/

systems use well-tuned domain-specific recognizers to generate text transcripts followed by information retrieval. Using ASR systems with low word error rate ($\sim$ 20%) and redundancy in audio and queries, retrieval accuracies similar to using human reference transcripts are achieved. They however conclude that "*But this approach is not suitable for conversational-speech scenarios with unpredictable vocabulary and language domain, and where queries are usually one word long.*".

Summary: Errors caused by out-of-vocabulary words in automatic speech recognition need not be serious problem for certain applications, for example SDR on broadcast news. This is caused by relatively long documents, where a query can be spoken several times. This increases the probability of correct retrieval of the document. SDR targets on relevant **document retrieval**. The user does not need to detect precisely all occurrences of a query. But there is another set of applications, for which errors caused by OOVs are a serious problem. For example, spoken term detection in security domain. Here, all term (queries) have to be correctly detected and a term spoken only once can be more important than a term spoken several times.

2.4.4 How to handle out-of-vocabulary words

Several papers suggesting different solutions of the problem of out-of-vocabulary words are reviewed in this section. A popular solution of searching an OOV is to recognize the whole utterance to subword units (phones, syllables etc.). Queries or terms (including possible OOVs) are then also transcribed to subword units. N-best variants generated from phone confusion matrices can be used to overcome phone recognition errors. The examples of the above mentioned approach are thesis of Wechsler [Wec98] and Ng [Ng00b]. Lee et. al. [LTI05] proposed sub-phoneme units and built an SDR system based on these units. The drawback is that it was tested only on a Japanese broadcast new corpus.

Another approach is a linear combination of word and subword systems used in SDR for retrieval of queries containing OOV words. Techniques published by Ng [Ng00a] expand an OOV word to phone string and then concatenate phones to overlapped phone n-grams. Word transcripts are also converted to phone n-grams. Several SDR systems were built having different lengths of n-gram. After that, standard information retrieval was applied for each system separately and finally, scores of retrieved documents (for each system) were combined. This approach is applicable on SDR and was evaluated on TREC-SDR data but its application on spoken term detection is limited.

Dharanipragada et al. [DFR98] proposed a two pass system for OOV queries for SDR and evaluated it also on TREC-SDR data. They use a phone recognizer for transformation of speech to phone n-grams which are indexed. They find probable places of a query (converted to phones) using indexed phone n-grams in the first pass. A fast keyword spotter is then used for more precise acoustic match between the query and hypothesized place.

In general, the drawbacks of subword-based approach are in increased number of false alarms. The problem is also in the search for several variants per OOV which leads to higher search time. Having another subword index costs also more disk and memory space.

Another set of solutions is based on bypassing the subword recognition. One approach is to use query expansion and stemming as it is used for information retrieval in textual data [RJ97]. Woodland et al. [WJJJ00] use query expansion for solving the OOV problem in SDR. The query expansion uses an additional set of documents from a different source to find relevant words instead of an OOV word. This approach is good for retrieval of relevant documents, but not for search for exact match of a spoken term. Also, this approach fails

in case of OOV proper names and it requires a corpus for query expansion.

Witbrock et al. [WH97] published an approach for search of OOV in the output of a word recognizer. It was done within Informedia project, dealing with a large collection of video and audio material. They convert word lattices to phone lattices by replacing words by their phone representations. These phone lattices were indexed (phone 3–6 grams). Input term was also converted to phone string and searched in the index. This approach is evaluated in this thesis in section 5.2.

Logan et al. [LT02] proposed a method which is based on substitution of out-of-vocabulary words by in-vocabulary words. The OOV word or phrase is transcribed to one phone string using a pronunciation dictionary or automatic transcriber. A lattice or N-best list of expanded query is generated by a modified Viterbi algorithm, which maps the phone string of original query to a sequence of words from LVCSR dictionary according to "acoustic" and language score. The acoustic score is a score from confusion matrix which quantifies the acoustic mismatch between two phones. Then the expanded queries are searched in word lattices generated by the LVCSR system. This approach was compared to the one published by Witbrock et al. [WH97] with just slightly worse results. Finally, both these approaches were linearly combined (addition of scores) with overall better results. Logan et al. also concluded that their approach is quite sensitive to grapheme-to-phoneme conversion errors. Witbrock's approach is more robust because there is high chance that a part of the phone string is correctly transcribed.

Saraclar et al. [SS04] proposed a method for spoken utterance retrieval (on the boundary of spoken document retrieval and spoken term detection). They claimed that indexing of lattices brings better retrieval results than indexing textual output, while textual output can be used when ASR output is mostly correct and documents are long enough. They claimed that 1-best output can be used for ASR error rates lower than 20%. Multiple hypothesis are required for higher error rates. Experiments were run on broadcast news (HUB4) and conversational telephone speech (Switchboard) corpora. They show lattice indexing superiority over 1-best output of ASR at first. Better retrieval results were obtained on lattices obtained by conversion of word lattices to phone lattices (similar to Witbrock [WH97]) rather than on phone lattices generated by a phone recognizer. Three experiments were done in case of word/phone combination:

1. Score combination of word and phone results (searched in parallel) gave the worst accuracy.

2. Search in vocabulary cascade. In-vocabulary queries are searched in words, out-of-vocabulary queries are searched in phones. This approach gave intermediate accuracy.

3. Search cascade: The query is searched in words and if no results are found than the query is searched in subwords. The results of this search cascade was found as the most accurate.

The authors have also shown that lattice based search is much better than 1-best search. The word and phone combination gave another improvement to word lattice based search. Saraclar concludes that the spoken utterance retrieval for broadcast news has different characteristics than spoken utterance retrieval for conversational speech.

Bisani and Ney [BN05a] proposed a different way of OOV handling. They built a hybrid (word-subword recognizer). The dictionary of standard LVCSR system was expanded by a set of subword units (phone multigrams). Then, several tests were done on Wall Street Journal with good results. The best results were obtained with multigram strings 1 to 4

characters long. The drawback of this method is that it leaves undetermined where word boundaries should be placed.

Another approaches to build hybrid recognizer were investigated by Yu and Seide in [YS04b]. The first approach was so-called posterior combination which means combination of output posterior probabilities of two systems (phone and word). The second and third approaches were so-called prior combination. Two lattices were created and then combined on word level (the second approach) or utterance level (the third approach). On LDC Voicemail corpus, they have shown that prior combination on utterance level does not work due to different likelihoods between word and phone lattices. On the other hand, they obtained good improvement from word-level prior combination (word-phone lattices).

Summary: OOV handling in spoken document retrieval and spoken term detection can be classified to four categories.

- Spoken data are processed by two stand-alone recognizers, word and subword. Out-of-vocabulary words are transcribed to subword units and searched in subword indices. The drawbacks are longer processing (two recognizers), high memory requirements, longer search times and higher number of false alarms.

- Subword transcriptions of spoken documents are generated from word recognizer output. Out-of-vocabulary words are transcribed to subword units and searched in the subword indices. The advantages are faster processing (only one recognizer), and lower number of false alarms. On the other hand, the accuracy can be negatively affected by the word-to-subword conversion.

- Out-of-vocabulary transformation to in-vocabulary. This method is not so widely used. The OOVs are converted to strings of in-vocabulary words which are likely to be recognized instead of OOVs. The drawback is higher number of false alarms.

- Hybrid word-subword recognizer. The recognizer generates word transcripts and only on parts of speech containing an OOV it switches to subword units. This should be the optimal way. The following section deals with this approach in more depth.

2.4.5 Hybrid language model

The solution of OOV problem can be viewed from a different point. Several researchers have tried to design a universal word-phone recognizer. This was applied for robust speech recognition or in a dialog-system to overcome failures caused by OOV.

Bazzi et al. [BG00] proposed a hybrid language model consisting of two branches. One branch was standard word language model and the second branch was a phone loop (with n-gram phone model) connected with the word model in parallel. The results were 3% relative deterioration of WER and 1.3% false alarms of OOV while they correctly detected 47% OOVs. The transitions between word and subword parts are controlled by an OOV penalty. They stated that there are three problems associated with OOV words:

1. Detection of presence of OOV in recognized utterance. Task called “detection of out-of-vocabulary words” aims at this problem.

2. Accurate recognition of the underlaying sequence of subword units.

3. Conversion of subword unit sequence to the actual word, so that it can be understood semantically.

The authors pointed out, that for domain-dependent tasks (telephone weather forecast system), syllables were better than phones.

Later, Bazzi [Baz02] wrote the thesis aimed at hybrid word-subword recognition. The main point he investigated was the *out-of-vocabulary detection task*. Its impacts on word recognition and word confidence score were studied. He used hybrid word-subword recognition network where the word language model contained a symbol substituting OOV words. The OOV words were modeled by phone sequences (bigram phone LM). He also proposed modeling of OOV words by variable-length phone sequences (multigrams), which brought significant improvement of OOV detection accuracy.

Yazgan et al. [YS04a] (inspired by [Gal03]) proposed a different word-subword hybrid language model for LVCSR. The subword units of this language model are phones or syllables. The language model is built in the following way: A dictionary sorted by counts is created on a language model data. First X most frequent words are kept in the dictionary. Words below X are transcribed to subwords. Then the language model is built. Subword "pronunciations" are added to the recognizer's dictionary. According to the authors, this approach has several advantages. There is no need for special insertion penalties. Also, "word to subword" and "subword to subword" prior probabilities are trained on rare words which is more natural. Experiments are evaluated on Switchboard database. Each sequence of phones longer than 3, without corresponding form in pronunciation dictionary of LVCSR, is denoted as OOV. Recognizer outputs are both 1-best and lattice. The conclusion is that the word-phone model is always better and has very low impact on word error rate (in several cases, it has no impact). The authors also obtained about 15% improvement of OOV detection.

One of the state-of-the-art approaches to handling the OOV terms using hybrid word-subword recognizer was proposed by Akbacak et al. [AVS08]. They used similar principle as Yazgan [YS04a]. The difference is in used subword units and in the way of OOV term detection. They used so-called graphones as subword units. A graphone [BN05b] is a pair of letter sequence and a phone sequence of possibly different lengths. Graphone units are trained on pronunciation dictionary. Information about word boundaries is incorporated into the graphones. This allows for easy reconstruction of word labels from graphones. OOV words in corpora for language model training are substituted by corresponding graphone sequences and the hybrid language model is trained. A hybrid index is created after the recognition. The hybrid index is then converted to regular word index by post-processing step that joins graphones back into words. No OOV word handling is needed in the search phase. The OOV word should appear in the post-processed word index.

Summary: Hybrid word-subword recognizer is a very promising approach for the solution of the OOV problem. Hybrid recognizer can produce hybrid lattice in one decoding step. The in-vocabulary speech should be transcribed by words and only OOV parts should be transcribed by subword units. Both should save time and space. Also, the word and subword scores should be comparable. We chose the hybrid word-subword recognition as our primary approach to open-vocabulary spoken term detection. Also, graphone units seem to be promising.

2.4.6 Confidence measures

Lot of progress has been done also in the field of confidence measures. The goal of confidence measure is to estimate the reliability of recognition result. Speech can be noisy and/or non-keyword utterances may appear in the speech. A recognizer must be able to spot keywords with other speech or sounds in background. Also, speech that does not

include any of the valid keywords, should be rejected. Likelihoods (state, phone, word, etc.) generated by a standard recognizer depend on speaker, channel, environment, etc. A confidence measure should suppress this dependency. The original purpose of confidence measure is to reject miss-recognized words or utterances for dialog or speech understanding systems [PSG98].

Detection of parts of utterance where OOVs are spoken is another application of confidence measures. Rejection of miss-recognized words or out-of-grammar sentences is called *Utterance Verification*. The confidence measure is usually based on *hypothesis testing* theory or on *posterior probability*. Utterance verification is often used as post-processing stage of recognition. Several papers also tried to incorporate confidence measures (utterance verification) directly in the decoder.

Hypothesis testing approach usually uses discriminatively trained anti-models which should compete with the original models. The alternative hypothesis can also be provided by N-best algorithm [TGT01]. Simple solution is a garbage model (background model) or an anti-model. "On-line" garbage model or anti-model can be used for applications with low computational resources.

A paper surveying confidence measures was written by Jiang [Jia05]. Confidence measures can be used to automatically label individual hypothesized words in the output of ASR system as either *correct* or *incorrect*. Algorithms for estimation of confidence measure can be classified into three major categories [CCW06]:

Feature-based: These approaches assess the confidence on some selected features, such as language model probability, acoustic score, word duration, number of phones, etc. [YZS06, SL96].

Explicit model-based: These approaches treat confidence measures as hypothesis testing problem and need to model extra alternative hypotheses. These techniques usually use background models or anti-models [KL99, MJ99, TGT01, JRH97, KLC02, RCAC96, SL96].

Posterior probability-based: The posterior probability estimated according to the standard Maximum a Posteriori (MAP) framework is a good candidate [Jia05] for confidence measures. It has a well bounded range between 0 and 1 and strong background model. Superior performance of the posterior probability has been demonstrated by using it as the confidence measure in [WSMN01, CCW06, BFHC03, KVBB06].

Posterior normalization using Maximum Entropy Criterion was proposed by Yu et al. in [YZS06]. The methods were evaluated on small conversational telephone speech and interview corpora (1h each). They proposed a feature-based function mapping raw posterior probability and keyword length to the confidence. The entropy of mapping function is maximized. Keywords are split to several classes (according to number of phones per keyword) and then several parameters of mapping function are trained. But comparison of different feature functions is missing in the paper. The authors also mentioned that posterior pruning of lattices can have small impact on word error rate but large impact on confidence because pruning can lead to unnormalized posteriors.

Wessel et al. [WMN99, WSMN01] proposed 3 methods to estimate confidence measure: sum over all arcs overlapping with a term, sum over all arcs overlapping with median time of a term and maximum of sum of arcs overlapping with given frame within a term (found to be the best). Confidence measures proposed by Wessel were verified by Chen et al. [CCW06] but all three approaches gave nearly the same results (evaluated on

Mandarin broadcast news corpus). Chen extended Wessel's confidence measures by the entropy information and obtained another improvement. Wessel [WMN99] also verified, that confidence measure based on word posterior probability in lattice is more accurate than N-best and alternative confidence measures as acoustic stability and hypothesis testing. The evaluation was done on CTS corpora.

Koo and Lee [KL99] proposed three different approaches for word confidence measure based on phone confidence. They used arithmetic mean, geometric mean and harmonic mean of phone confidences. The geometric mean gives the best results (tested on Korean telephone corpora including stock-exchange information). Also, they investigated phone confidence given by log-likelihood ratio of phone model and anti-model (based on model trained on given phone cohort) scaled by sigmoid.

Later, Benayed et al. [BFHC03] proposed similar confidence measure of a word based on means of phone confidences. They used also arithmetic, geometric and harmonic means plus they tried to normalize phone confidence by its length. Finally, all three means were combined using a support vector machine (SVM) to get the final confidence. Their results showed that geometric mean was the worst. The arithmetic mean was the best for time non-normalized phone confidence and harmonic mean for time normalized phone confidence. Benayed's systems were evaluated on French SpeechDat.

Moreau et al. [MJ99] proposed frame-level anti-models training on a corpus corresponding to a specific type of error (substitution, out-of-vocabulary and noise). They found out that the best anti-model is trained only on out-of-vocabulary and noise data. An anti-model trained on the "substitution error" data achieved only small improvement of the substitution error but the accuracy degraded significantly on the "out-of-vocabulary" and "noise error" data. The experiments were evaluated on French telephone corpus of surnames.

Junkawitch et al. [JRH97] proposed a confidence measure for acoustic keyword spotting which can be incorporated directly to the decoder. They proposed two methods. The first is based on normalization of frame emitting state likelihood by sum of the likelihood of all states in HMM. This is equal to posterior probability of the HMM state. The second one does not take into account the emitting state likelihood for the normalization. This is comparable to the hypothesis testing (i.e. likelihood ratio) of the HMM state. They also proposed new decoding algorithm based on Viterbi decoding. This algorithm takes into account the length of keyword and does the time normalization directly during the decoding. The conclusion is that time-normalized decoding is better. Using the time-normalized decoding, Junkawitch et al. found the second confidence measure (the likelihood ratio) better. On the other hand, the difference of accuracies was not significant (tested on German SpeechDat corpus). This is interesting, because the posterior probability based confidence is expected to be better.

Tan et al. [TGT01] proposed a confidence measure based on N-best list for fast on-line recognizers. Their approach is based on switching between on-line garbage model and second-best likelihood ratio depending on N-best homogeneity measure.

Kim et al. [KLC02] claimed that the dynamic of a confidence measure depends on the keyword. That is why certain keywords can be systematically rejected for a certain confidence threshold. They proposed a hybrid confidence measure for domain-specific keyword spotting. The phone confidence is defined as likelihood of the recognized phone minus likelihood of phone anti-model, normalized by phone length. The word confidence is a non-trivial combination (similar to geometric mean) of phone confidences. They proposed new normalized confidence measure based on normalization of phone confidence to unit variance and mean. This is motivated by the fact that each phone has different confidence

statistics. Their hybrid confidence measure is a linear combination of normalized confidence and of anti-model based on background model. They evaluated on 10h of 10 male speakers and concluded that hybrid confidence measure was the best. The normalized confidence was significantly better than the baseline (non-normalized) confidence.

Ketabdar et al. [KVBB06] proposed confidence measure based on summing up the deterministic decisions (1 or 0). The decision is made on probability of being in a given keyword at a certain time t knowing to be in a certain state at the same time t. The information of being in the state is based on state posterior probability. The method was evaluated on telephone corpus of connected digits.

Garcia-Mateo et al. [GMRO99] proposed several combinations of posterior based confidence measures and hypothesis testing based confidence measures for isolated proper name recognition in telephone data. The hypothesis testing is based on standard phone model/anti-model principle. Ratio between the best word occurrence (of overlapped occurrences) and sum of all other words overlapped with this word is taken as the posterior probability-based confidence. Authors proposed three combination techniques: addition, multiplication and a perceptron. The best one of the single confidences is the posterior probability based confidence measure, and the best combinations are the perceptron and the addition (the perceptron is slightly better).

Summary: Many papers [KLJ01, RLJ97, KLJ98, RLJ97, SL96, SSJ96, RCAC96] deal with confidence measures using hypothesis testing (likelihood ratio). The majority of these papers deal with different types of anti-models. Hypothesis testing approaches were widely applied on-line with the usual application being a telephone dialog system. This kind of applications used small vocabulary and required on-line processing. Paper [KLJ01] concluded, that anti-models do not play a strong role in rejecting unlikely hypothesis during decoding as the number of vocabulary word is increased, which is our case in this thesis.

Several papers point out superiority of posterior probability over hypothesis testing based confidences. This is supported by a possibility to generate more output hypothesis: the more hypothesis, the better estimation of background normalization for posterior probability. That is why lattices outperform N-best lists. The spoken term detection task is an off-line process in our case, so posterior probabilities based on lattices are suitable confidence measure. We want to separate the process of recognition (lattice generation) and spoken term detection (confidence estimation and word-subword combination).

Discriminative training techniques were also found beneficial. In case of lattice generation, the discriminative training can be incorporated directly to the acoustic models instead of the anti-model.

Several papers deal with correct estimation of posterior probability of a group of overlapped words. This should be taken into account for STD. A part of this thesis deals with keyword confidence measure estimation. The idea of using mean and variance normalization of confidence was found interesting and is evaluated in this thesis.

2.5 Application areas

The results and conclusion of this thesis should also have practical impact. Our research group participates in several large projects and actively cooperates with several business partners that are interested in the conclusions of this thesis.

One important application area of this thesis is in research. The topic of this thesis was partly initiated by the NIST Spoken Term Detection evaluations in 2006. We have planned to evaluate proposed approaches in the following STD evaluation. But unfortunately, the

next evaluation has been postponed several times. The outcome of the thesis was also used in DIRAC project[4] as an OOV detector.

We have shown that hybrid system can be easily used in existing indexing and search systems for spoken documents. Only a module which will convert out-of-vocabulary terms to multigram representation is needed in the searcher.

The demand for spoken term detection applications is growing. The results are usable in commercial domain (call-centers or operators).

The security domain can profit from proposed subword or hybrid systems, too. Errors caused by OOVs are serious problem in security area. Here, all term (queries) have to be correctly detected and a term spoken only once can be more important than a term spoken several times. That is why accurate and "open vocabulary" spoken term detector is needed. We successfully used proposed hybrid system in our project[5] aimed to spoken term detection for security. The problem caused by OOVs is even more important for inflectional languages (like Slavic languages) due to large size of vocabulary (milions of words).

A way how to easily add new words to the LVCSR will be shown in the experiment with HybridOracle system (section 7.3). A minimalistic LVCSR-based spoken term detector can be built on small LVCSR (5k words for example). The searched terms which are OOVs can be added in system start-up into the recognition network in the same way as is in the HybridOracle. This can not be used in document indexing and search but it can be an interesting alternative to the acoustic keyword spotting, where lots of data is searched once or few times and the processing time must be fast.

An experimental tool (section 3.5.5) was implemented for searching given sequences in lattices and estimation of its confidences. This software is freely available for research purposes[6].

[4] `http://www.diracproject.org`

[5] Project VD20072010B16 of Czech Ministry of Interior.

[6] `http://speech.fit.vutbr.cz/en/software/lattice-spoken-term-detection-toolkit-latticestd`

Chapter 3

Evaluation

Clear definition of evaluation setup is important to correctly compare different systems and to make conclusions. This chapter aims at the description of evaluation *data* and *metrics* used in this thesis.

3.1 Evaluation data

Well defined evaluation data is important for objective evaluation and comparison of different systems. Unfortunately, each published spoken term detection system was evaluated on different data and with different term set. There were 4 TREC NIST evaluations in years 1997–2000. The TREC evaluations had only partial overlap with spoken term detection task. The goal of TREC was *Spoken Document Retrieval* (TREC-SDR) on broadcast news. The broadcast news recordings were recognized by *Automatic Speech Recognizer* and then processed by *Document Retrieval* system. The goal was to find the relevant document, but not to find all term occurrences. The TREC-SDR was declared a solved problem at TREC-9 in 2000. There were no spoken term detection evaluations organized by NIST from year 2000 till 2006. New evaluation track was announced by NIST in 2006. It was called *Spoken Term Detection* (STD) evaluation[1].

The goal of the first NIST STD (2006) evaluation was to explore promising new ideas in spoken term detection and measuring the performance of this technology [FAD06]. The spoken term detection system should consist of two parts:

- The first part is an *indexing sub-system*. It processes all input speech data (audio signal) into indices. This step can take longer time (hours of processing time per hour of speech data) and is run on the data only once.
- The second part is a *search sub-system*. It should find a given term as fast as possible (milliseconds of processing time per one term) in the indices.

The results and experiences were discussed at post-evaluation workshop. The NIST organizers stated that this was a pilot evaluation for overview of available techniques. The conclusion of NIST STD 2006 workshop was the opinion that nowadays spoken term detection systems are relatively accurate but still too slow and resources consuming. They are still inapplicable for large-scale data (more than 10^6h of speech).

The STD 2006 evaluation task was to find all of the occurrences of a specified term in a given corpus of speech data. The "term" is a sequence of one, two, three or four

[1] `http://www.itl.nist.gov/iad/mig/tests/std/2006/index.html`

words. The words in a term have to be said by the same speaker, channel and file. The gap between adjacent words must not be longer than 0.5s. Terms are specified only by orthographic representation so "wind" (moving air) will match "wind" (twist) but "cat" will not match "catalog". The evaluations ran for 3 different domains and 3 languages, see table 3.1.

Domain \ Language	English	Arabic	Mandarin
Broadcast News (BCN)	$\sim$ 3 hours	$\sim$ 1 hour	$\sim$ 1 hour
Telephone Conversations (CTS)	$\sim$ 3 hours	$\sim$ 1 hour	$\sim$ 1 hour
Round-table Meetings (MTG)	$\sim$ 2 hours	No	No

Table 3.1: Durations of indexed audio for both, the DevSet and the EvalSet.

NIST provides three data sets. A *Development set* (**DevSet**), a *Dry Run set* (**DryRunSet**) and an *Evaluation set* (**EvalSet**). The *DevSet* was offered for system development. It contains speech data, reference transcripts and a list of 1099 terms. The *DryRunSet* differs from DevSet only in different term list (1099 terms). The dry run was just for evaluation of participant competence to use NIST scoring tools and to generate correct result files.

The *EvalSet* contains different speech data and a different term list (1099 terms). Unfortunately, NIST decided not to publish reference transcriptions. The EvalSet will be reused for next evaluations due to lack of speech data. This complicates evaluation of STD systems, because there is only the DevSet.

Using *round-table meeting data* (MTG) and *conversational telephone speech* (CTS) brings more objectivity, because it is more natural form of speech (in comparison to *broadcast news data* (BCN)). Meeting or telephone dialog participants speak informally and the speech is spontaneous containing lots of hesitations, crosstalk, smacks and background noises. This data is closer to the security domain.

With regards to our participation in European projects M4[2], AMI[3] an its follow-up AMIDA[4], we are experienced with automatic processing of conversational speech (both meeting and telephone). These projects concentrate on processing of round table meetings. However, in this thesis, we decided to use the CTS data because the MTG data consisted of several different subsets, recorded at different sites. This could negatively influence the evaluation by a hidden error caused by inaccurate adaptation of the recognizer.

The CTS data of NIST STD 2006 DevSet is used in this thesis for STD evaluation. As because the speech recognizer (chapter 4) is taken as a "black box" and NIST released only the *DevSet*, several system coefficients are tuned on the DevSet: unit (word, phone or multigram) insertion penalty and language model or acoustic model scaling factors. We assume that tuning of these parameters has no impact on the correctness of the results and conclusions.

3.1.1 Term set modification and vocabulary reduction

The original term set for English part of NIST STD 2006 evaluations is not representative for our experiments, because it contains low number of out-of-vocabulary (OOV) words.

[2] `http://www.dcs.shef.ac.uk/spandh/projects/m4/index.html`
[3] `http://www.amiproject.org/`
[4] `http://www.amiproject.org/ami-scientific-portal`

We decided to make several changes to the STD term list and our speech recognizer vocabulary to achieve higher OOV rate. Distribution of term occurrences among data sets and numbers of in-vocabulary and out-of-vocabulary terms with the original recognizer's vocabulary are summarized in table 3.1.1. The *in-vocabulary* (IV) term contains all words which are in the speech recognizer vocabulary. The *out-of-vocabulary* (OOV) term contains at least one word which is not in the speech recognizer vocabulary.

Term			CTS	
length [words]	count		terms occur.	
	IV	OOV	IV	OOV
1	555	44	4800	18
2	346	54	126	8
3	66	19	10	2
4	9	6	1	0
sum	976	123	4937	28

Table 3.2: Distribution of terms for `full LVCSR 50k vocabulary`. The second and third columns give the numbers of IV or OOV terms in the term list. The last two columns represent the numbers of occurrences of IV and OOV terms in CTS set.

First of all, all terms containing true OOV words or 1 phone long[5] were omitted. The 1 phone long term is not a big problem for word-based STD, but serious problem for phone based STD (huge number of detections).

Then a set of "artificial" OOV words is defined – these are originally in the recognition vocabulary, but deleted for future experiments to create more OOVs. Their selection is done in the following way: Word counts are collected over the *DevSet*. Based on these counts, a suitable set of OOVs was selected: The word had to have several occurrences, but generally less than 10. Only 5 OOVs have more than 10 occurrences. In total, 880 words were deleted in this way, of which 440 do appear in NIST dev-set transcriptions. Another 440 words which do not appear in the transcriptions were simply selected from the LVCSR vocabulary. They are of no use in this these, but reserved for future work.

A limited LVCSR system was created (denoted by **WRDRED** which means "**reduced vocabulary**") where these 880 words were omitted from the vocabulary. This system has reasonably high OOV rate on the NIST STD06 DevSet. The term set has 975 terms of which 481 are in-vocabulary (IV) terms and 494 are out-of-vocabulary OOV terms (terms containing at least one OOV) for the reduced system. The number of occurrences is 4737 and 196 for IV and OOV terms respectively. We can detect all the "artificial" OOV terms by the original **full vocabulary** LVCSR (denoted as **WRD**) and evaluate the "oracle" OOV term detection accuracy.

Reference transcription of the NIST STD 2006 DevSet has 32002 tokens. Defined "artificial" OOVs appear 799 times in the corpus. So the OOV rate is 2.5%, which is close to real tasks.

Table 3.1.1 summarizes the numbers of terms and term occurrences for different term length and data types in DevSet.

It holds in this thesis, that all systems with reduced vocabulary (denoted with suffix *RED*) are derived from the corresponding full dictionary systems. All parameters are the

[5] term "A."

Term			CTS			
length [words]	count		count		terms occur.	
	IV	OOV	IV	OOV	IV	OOV
1	309	245	214	76	4640	156
2	149	197	42	30	92	34
3	21	45	5	5	5	5
4	2	7	0	1	0	1
sum	481	494	261	112	4737	196

Table 3.3: Distribution of terms for `reduced LVCSR 50k vocabulary` – **WRDRED** system. The second and third columns give the numbers of IV or OOV terms in the term list. The next two columns summarize the numbers of the terms appearing in the CTS. The last two columns represent the numbers of occurrences of IV and OOV terms in CTS set. The true OOV terms and 1 phone long terms are omitted.

same (pruning, penalties, etc.), only the vocabulary is reduced, unless it is mentioned otherwise.

3.2 Recognition evaluation metrics

Standard metrics are used for evaluation of underlying recognizers. Two different kinds of outputs and two different kinds of units are evaluated. The outputs can be either 1-*best* (string) or *lattice*. The units can be either *words* or *phones*. The 1-best string output is represented in the lattice – it is the best path through the lattice.

Recognized and reference strings are compared for evaluation of recognition accuracy. The two strings are matched using dynamic programming. The goal is to minimize the error between these two strings. Several cases can happen:

Match M : Recognized unit is the same as the reference unit. $Cost = 0$

Substitution S : Recognized unit is different from the reference unit. $Cost = 1$

Insertion I : A unit is recognized but it cannot be compared to any of reference units. This happens in case of more recognized units then reference ones. $Cost = 1$

Deletion D : There is a reference unit but no unit is recognized. This happens in case of more reference units then recognized units. $Cost = 1$

The dynamic programming must minimize the overall cost of recognized/reference strings alignment. The number of units in the reference string is denoted N. Having the numbers of M, S, I, D and N, the **accuracy** Acc and the **error rate** ER can be calculated as:

$$Acc = \frac{M - I}{N} \tag{3.1}$$

$$ER = 1 - Acc \tag{3.2}$$

If the units are words, we are speaking about the **word error rate** – WER or **word accuracy** – WAC, if the units are phones we are speaking about the **phone error rate** – PER or **phone accuracy** – PAC.

WER/WAC and PER/PAC are metrics for 1-best string level comparison of recognizers. **Lattice error rate** – LER or so-called **Oracle error rate** can be defined on lattice output. Lattice contains lots of hypothesis and some of them are more or less accurate. An error rate can be assigned to each path in the lattice. Lattice error rate is the error rate of the closest path to the reference string. The difference between LER and WER tells us, how much of the information is hidden in parallel hypothesis against the 1-best output. We should be able to detect 100% occurrences of terms in case of lattice error rate $LER = 0\%$.

The lattice error rate variants of WER/WAC and PER/PAC are denoted as *WLER* or *WLAC* and *PLER* or *PLAC* in this thesis.

3.3 Spoken Term Detection evaluation metrics

This section presents evaluation metrics which are used for spoken term detection and keyword spotting task. Each detected term has a confidence attached. The confidence is a continuous value quantifying, how sure the spoken term detector is about the detection of the term. Some users of spoken term detection application expect hard *YES/NO* decision whether a term is present or not. Another users expect only *YES* decision (rising of an alarm). *NO* decision is the complement to *YES* decision over input speech data. Confidence thresholding is used mapping of confidence to hard binary YES/NO decision. Let us assume that the term confidence is based on term posterior probability. The higher confidence value the higher probability of correct term detection. Let us set the threshold thr to a certain value. The term confidence $c(term_t)$ thresholding is defined by:

$$Decision(c(term_t), thr) = \begin{cases} YES, & c(term_t) > thr \\ NO, & c(term_t) <= thr \end{cases} \tag{3.3}$$

where *Decision* function returns the hard decision whether the term is found or not. Several cases can occur in comparison of detected terms against reference detections (transcription):

1. The decision is *YES* (alarm is raised) and there is a reference term *overlapped* with the detected term in time (figure 3.1a). This case is denoted as **HIT**. We want to maximize the number of hits.

2. The decision is *YES* (alarm is raised) and there is no reference term *overlapped* with the detected term in time (figure 3.1b). This case is denoted as **false alarm** – **FA**. We want to minimize the number of false alarms.

3. There is a reference term in utterance but no *overlapped* term is detected at that place (figure 3.1c) or *overlapped* detected term is marked by *NO* decision (no alarm is raised at the same time). This case is denoted as a **false rejection** – **FR** or a **MISS**. We need also to minimize the number of false rejections.

The definition of "*overlapped*" for reference and detected term varies for different evaluation metrics. It is defined as:

- The mid-point of reference term is between start and end times of detected term for the **figure-of-merit** – **FOM** metric (section 3.3.1) calculated by HTK (section 3.5.1). Also, if several detections overlap with one reference, all of them are considered as HIT in HTK implementation (figure 3.1a).

- The mid-point of detected term is less than or equal to 0.5s from the time span of reference term for **term weighted value – TWV** metric [FAD06] (section 3.3.3) used in NIST STD 2006 evaluations. If more detections overlap with one reference, only one is considered as HIT and the others are considered as FAs (figure 3.2).

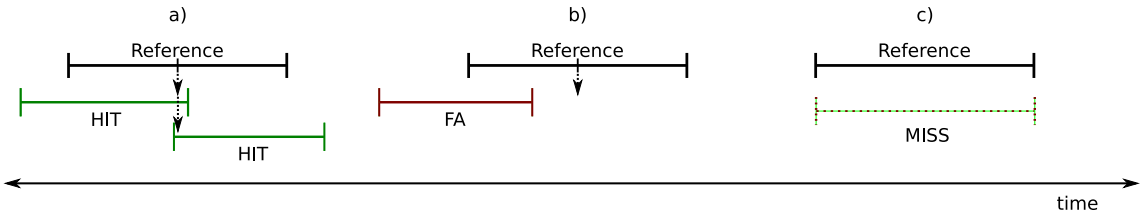


Figure 3.1: Examples of HIT, FA and MISS. Overlap of HIT and reference is implemented in HTK. If two detections overlap with one reference, both are considered as HITs.

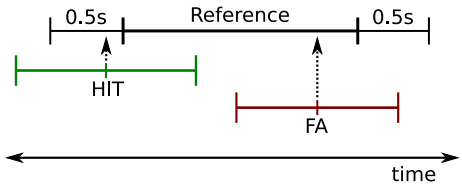


Figure 3.2: Example of HIT and reference overlap defined by NIST for STD evaluation and TWV metric. If two detections overlap one reference, only one is considered as HIT and the other is considered as FA.

The level of threshold can be set for each term. More HITs (and also more FAs) and less FRs are received by lowering the threshold, less HITs (and also less FAs) and more FRs are received by increasing the threshold. The numbers of HITs and FAs are correlated and as number of HITs rises so does the number of FAs. The user must set the threshold to obtain the desired system behavior (high number of HITs or low number of FAs). The accuracy of a term detection system rises as rises the separability of HITs and FAs. A system will have 100% of HITs and 0% of FAs for a certain threshold in an ideal case of the best accuracy. Setting of optimal threshold is nontrivial especially if one global threshold applied over a large set of terms.

The probability of correct detections p_{HIT}, false rejections p_{MISS} and incorrect detections p_{FA} can be calculated by the following formulas. Let us denote:

- $term$ searched term
- thr set threshold
- $N_{target}(term)$ the number of all correct occurrences of $term$ in the data set
- $N_{HIT}(term, thr)$ the number of detections having $Decision(c(term), thr) = YES$ which are classified as HIT
- $N_{nontarget}(term)$ the number of all non-occurrences of $term$ in the data set
- $N_{FA}(term, thr)$ the number of detections having $Decision(c(term), thr) = YES$ which are classified as FA

The $N_{nontarget}(term)$ means all places, where false alarms of the *term* can occur. This value is used by *DET curve* and *TWV metric* and it is discussed in section 3.3.3.

The probability of HIT is defined as:

$$p_{HIT}(term, thr) = \frac{N_{HIT}(term, thr)}{N_{target}(term)} \tag{3.4}$$

The probability of MISS is defined as:

$$p_{MISS}(term, thr) = 1 - p_{HIT}(term, thr) = 1 - \frac{N_{HIT}(term, thr)}{N_{target}(term)} \tag{3.5}$$

The probability of False Alarm is defined as:

$$p_{FA}(term, thr) = \frac{N_{FA}(term, thr)}{N_{nontarget}(term)} \tag{3.6}$$

The performance of spoken term detection system is defined by the trade-off between p_{HIT} and p_{FA}. As this is not a scalar value, different systems can not be easily compared according to p_{HIT} and p_{FA}. That is why several metrics have been proposed for calculation of one scalar value from p_{HIT} and p_{FA}. Some of them are used for comparison of detectors in this thesis. Their brief description and definition follows in sections below.

3.3.1 Receiver operating curve and figure-of-merit

A **receiver operating curve** (*ROC*) for one term is term's $p_{HIT}(term, thr)$ as function of $N_{FA}(term, thr)$ per hour. An example of ROC is shown in figure 3.3.

Spoken term detection evaluation metric called **Figure-of-Merit** (*FOM*) was defined by NIST [NIS91]. The following definition is adopted from HTKBook [You99]. Figure-of-Merit is an upper-bound estimate on spoken term detection accuracy averaged over 1 to 10 false alarms per hour. The FOM estimation assumes that the total duration of the test speech is T hours. For each term, all detections are sorted by confidence. The hits probability $p'_{HIT}(term, i)$ of term *term* found before the i'th false alarm are calculated for $i = 1 \ \ldots \ N + 1$ where N is the first integer $\geq \ 10T - 0.5$. The figure of merit is then defined as

$$\begin{aligned} FOM(term) \quad = \quad & \frac{1}{10T}(p'_{HIT}(term, 1) + p'_{HIT}(term, 2) + \ldots + \\ & p'_{HIT}(term, N) + ap'_{HIT}(term, N + 1)) \times 100\% \end{aligned} \tag{3.7}$$

where $a = 10T - N$ is a factor that interpolates to 10 false alarms per hour. The overall *FOM* is computed as the average of $FOM(term)$ over all terms from a term-list. The *FOM* can be understand as area under ROC from 0 to 10 false alarms per hour (see figure 3.3).

However, the FOM metric is "artificial" and has several drawbacks. It considers only a part of threshold domain (to 10 FAs per hour). The overall FOM is estimated as average of single term's FOM. That is why each term is thresholded separately. One global threshold is usually needed for end-user applications. Also, FOM is sensitive to the term-list selection [SV05].

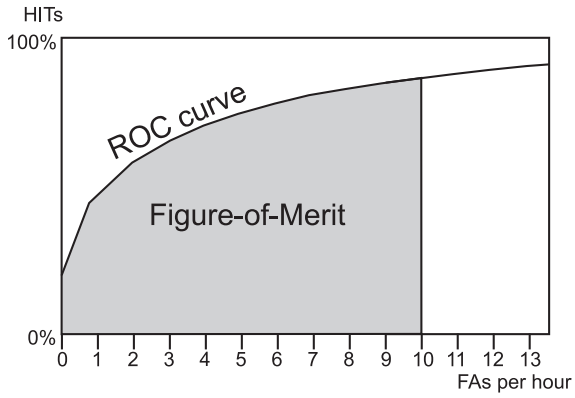


Figure 3.3: An example of Receiver Operating Curve (ROC): Dependency of successfully detected terms on number of false alarms per hour. The gray area represents Figure-of-Merit.

3.3.2 Detection error trade-off curve – DET

Detection error trade-off (**DET**) curve is dependency of term miss probability $p_{MISS}(term, thr)$ (y-axis) and false alarm probability $p_{FA}(term, thr)$ (x-axis) on threshold thr. DET curve is also widely used in other evaluations such as Speaker Recognition (SRE) or Language Recognition (LRE). The DET curve is a plot for all possible values of the threshold (on contrary to ROC). The definition of DET used in this thesis is adopted from NIST STD 2006 evaluations [FAD06]. DET is defined as $p_{MISS}(thr)$ (y-axis) as function of $p_{FA}(thr)$ (x-axis) with thr as parameter:

$$p_{MISS}(thr) = \underset{term}{average}\{p_{MISS}(term, thr)\} \tag{3.8}$$

$$p_{FA}(thr) = \underset{term}{average}\{p_{FA}(term, thr)\}. \tag{3.9}$$

The number of non-target occurrences of $term$ is defined as:

$$N_{nontarget}(term) = T_{speech} - N_{target}(term) \tag{3.10}$$

where T_{speech} is the length of speech test data in seconds. This definition of $N_{nontarget}(term)$ assumes that a false alarm can occur every second.

The p_{MISS} and p_{FA} are evaluated for each term which has at least one occurrence in test data ($p_{MISS}(term, thr)$ is defined). As DET curve is not a scalar value, two systems can be compared using DETs only by their relative position, shape and slope. An example of DET is in figure 3.4. Every point of DET corresponds to a particular operating point of the detection system. The better DET, the closer to bottom left corner. Unlike the *speaker recognition* or *language recognition* DET curves (figure 3.5), the STD DET curve does not lead to the very bottom right corner of the graph. The right end of DET means the lowest level of threshold. The system cannot detect more terms than it does at this point.

3.3.3 Term-weighted value – TWV

The drawback of DET curve is the impossibility of comparison of two systems by one number. That is why NIST defined two scalar metrics for comparison of spoken term detection systems: *Occurrence-Weighted Value* and the *Term-Weighted Value*. The **term-weighted value** (TWV) is used further in this thesis, because it was selected as the

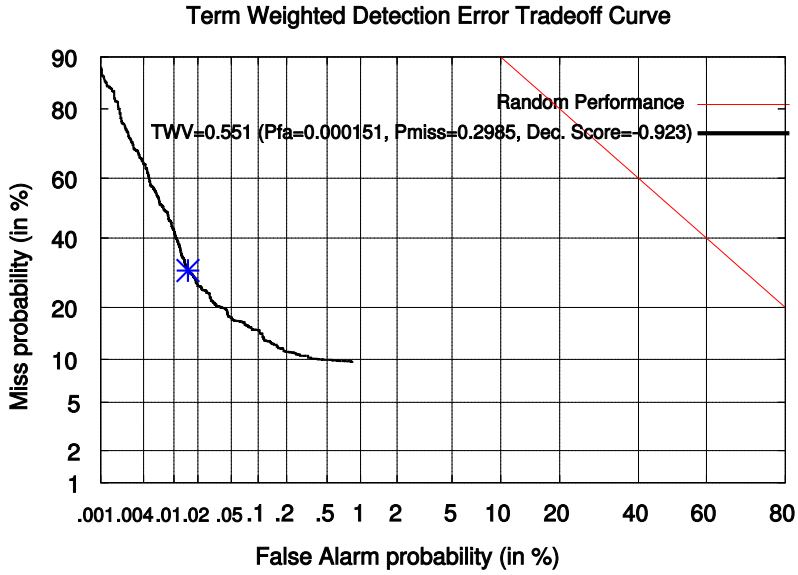


Figure 3.4: Example of detection error trade-off curve of one of our spoken term detection system. The blue star denotes operating point, where term weighted value is maximized. This point corresponds to threshold -0.923.

primary metric for NIST STD 2006 evaluations. The definition is the following:

$$TWV(thr) = 1 - \underset{term}{average}\{p_{MISS}(term, thr) + \beta p_{FA}(term, thr)\} \tag{3.11}$$

where

$$\beta = \frac{C}{V}(Pr_{term}^{-1} - 1). \tag{3.12}$$

C is the cost of incorrect detection (FA), V is value of correct detection (HIT), Pr_{term} is the prior probability of the term. C/V was set to 0.1 and Pr_{term} was set to 10^{-4} for NIST STD 2006 evaluations. Then, the β is constant equal to 999.9. An example of the TWV and DET curve is in figure 3.4.

Let us theoretically analyze the behavior of TWV for a term. The probability of false alarm $p_{FA}(term, thr)$ depends on number of false alarms $N_{FA}(term, thr)$ and number of non-target trials (possible places for rising a false alarm) $N_{nontarget}(term)$. According to (3.10), $N_{nontarget}(term)$ depends on T_{speech} and $N_{target}(term)$. T_{speech} is equal to $3 \times 60 \times 60 = 10800$ for 3 hours of test data (the length of NIST STD test data). The average $N_{target}(term)$ is between 1 and 700 with mean $\approx$ 15. Because $T_{speech} \gg N_{target}(term)$, $N_{nontarget}(term)$ primarily depends on T_{speech}, which is constant (for certain data set). That is why $p_{FA}(term, thr)$ depends "linearly" only on the number of false alarms. Each false alarm has approximately the same cost independently on number of reference term occurrences $N_{target}(term)$.

On the other hand, the cost of miss is highly dependent on the number of occurrences of reference term $N_{target}(term)$. $p_{MISS}(term, thr)$ depends on $N_{HIT}(term, thr)$ divided by $N_{target}(term)$ according to (3.5). For example, if a term has $N_{target}(term) = 2$ and the system misses one detection $N_{HIT}(term, thr) = 1$, the $p_{MISS}(term, thr)$ is equal to 0.5. The cost of one miss (by TWV point of view) is 0.5 according to (3.11). If a term has $N_{target}(term) = 100$, the cost of one miss (by TWV point of view) is 0.01.

Conclusion on TWV: A miss is much more expensive for less occurring terms than for frequently occurring terms. On the other hand, a false alarm is equally expensive for less

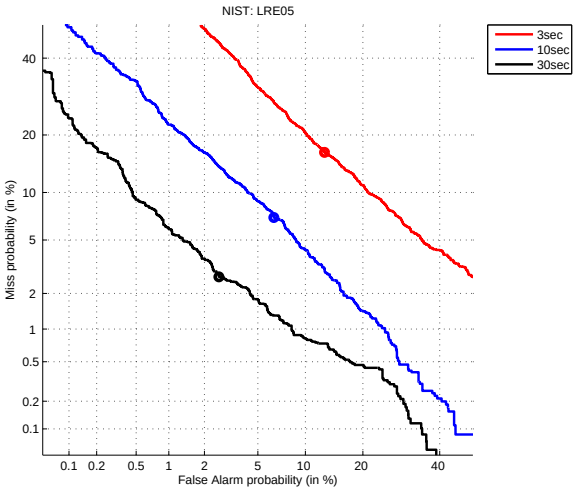


Figure 3.5: Example of detection error trade-off curve of one of our language recognition system.

as well as for widely occurring terms. The global TWV is averaged over each term's TWV, therefore each term has equal weight. That's why TWV "forces" to lower the threshold for less occurring terms. It is better to "pay a bit" for several false alarms than to "pay a lot" for one miss (especially for less occurring words). This leads to dependency of threshold on the number of term occurrences.

NIST offered a scoring tool providing the DET curves and computation of TWV. The scoring tool produces two values: an *Actual TWV* (ATWV) and a *Maximum TWV* (MTWV). The Actual TWV is TWV for the threshold set by STD system and this value was used as the primary metric for system comparison in NIST STD 2006 evaluation. Maximum TWV is maximized TWV and does not depend on the threshold of the STD system. *MTWV* is used further in this thesis to overcome dependency of system accuracy on the selected threshold. MTWV is further denoted as TWV for simplifying the notation.

3.3.4 Upper bound term-weighted value – UBTWV

One feature of TWV metric is its one global threshold for all terms. This is good for evaluation for end-user environment. On the other hand, it leads to uncertainty in comparison of different experimental system setups. We do not know if the difference is caused by different systems or different normalization and global threshold estimation. This is reason for our definition of **Upper Bound TWV** (UBTWV). The difference to TWV is in individual threshold per each term. The ideal threshold for each term is found to maximize term's TWV:

$$thr_{ideal}(term) = \arg\max_{thr} TWV(term, thr), \tag{3.13}$$

and UBTWV is then defined as:

$$UBTWV = 1 - \underset{term}{average}\{p_{MISS}(term, thr_{ideal}(term)) + \beta p_{FA}(term, thr_{ideal}(term))\} \tag{3.14}$$

This is equivalent to shifting the score of each term, so that maximum $TWV(term)$ is obtained at threshold 0.0. Two systems can be compared by UBTWV without any influence of normalization and ideal threshold level estimation in systems producing TWV score. The *actual* and *maximal* values are equal for UBTWV and both are denoted by

UBTWV. An example of DET and UBTWV are shown in figure 3.6. However, due to the fact that each term has its ideal threshold, DET curve for such ideal system has not much sense. Only the point corresponding to the ideal threshold is important. This point is supplied by the UBTWV. That is why only UBTWV values without DET curves are reported in this thesis.

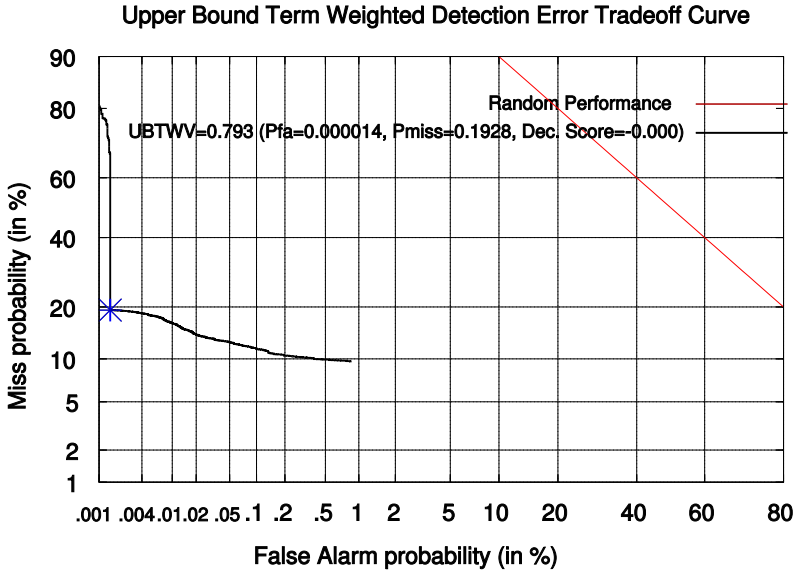


Figure 3.6: Example of detection error trade-off curve of one of our spoken term detection system when term score calibration was used. The blue star denotes the threshold level 0.000 where the UBTWV is calculated.

3.4 Other Evaluation Criteria

System time and computational requirements are usually not taken into account in evaluation of accuracy of a system. Primary scoring criteria are aimed to the pure accuracy as word accuracy, DET curves, term weighted value, FOM, equal error rates etc. But computational requirements are also important in practical use of the system. If one of two STD systems is only 0.5% less accurate but 5 times faster then the other, the computational requirements can be more important than the accuracy for customer.

3.4.1 Lattice Size

The output of the speech recognition system presented in this thesis is a lattice. Lattices are indexed in STD applications which should handle lots of data and should provide fast search. The size of lattice is the important value. Lattice size can be easily expressed by the number of links, nodes or gzipped file size. However, this expression does not reflect what is happening during lattice indexing. The size of index is more important than the number of nodes. Our **lattice size** (denoted as *SIZE*) reflects the size of index created during indexing of the lattice.

The procedure of lattice size calculation is the following:

1. Groups of the same overlapped words are found in the lattice.

2. Each group is substituted by one candidate and this is the count in the index size. This reflect things happening during the term indexing and search. Exactly the same process is done during term confidence estimation described in section 2.3.2. This holds however only for word lattices. The size of word lattice is denoted **wrdSIZE**.

Indexing of phone lattices cannot be done by direct indexing of phones. Because of low number of phones (45 in our case) and large number of phone occurrences, the index would be unbalanced and the search slow. Considering the past research of subword unit indexing [Ng00b], the optimal way to index phones seems to index phone trigrams. We reflect this fact also in phone lattice size calculation. Phone trigrams are generated first, then the same procedure is applied as is for the word lattices. Groups of the same phone trigrams are identified and each of this group is substituted by one candidate. Then these candidates are counted and represent the lattice size. The size of phone lattice is denoted **phnSIZE**.

The number of indexed tokens (words or phone trigrams) is usually in millions in our experiments. That is why, all reported values of wrdSIZE and phnSIZE are given in millions (1×10^6) in tables or graphs.

3.4.2 Decoder Requirements

Computational requirements are also important for the practical use of a STD system. On the other hand this is not the primary evaluation criterion. We do not evaluate the real-time factor[6] or memory consumption of each system. We use this information just for different systems to show if there is an essential difference in speed or memory consumption. The RTF is evaluated on Intel® Xeon® CPU, model E5345 at frequency 2.33GHz processor with sufficient size of RAM.

3.5 Used Tools

Taking into account that spoken term detection is a complex task, we used several software toolkits in our work. Brief description of used programs and toolkits is mentioned in this section. We used standard and well known toolkits and several "in house" programs. All software is public and freely available for research purposes. All experiments were done on Linux operating system.

3.5.1 Hidden Markov Model Toolkit – HTK

Hidden Markov Model Toolkit – *HTK*[7] was developed by Steve Young et al. at Machine Intelligence Laboratory (formerly known as the Speech Vision and Robotics Group) of Cambridge University Engineering Department (CUED). This toolkit is freely usable for research. The following description was taken from the project web page and HTKBook.

[6] Real-time factor: amount of CPU time needed to process certain amount of speech. Higher RTF means slower system.

[7] HTK: `http://htk.eng.cam.ac.uk/`

> *HTK is a toolkit for building Hidden Markov Models (HMMs). HMMs can be used to model any time series and the core of HTK is similarly general-purpose. However, HTK is primarily designed for building HMM-based speech processing tools, in particular recognizers. Thus, much of the infrastructure support in HTK is dedicated to this task. HTK contains both, the HTK training (feature extraction, tool for discriminative training and feature transformations) and the HTK recognition tools (large vocabulary decoder).*

In this thesis, HTK is mainly used for feature extraction. Our *STK* (section 3.5.4) is then used for training and recognition step.

3.5.2 SRILM toolkit

The **SRI Language Modeling Toolkit** – *SRILM*[8] has been under development by Andreas Stolcke et al. in the SRI Speech Technology and Research Laboratory since 1995.

SRILM is a toolkit for building and applying statistical language models (LMs), primarily for use in speech recognition, statistical tagging and segmentation, and machine translation. The toolkit supports creation and evaluation of a variety of language model types based on N-gram statistics, as well as several related tasks, such as statistical tagging and manipulation of N-best lists and word lattices. SRILM can be used freely for non-profit purposes.

We use SRILM for estimation of n-gram language models from textual corpora.

3.5.3 OpenFST library

We are using **Open Finite State Transducers** – *OpenFST*[9] library for building the recognition network. The following description was taken from the project web page:

> *This library was developed at Google Research (M. Riley, J. Schalkwyk, W. Skut) and NYU's Courant Institute (C. Allauzen, M. Mohri). OpenFST is a library for constructing, combining, optimizing, and searching weighted finite-state transducers (WFSTs). Weighted finite-state transducers are automata where each transition has an input label, an output label, and a weight. WFSTs have key applications in speech recognition and synthesis, machine translation, optical character recognition, pattern matching, string processing, machine learning, information extraction and retrieval among others. Often a weighted transducer is used to represent a probabilistic model (e.g., an n-gram model, pronunciation model). WFSTs can be optimized by determinization and minimization, models can be applied to hypothesis sets (also represented as automata) or cascaded by finite-state composition, and the best results can be selected by shortest-path algorithms. It is intended to be comprehensive, flexible, efficient and scale well to large problems. It is an open source project distributed under the Apache license.*

Finite state transducers framework is widely used for building static recognition network for speech recognition. We use this library in the same way. Language model and pronunciation dictionary are converted into weighted finite state transducers and using a composition operation, the network is built.

[8] SRILM: `http://www.speech.sri.com/projects/srilm/`
[9] OpenFST: http://www.openfst.org/

3.5.4 STK toolkit

Speech recognition software (decoder) used in this thesis is part of the HMM Toolkit **STK**[10]. The STK was used also for discriminative training of acoustic models. STK is publicly available under the GNU General Public License.

STK is a set of HMM-tools created at Speech@FIT, Brno University of Technology, Faculty of Information technology (FIT BUT) in closed connection to EU-funded projects M4[11], AMI[12] and AMIDA[13]. STK was inspired by HTK (section 3.5.1) which has become a popular tool for almost all people involved in speech processing research and applications that have anything to do with Hidden Markov models (HMMs). On the other hand, we identified several problems of HTK:

- HTK, even if composed of libraries and associated executables, is quite hard to be modified by a person exterior to Cambridge University HTK team.
- People often consider HTK to be a black-box ... what's not possible with HTK, is not possible at all.
- There are serious license issues (limited exploitation of the standard tools and some tools, such as `HDecode` not available at certain times).
- One learns often better by writing his/her own software rather by analyzing and modifying other's one.

This lead us to build a home-made toolkit for the work with HTK. All the code was written at FIT BUT although we to stay compatible with HTK especially on the level of model-definition files and command-line interface. New features the toolkit brings in comparison to HTK are:

- The possibility to use and train linear transforms at different places in the set of HMMs.
- Static Viterbi decoder working with generally built recognition networks.

3.5.5 LatticeSTD

The **Lattice Spoken Term Detection** – *LatticeSTD*[14] tool is written by the author of this thesis. It is used in majority of reported STD experiments. This tool searches given lattices for a set of terms. Each term is defined as a pair `Output label` and `Input labels`, for example `Term01 PRESIDENT GEORGE BUSH` or `IGOR SZÖKE ih g oh r s z ow k eh`. If the `Input labels` sequence is found in lattice, confidence is estimated and the output label is written to the output. This tool *searches for exact sequence* of links (or nodes) representing the output label. No other links (or nodes) are allowed to be among the searched link sequence. *LatticeSTD* tool allows estimation of several types of confidence measures, filtering of detections and substitutions/insertions/deletions of phones in case of phone lattices search.

The *LatticeSTD* tool is publicly available under the GNU General Public License.

[10] STK: `http://speech.fit.vutbr.cz/en/software/hmm-toolkit-stk`
[11] M4: `http://www.dcs.shef.ac.uk/spandh/projects/m4/`
[12] AMI: `http://www.amiproject.org/`
[13] AMIDA: `http://amidaproject.org/`
[14] LatticeSTD: `http://speech.fit.vutbr.cz/en/software/lattice-spoken-term-detection-toolkit-latticestd`

3.5.6 LSE

Lattice Search Engine – *LSE*[15] has been developed by Michal Fapšo at BUT. LSE was used in the NIST STD evaluation in 2006 as an application providing indexing and fast search.

LSE indexer indexes lattices produced by speech recognizer. Several types of indices are stored and accessed by *LSE searcher*. The searcher searches for occurrences of given queries in the indices. The queries can be composed of single or multiple words. A query can also contain time constrains – maximal allowed time distance between words. Phone trigram based STD was implemented in LSE as subword version for OOV word search. LSE can identify OOV parts of query and search them in phone lattices. Finally the IV and OOV results are combined together into results fulfilling the time constrains. LSE allows also to interconnect query words with surrounding words using indexed lattices. Detailed description of the engine was published in [Fap07] and it is over the scope of this thesis.

The *LSE* is publicly available under the GNU General Public License.

3.5.7 G2P

Grapheme to phoneme – *G2P* converter was developed by Speech@FIT student Stanislav Kontár. Unfortunately, he left our laboratory without publishing a paper about the approach he used. A brief description of the method is provided in appendix B. The G2P system was trained on pronunciation vocabulary used in here presented LVCSR system (chapter 4). The 50k word vocabulary was split to 9/10 used for training and 1/10 used for test. The accuracy of generated pronunciation was: 73% correctly generated word pronunciations and 16% word pronunciation having error only in one phone.

[15] LSE: `http://speech.fit.vutbr.cz/en/software/lattice-search-engine-lse`

Chapter 4

Word recognition

This section deals with the description of our large vocabulary continuous speech recognition system (LVCSR) used for experiments stated in this thesis. Presented LVCSR is a state-of-the-art system derived from AMI LVCSR[1] [HWB+07]. The AMI LVCSR system was slightly modified and used in the NIST STD 2006 evaluation. The decoder was changed from HTK *HDecode* to "in-house" STK *SVite* in the third (final) pass and produced lattices are directly used for STD. In the AMI LVCSR, the lattices were expanded by fourgram language model and confusion networks were applied.

The reason for the decoder replacement is in the principle of decoding. The HTK HDecode is a dynamic decoder whereas STK SVite is static decoder. By the term *dynamic*, we mean that the decoder loads n-gram language model, pronunciation dictionary and acoustic models (GM-HMMs) and builds the recognition network on-line. This is optimal from the computational resources point of view. Only promising parts of the network (search space) can be built according to words spoken in the decoded utterance.

By the term *static* decoder, we mean that the decoder loads already "compiled" network and acoustic models. The decoder is just a "simple" machine which propagates tokens through the network and evaluates the acoustic models according to given utterance. The disadvantage is in required resources, because complete recognition network must be loaded into memory. Depending on the utterance, considerable part of the network is not used (no token passes through) during decoding. On the other hand, advanced techniques and optimizations can be applied during the network compilation, especially if finite state machines (FSM) framework is used (section 4.2). The most important advantage of building the recognition networks outside of the decoder is in generating the networks for hybrid word-subword recognition (section 7.1.1).

4.1 The recognizer

The input data (conversational telephone speech) is first converted to linear coding 16-bits per sample and 8kHz. The data is then segmented to speech/silence according to energy in channels and by a neural net based phone recognizer [SMČ04]. All phone classes are linked to "speech" class.

The data is split into shorter segments on silences (output of speech/non-speech detector) longer than 0.5s. If the speaker changes, the data is also split. Segments longer than 1 minute are split into 2 parts in silence closest to the center of the segment. This

[1] The LVCSR was developed in cooperation with AMI-project partners, see `http://www.amiproject.org`.

is done to overcome long segments and accompanying problems during decoding (long decoding time and high memory consumption).

The large vocabulary continuous speech recognition system (LVCSR) system used in this thesis is a simplified version of AMI LVCSR system used for NIST RT 2006 evaluations [HBD+06]. The system operates in 3 passes (figure 4.1):

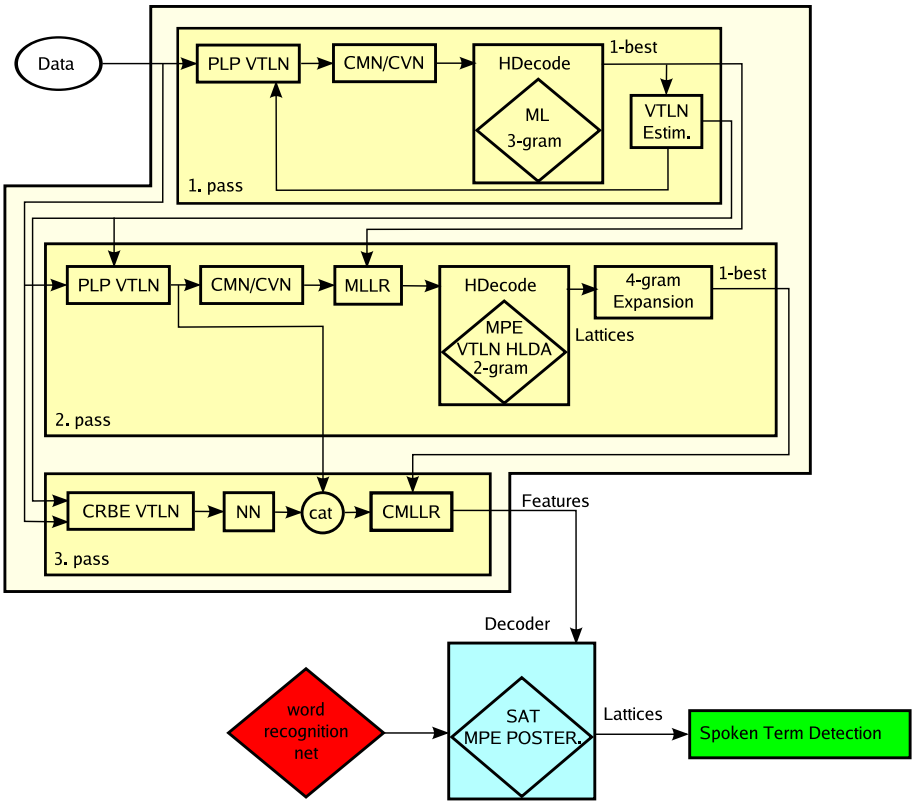


Figure 4.1: Schema of 3-pass recognition system used in this thesis. The system is derived from AMI LVCSR.

In the **first pass** – **P1**, the front-end converts the segmented recordings into feature streams, with vectors comprised of 12 *Mel-Frequency Perceptual Linear Prediction* (MF-PLP) features and raw log energy. First and second order derivatives are added. After, *Cepstral Mean and Variance Normalization* (CMN/CVN) is performed on a per channel basis. The first decoding pass yields initial transcripts that are subsequently used for estimation of *Vocal Tract Length Normalization* (VTLN) warp factors. The feature vectors and CMN and CVN are recomputed after the application of VTLN.

The **second pass** – **P2** processes the new features and its output is used to adapt models with *Maximum Likelihood Linear Regression* (MLLR). Bigram lattices are produced and re-scored by trigram and fourgram language model.

In the **third pass** – **P3**, posterior features [GKKČ07] are generated. The output from the second pass is used to adapt models with *Constrained MLLR* (CMLLR) and MLLR. *In the original AMI LVCSR*, bigram lattices were produced by HDecode decoder and re-scored by fourgram language model. *In this thesis*, the output of the third pass are the features which are processed by SVite decoder.

4.1.1 Feature extraction

The CTS system uses standard cross-word tied states HMM using MF-PLP's generated in classical way with 15 filterbank channels. The resulting number of cepstral coefficients is always 13.

The following techniques are used in HMM training:

- CMN/CVN is applied per speaker.
- VTLN warping factors are computed using Brent search method and features are re-computed.
- Deltas, double- and triple-deltas are added into the basic PLP feature stream, so that the feature vector has 52 dimensions. *Heteroscedastic Linear Discriminant Analysis* (HLDA) is estimated with Gaussian components as classes [KGS+06]. HLDA is estimated to reduce the dimensionality to 39.
- Bottle-neck [GKKČ07] LC-RC [SMČ04] posterior system splits 310ms temporal context in each filterbank output into two halves and each half is processed by one neural net producing phone posteriors. These are merged by a third neural net (the merger). It is 5-Layer NN with middle layer containing 35 neurons only. Nonlinearly compressed information here is used as the output. The HLDA is estimated to de-correlate and to reduce dimensionality from 35 to 25. Again, the resulting features are concatenated with PLP features ($25 + 39 = 64$) and mean and variance normalized. This procedure is shown in figure 4.2.

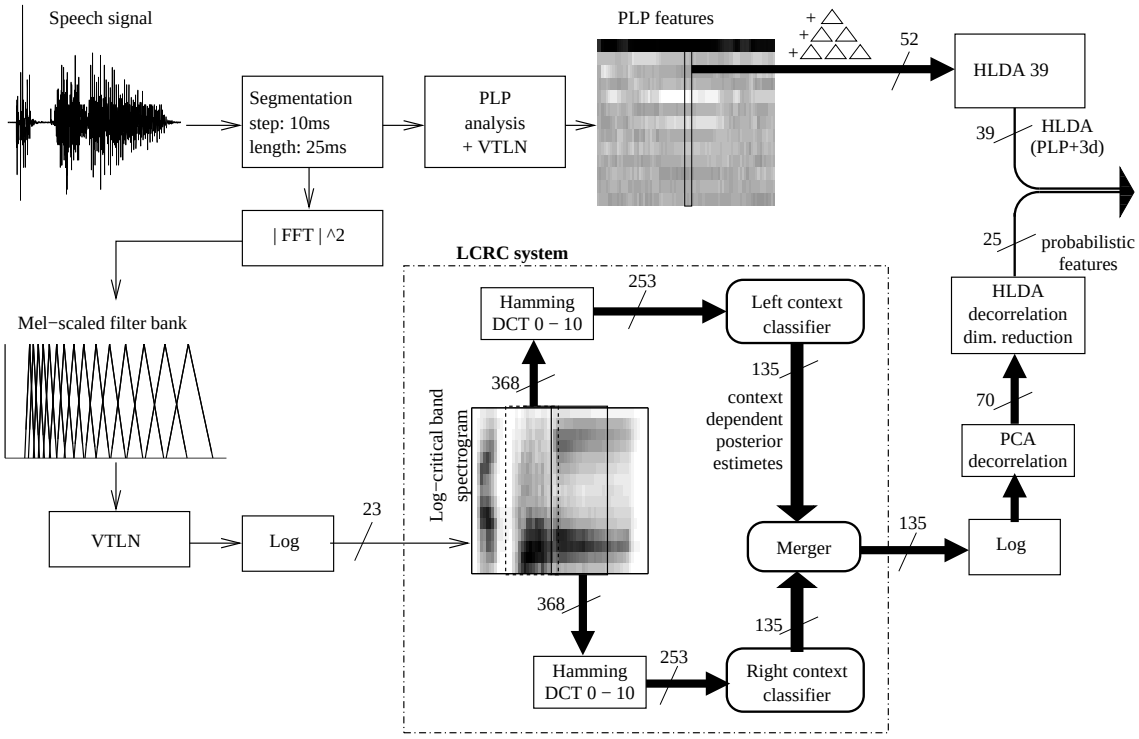


Figure 4.2: Scheme of feature extraction during the third pass **P3** in the recognition system. Partly reproduced from [Gré07].

4.1.2 Acoustic model

Training of posterior features At first, the neural network training is done on 30h data used for training of LVCSR acoustic models. The data has cepstral mean and variance normalization and VTLN. Using these nets, full features are generated on all the data. The outputs are concatenated with PLP VTLN HLDA feature stream. The CMN/CVN are re-computed again and the models are trained by single-pass re-training. Further, the models are re-clustered and trained by mix-up procedure from 1 to N Gaussians. The optimal number of Gaussians per state is tuned to 26.

Speaker-adaptive training One single Constrained MLLR (CMLLR) transform is trained per each conversation-size. Features are mapped to unique *Speaker-Adaptive Training* (SAT) space by CMLLR and 8 iterations of *Maximum Likelihood* (ML) training (standard Baum-Welch) are run. After, new CMLLR transforms are trained, features transformed and 8 ML-iterations followed. And once more, so that the number of CMLLR+re-training macro-iterations is 3.

Discriminative training The models are re-trained in 15 iterations of *Minimum Phone-Error* (MPE) training [Pov03]. The alternative hypotheses for MPE are generated by much simpler system including just ML-trained models on PLP+HLDA without any adaptation. In case of SAT-MPE-training, we do not re-train the CMLLR transforms.

Pass 1	HLDA
Pass 2	VTLN HLDA MPE
Pass 3	VTLN Bottleneck-LC-RC SAT MPE

Table 4.1: Brief overview of techniques used for acoustic models in each pass.

The *ctstrain04* corpus containing about 278 hours of well transcribed speech data from Switchboard I, II and Call Home English is used for the training. It is a subset of *h5train03* set defined at Cambridge University as a training set for Conversation Telephone Speech (CTS) recognition systems [HBD$^+$05]. The Hungarian SpeechDat-E [PČB$^+$00] data is used for the speech/silence detector (phone recognizer).

4.1.3 Word language model

The training of 4-gram language models was done within the framework of AMI project at University of Edinburgh by Vincent Wan. Table 4.2 shows the training data used. The perplexity was maximized for the CTS task. A closed vocabulary bigram language model is used for generating the final lattices.

4.1.4 Open vocabulary word language model

Experiments in chapter 7 are aimed to combined word-subword recognition. We need open vocabulary language model with a symbol which will represent OOV words for these experiments. Vincent Wan created and sent us corpora and training scripts for building open vocabulary language models. The description of the corpora is in table 4.3. All words which are not present in the target vocabulary (LVCSR vocabulary) are marked as `<unk>` symbol (unknown word).

Corpus	# of words
Swbd/CHE	3.5M
Fisher	10.5M
Web (Swbd)	163.0M
Web (Fisher)	484.0M
Web (Fisher topics)	156.0M
BBC-THISL	33.0M
HUB4-LM96	152.0M
SDR99-Newswire	39.0M
Enron email	152.0M
ICSI/ISL/NIST/AMI	1.5M
Web (ICSI)	128.0M
Web (AMI)	100.0M
Web (CHIL)	70.0M
Sum	**1492.5**M

Table 4.2: Corpora and number of words per corpus used for language model training.

4.2 Building a recognition network

Not only the language model is needed for the word recognition. Also, a word dictionary mapping word labels to pronunciation forms is incorporated in the recognizer. If the context-dependent phones (triphones) are used as acoustic models, usually a "tied-list" is needed. The number of all triphones is theoretically $N_{triphones} = N^3_{phones}$. Many of the triphones are rare or even never occur. So a mapping (tying) of these rare triphones to similar more frequent triphones is used during the acoustic model training.

The recognition done by the decoder is driven by the **recognition network**. The components needed to build such a network are **language model**, **pronunciation dictionary** and **tied-list**. The compiled network contains connections of words with probabilities given by the language model, and mapping from words to pronunciations down to decomposition of context-dependent phones to tied states. The recognition is then done

Corpus	# of words
CMU+ICSI+NIST meetings	153.0M
Fisher	600.0M
AMI	1.0M
h5etrain03v1	4.0M
ICSI	0.8M
ISL	0.1M
lm96	148.0M
NIST	0.1M
rt06s-AMI-CMU-NIST	70.0M
Sum	**977.0**M

Table 4.3: Corpora used for the open vocabulary language model training.

by token-passing within this network.

Standard decoders, such as HTK *HDecode* build the network internally and in dynamic way. However, our work requires full control over the creation of the network. That is why we used our static decoder *SVite* and compiled the network off-line.

Theoretical *finite state machine* framework can be used for building the recognition networks. The recognition network can be seen as a **weighted finite state transducer** (*WFST*) which maps a sequence of hidden Markov models to a sequence of word labels accepted by a language model (weighted finite state acceptor – WFSA). WFSTs provide a natural representation of hidden Markov models, pronunciation dictionary and language model [MPR08] in automatic speech recognition.

The *weighed finite state transducer* is a finite state machine that encodes a mapping between input and output symbol sequences. Weighted transducer associates weights such as probabilities, durations, penalties or any other quantities that accumulate linearly along paths, to each pair of input and output symbol sequence. Weighted determinization and minimization algorithms optimize their time and space requirements, and a weight pushing algorithm distributes the weights along the paths of a weighted transducer optimally for speech recognition.

Consider a pronunciation lexicon and take its Kleene closure by connecting an ϵ-transition from each final state to the initial state. The resulting pronunciation lexicon can transcribe any sequence of words from the vocabulary to the corresponding phone sequence.

Consider a language model G and a pronunciation lexicon L. The composition of these two WFSTs,

$$L \circ G, \tag{4.1}$$

gives a transducer that maps from phones to word sequences while assigning a language model score to each such sequence of words. Incorporating context-dependent triphone models is a simple matter of composing

$$C \circ L \circ G, \tag{4.2}$$

where C represents the mapping from context-dependent to context-independent phonetic units. Then, incorporating HMM models H:

$$H \circ C \circ L \circ G, \tag{4.3}$$

results in a transducer capable of mapping distributions to word sequences restricted to the language model G.

We are using the *OpenFST library* (section 3.5.3) for building the word-recognition network. Thorough description of WFST framework and its application in speech processing was given by M. Mohri et al. in [MPR08].

4.3 Word recognition results and tuning of decoder parameters

Before we start STD experiments, we must find optimal parameters of the decoder: *word insertion penalty*, *language model scale factor*, *beam pruning* and *maximum active models*. Of course, we should experiment with pruning off or as little as possible, but the lattices would be huge, decoding would take to much time and search in the lattices would require prohibitive times. Optimal choice of parameters can speed-up decoding significantly without any significant loss of accuracy.

The recognition system produces word lattices. Lattice size can be controlled by **beam pruning** parameter (Pr) and **maximum active models** parameter (MAM) in the *SVite* decoder. Pruning parameters have standard "HTK behavior": all tokens with likelihood lower than the best one minus Pr are killed. The MAM limits the number of active models (models having a token) per each frame. This parameter has influence on the lattice width. If the decoder is uncertain in a part of utterance, it can produce very wide lattices. This can usually happen in OOV, non-English or non-speech parts of utterances. MAM can effectively reduce the size of these lattice parts. The word insertion penalty and language model scale factor are tuned to maximize UBTWV[2].

Also, the language model has significant impact on speed and memory consumption during decoding. As *SVite* is a static decoder, it loads the whole recognition network into memory. The size of language model can be reduced by pruning of low probable bigrams[3]. The impact of language model pruning on the lattices and recognition results (word accuracy, word lattice accuracy and UBTWV) is shown in table 4.4. We can see that the language model can be reduced down to one third of bigrams without any significant loss of accuracy. If the pruning is higher than 2×10^{-9}, word accuracy and UBTWV start to decrease.

LM Pruning	wrdSIZE	WAC	WLAC	Word UBTWV-ALL	# bigrams
none	0.506	70.84	88.22	0.793	10121k
1×10^{-10}	0.506	70.84	88.21	0.793	7542k
1×10^{-9}	0.509	70.81	88.23	0.793	4409k
$\mathbf{2 \times 10^{-9}}$	**0.510**	**70.78**	**88.22**	**0.794**	**3369k**
5×10^{-9}	0.512	70.59	88.21	0.789	2311k
1×10^{-8}	0.513	70.50	88.24	0.783	1715k

Table 4.4: Different lattice size, word accuracy, word lattice accuracy and UBTWV depending on word language model pruning (MAM 5000, Pr 260).

The following set of experiments aimed to find the optimal pruning parameters. We tune the pruning and the MAM parameters and watch the wrdSIZE of produced lattices, word accuracy, word lattice accuracy and UBTWV. The values are plotted for better readability (figure 4.3).

The UBTWV accuracy does not significantly depend on MAM for both 220 and 260 pruning factors. The effect of MAM is more important for higher pruning factor (figure 4.3). MAM saturates around 5000 maximum active models for pruning 220 and 260. The word accuracy and word lattice accuracy improve only negligibly for MAM higher than 5000. The lattice size has saturating tendency too.

The accuracies do not saturate so clearly depending on pruning. They still slowly increase when the pruning is lowered. We can find optimal pruning factor lying close to 260 for UBTWV and WAC. The WLAC has rising tendency for the whole tested range of the pruning factor. The lattice size rises more than linearly depending on the pruning factor. Selected decoding parameters are maximum active models 5000 and pruning 260 for word lattices on conversational telephone speech STD task in this thesis.

[2]Word insertion penalty and language model scale factor for maximizing word accuracy or UBTWV are different.

[3]We use only bigram word language model

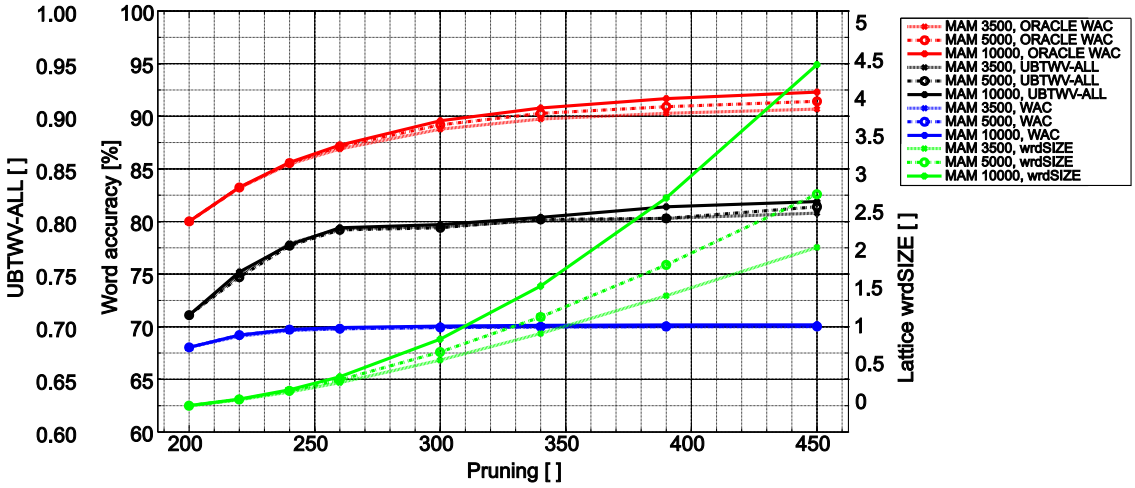


Figure 4.3: Dependency of word accuracy, word oracle accuracy, upper bound term weighted value and lattice size on pruning factor (for 3 different maximum active models factors).

4.4 Comparison of confidence estimations for word system

Several approaches for estimation of term confidence were proposed in section 2.3.2. UBTWV and TWV metrics used for evaluation of STD were described in section 3.3. The WRD system is used for evaluation of different term confidences. The influence of confidence measure is summarized in table 4.5. The results show, that SOLP and C_{max} confidence measures gave about 0.030 better results than the baseline LP on the TWV metric. The different term confidences are comparable for the UBTWV. As the UBTWV has individual threshold per each term, the difference in the UBTWV accuracy should be caused only by shifting possible hits and false alarms within the term detections. Significant accuracy improvement is seen, if the confidence estimation methods are evaluated by the TWV. The TWV has one global threshold, so the candidate score is also important in comparison to other terms.

This led to the conclusion, that SOLP and C_{max} confidence measures are better and produce scores that result in comparable accuracies. In experiments related to comparison of confidence measures, we stick with SOLP as computational requirements are higher for C_{max}.

Evaluation metric	Word TWV			Word UBTWV		
Confidence measure	ALL	IV	OOV	ALL	IV	OOV
LP	0.552	0.529	0.719	0.793	0.773	0.840
SOLP	**0.580**	**0.579**	**0.717**	**0.795**	**0.777**	**0.838**
SCOLP	0.540	0.513	0.719	0.787	0.764	0.840
C_{max}	0.580	0.580	0.717	0.795	0.777	0.838

Table 4.5: Comparison of different methods of term confidence estimation and its influence on the TWV and UBTWV scores for the WRD system.

The results in table 4.5 also show, that UBTWV accuracies of LP, SOLP and C_{max} confidence measures are nearly the same. In section 2.3.2, we stated that the LP confidence measure will be used further in this thesis – if the UBTWV metric is used, the LP confidence should be "good enough"; the advantage is its computational simplicity.

4.5 Baseline word recognition systems

Selected LVCSR system parameters are *LM pruning* 2×10^{-9}, $Pr = 260$ and $MAM = 5000$. System with these parameters achieved very good accuracy, small size of lattices and low decoding time. It is important to note, that this was original AMI CTS system with closed vocabulary language model. This baseline closed vocabulary LVCSR system was denoted as *WRD*. Word recognition system with reduced vocabulary (derived from *WRD* system) was used in two following chapters 5 and 6. It was *WRD* system where 880 words were omitted from the vocabulary. Details of vocabulary reduction were given in section 3.1.1. This system was denoted as *WRDRED*. Both these baseline systems were compared in the upper part of table 4.6.

However, open vocabulary language model is needed for our later experiments in chapter 7. This language model was trained on data stated in section 4.1.4. The open vocabulary language model had to be trained "from scratch" (not only by omitting 880 words) in order to correctly estimate the probabilities of the "out-of-vocabulary" symbol `<unk>`. The optimal language model pruning 3^{-9} was found for the open vocabulary LM in the same way as was presented in this chapter. This LM has 1.95M of bigrams. We also found the pruning $Pr = 260$ too soft in hybrid word-subword recognition experiments. Pruning $Pr = 220$ was found more practical for hybrid recognition in chapter 7.

To make the systems comparable, we created *WRDforHYB* system, which should be comparable to baseline open vocabulary word recognition systems presented in chapter 7. The *WRDforHYB* system used $Pr = 220$ and LM pruning 1×10^{-8} because it is close to the open vocabulary LM in terms of number of bigrams: The accuracies of *WRDforHYB* system are presented in the bottom part of table 4.6.

System	Decoder Pruning	LM Pruning	wrdSIZE	WAC	WLAC	Word UBTWV		
						ALL	IV	OOV
WRD	260	2×10^{-9}	0.510	70.78	88.22	0.795	0.777	0.838
WRDRED	260	2×10^{-9}	0.507	68.41	85.75	0.522	0.747	0.000
WRDforHYB	220	1×10^{-8}	0.252	69.04	83.21	0.738	0.734	0.746

Table 4.6: Comparison of lattice size, word accuracy, word lattice accuracy and UBTWV of different baseline LVCSR systems. WRD is the "full" vocabulary baseline, WRDRED is the reduced vocabulary baseline and WRDforHYB is "full" vocabulary baseline comparable to open vocabulary LVCSR in terms of LM size and decoder pruning.

Chapter 5

Subword recognition – phones

This chapter deals with comparison of baseline subword techniques for the spoken term detection task. The subword units are used for detection of out-of-vocabulary words which are not present in the word recognizer vocabulary. Subword recognition system architecture and training corpora are described at the beginning of this chapter. Then phone units are evaluated. We experimented with phones generated by a standard "free phone loop" and also phones produced by the LVCSR system. The following chapter deals with multigrams. Phone multigram units are built-up by concatenation of phones.

5.1 Subword recognition baseline system

Presented subword recognizers (phone, phone multigram, etc.) are based on the third pass of the LVCSR recognizer. The structure is the same and it uses the same features and models as the LVCSR. The difference is only in the recognition network. The schema of subword recognizer is in figure 5.1.

5.1.1 Training data for subword language model

ctstrain04 corpus (278h of conversation telephone speech, subset of *h5train03* corpus [HBD+05] defined on Cambridge University) is used for estimation of subword language model. The corpus has phone alignments and it is used for training of acoustic models. The *ctstrain04* corpus contains 11.5M phones. The corpus is searched for utterances containing out-of-vocabulary words and these utterances are omitted (denoted *LnoOOV*). This is done to remove OOV words from subword models training data (to prevent cheating by knowing the OOVs). Corpus denoted as *LOOV* and having the same size as *LnoOOV* is derived from *ctstrain04* to produce results for comparison. The sizes of corpora are summarized in table 5.1.

Notation	# of utterances	# of phones (incl. sil)	# of phones (w/o sil)
ctstrain04	300.5k	11.21M	9.97M
LnoOOV	237.2k	6.40M	5.60M
LOOV	169.1k	6.26M	5.60M

Table 5.1: Comparison of corpora used for training of subword systems.

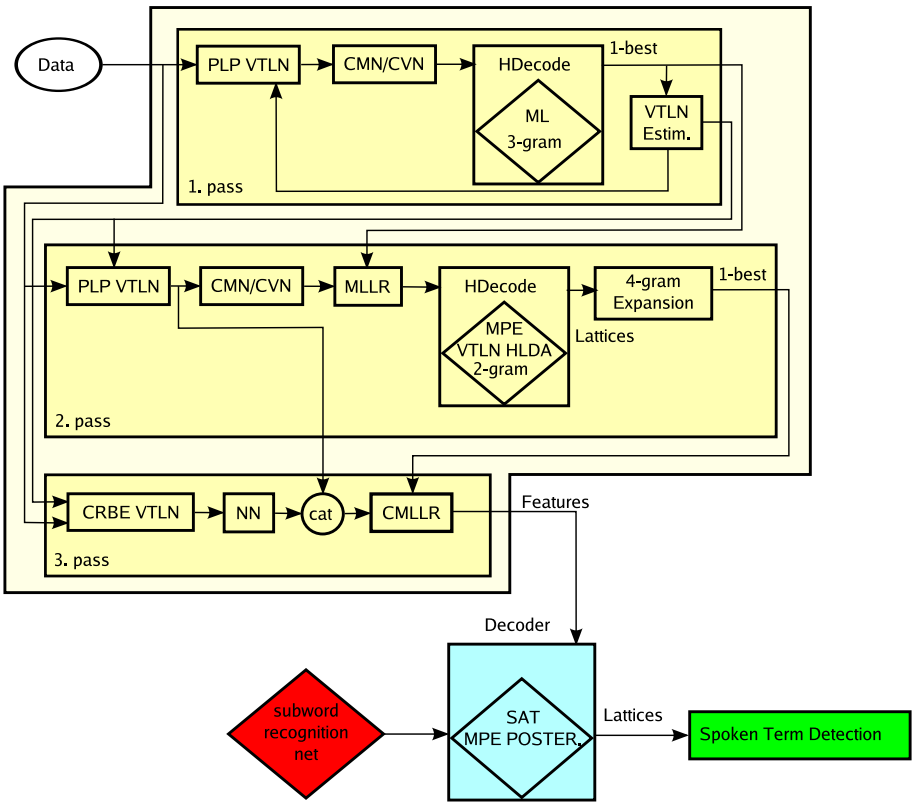


Figure 5.1: Schema of 3-pass recognition system used for phone or subword lattice generation.

5.1.2 Phone Loop

The phones are used as the most general subword units. Our phone recognizer uses the LVCSR acoustic models. They are cross-word context-dependent phones (cross-word triphones). These phone models are connected into a phone loop, which is simple in comparison to complex word recognition network in the LVCSR. The output of this phone recognizer is a string of phones or a phone lattice. The phone lattice is used for further experiments. Better STD accuracy is expected due to parallel hypothesis in the lattices.

Phone recognition is based on the same features and acoustic models as the LVCSR. The decoder parameter tuning of the phone system is done in similar manner to the LVCSR system. The bigram language model is trained on $LOOV$ corpus (see table 5.1) for this experiment.

We tune the beam pruning Pr and the maximum active models MAM parameter and regard the size of produced lattices *phnSIZE*, phoneme accuracy PAC, phoneme lattice accuracy $PLAC$ and $UBTWV$. The values are plotted for better readability in figure 5.2.

Phone accuracy does not depend much on the Pr and MAM parameters. UBTWV and PLAC do not saturate for higher Pr and MAM (in contrast to words). The main goal is therefore to obtain reasonably small phone lattices. That is why pruning $Pr = 180$ and maximum active models $MAM = 5000$ are chosen.

After we fixed the basic decoder parameters, we concentrated on phone language model. Experiment evaluating dependency of the accuracy on language model training corpora and the degree of n-gram language model is done. The corpora for language model training are described in section 5.1.1. Phone system results are summarized in table 5.2. We conclude that absence of OOV in the language model training corpora has no influence on

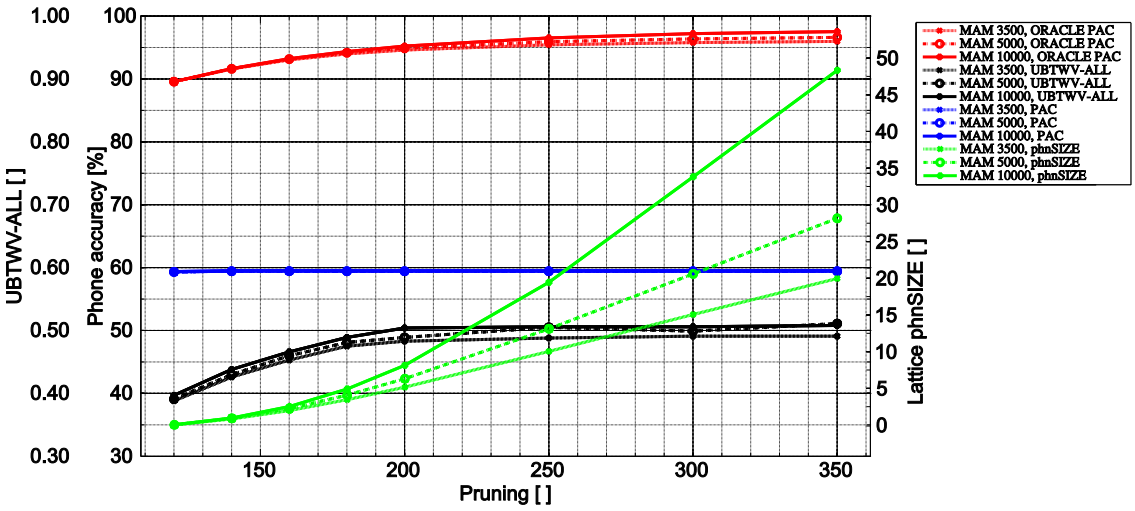


Figure 5.2: Dependency of phone accuracy PAC, lattice phone accuracy PLAC, upper bound term weighted value UBTWV and lattice size phnSIZE on pruning factor Pr (for 3 different maximum active models factors MAM).

the accuracy. The trigram LM provides the best phone accuracy. The STD accuracy is equal for bigram and trigram language models. The phone lattice size generated by the trigram are about 3 times smaller in comparison with the bigram. On the other hand, the recognition network is 7 times bigger compared to the bigram network, and also the decoding is significantly slower.

Corpus	LM	Phone		
	n-gram	PAC	UBTWV-ALL	phnSIZE
LOOV	1	53.85	0.365	16.93
LOOV	2	58.40	**0.483**	14.81
LOOV	3	**59.75**	0.481	**5.07**
LnoOOV	1	53.83	0.362	17.02
LnoOOV	2	58.42	0.483	14.92
LnoOOV	3	**59.66**	**0.483**	**4.88**

Table 5.2: Comparison of training corpora and order of n-gram language model on accuracy for phone based experiments. Decoding parameters: beam pruning $Pr = 180$ and maximum active models $MAM = 5000$.

5.2 Phones from words

The second phone-based system generates phone lattices from the LVCSR. Generating phone lattices from an LVCSR system has already been done by Witbrock et al. [WH97]. They stated that it achieved better accuracy than the free phone loop system in subword STD task. We used full LVCSR system, only the decoder was set to produce model labels (phones) instead of word labels as the output. The decoder (Pr and MAM) settings are the same as for the word decoding ($Pr = 260$ and $MAM = 5000$). Language model scaling factor and insertion penalty used for "word" decoding, and LM scale factor and insertion

penalty used for STD are tuned to achieve maximal accuracy.

The results are summarized in table 5.3. Note the impact of missing OOV words on the recognized phone sequences. This leads to drop of the UBTWV-OOVby 0.263.

Data type	WRDtoPHN	WRDREDtoPHN
PAC	67.28	65.40
UBTWV-ALL	0.634	0.540
UBTWV-IV	0.575	0.554
UBTWV-OOV	0.771	0.508
phnSIZE	3.80	4.34

Table 5.3: Phone accuracy and UBTWV for system where phone lattices are generated by full vocabulary and reduced vocabulary LVCSR systems.

5.2.1 Correction of language model likelihoods in phone lattices generated by word system

Phone lattices generated by the LVCSR system can be easily produced using *SVite* decoder. *SVite* internally produces phone lattices containing also word labels. The choice whether a word or a phone lattice is produced is set by an *SVite* switch and the lattice is modified after the decoding. One problem occurs in phone lattice generated in this way. The recognition network contains word language model which remains in the lattice. The phone lattice contains word language model likelihoods. These are pushed by a weight pushing along the whole recognition network during the construction of the network by *OpenFST*. This is why these pushed likelihood appear also in the phone lattices. If a term represented by a sequence of phones is searched, it can appear across several word parts, and the total language model likelihood is incorrect. This was corrected by replacing the "bad" word LM likelihoods by correct phone bigram language model likelihoods. The impact of replacing language model likelihoods on the accuracies is shown in table 5.4.

Data set	LM type	PAC	PLAC	Phone UBTWV		
				ALL	IV	OOV
WRDtoPHN	(Original) LVCSR	67.28	94.71	0.634	0.575	0.771
WRDtoPHN	(Replaced) Phone	62.66	94.71	0.559	0.490	0.718
WRDREDtoPHN	(Original) LVCSR	65.40	95.30	0.540	0.554	**0.508**
WRDREDtoPHN	(Replaced) Phone	61.28	95.30	0.483	0.473	**0.506**

Table 5.4: Comparison of phone accuracy, phone lattice accuracy and UBTWV depending on language model probabilities (word or phone) and the type of LVCSR.

We conclude that phone lattices generated by the LVCSR system with word language model likelihoods achieve higher overall accuracy. The phone accuracy deterioration is about 4% absolutely when the word language model is replaced by phone language model in the lattices. The overall spoken term detection accuracy also deteriorated. UBTWV deterioration (about 0.070) for full dictionary system has a reason. The word language model is better for word recognition (and spoken term detection). The deterioration for out-of-vocabulary terms (for reduced dictionary system – WRDRED) is negligible

(about 0.002) on the other hand. This proves that word language probabilities attached to the phones for OOV terms are not correct, but not so bad, because the OOV terms with correct phone probabilities have approximately the same score.

5.3 Comparison of word and phone based systems

This section aims at overall comparison of the phone loop based systems. We compare word system (denoted as *WRD*), phone system from phone loop (denoted as *PHN*) and phone system derived from the LVCSR (denoted as *WRDtoPHN*). We also aim to evaluate the accuracies of systems on OOVs. That is why we use the LVCSR with reduced vocabulary (see section 3.1.1 for description of this system). The systems with reduced vocabulary are denoted as *WRDRED* and *WRDREDtoPHN* respectively. Table 5.5 compares the results.

Data type	WRD	WRDRED	WRDtoPHN	WRDREDtoPHN	PHN
WAC	70.78	68.41	-	-	-
WLAC	88.22	85.75	-	-	-
PAC	-	-	67.28	65.40	59.65
PLAC	-	-	94.71	95.30	94.11
UBTWV-ALL	0.794	0.522	0.634	0.540	0.481
UBTWV-IV	0.775	0.747	0.575	0.554	0.453
UBTWV-OOV	0.841	0.000	**0.771**	**0.508**	**0.542**
wrdSIZE	0.51	0.51	-	-	-
phnSIZE	-	-	3.80	4.34	4.40

Table 5.5: Word, phone 1-best and lattice accuracies, STD accuracies (overall, IV and OOV) for different systems. PHN system is trained on LnoOOV, trigram LM and beam pruning $Pr = 175$ (to achieve comparable lattice size to WRDREDtoPHN system). The UBTWV for WRD and WRDRED systems is the Word UBTWV. The UBTWV for WRDtoPHN, WRDREDtoPHN and PHN systems is the Phone UBTWV.

We see the impact of out-of-vocabulary words on spoken term detection accuracy. We lose about 0.028 of the UBTWV accuracy on in-vocabulary words for LVCSR systems, which is caused by the OOVs. Similar trend is seen in WRDtoPHN systems (from 0.575 to 0.554). Important fact is that an STD system based on directly generated phone lattices PHN detects better the out-of-vocabulary terms. The PHN system has modified pruning to achieve comparable phone lattice size to the WRDREDtoPHN system. The conclusion is that **there is no need for using phone lattices generated by the LVCSR system for spoken term detection**. In-vocabulary terms which can be detected more accurately by WRDtoPHN system can be detected even more accurately by the LVCSR. Out-of-vocabulary terms, which can not be detected by the LVCSR are better to be detected in directly generated phone lattices than by a system converting word-to-phone lattices. The only advantage of WRDtoPHN system is the higher overall phone accuracy, which is not really interesting for the user.

Chapter 6

Subword recognition – phone multigrams

This chapter deals with theoretical description and experimental evaluation of multigram units. Recognizer used for multigrams is the same as is for the phones.

Examples of other subword units [Ng00b] besides phones are *syllables*, *phone n-grams*, *phone multigrams*, *broad phone classes*. All these units are based on phones. Phone recognition and search using such units has its advantages and drawbacks. The advantage of phone recognition (using simple phone loop) is its relative simplicity and presence of minimum constraints. Produced phone string precisely reflects a spoken word or term. This holds if the acoustic models are highly accurate and the word (or term) was uttered correctly. Then phone string produced by the phone recognizer perfectly matches the searched phonetic word form. But these two conditions are rarely fulfilled.

The drawbacks of phones are the following: If the model is not 100% accurate, the speaker does not pronounce well, or there is a background noise, recognized phones do not match the speech well. Also, decoding from free phone loop with higher order of n-gram language model is computationally more expensive than the decoding from LVCSR network[1]. Longer units should be more robust for incorrect pronunciation of a term too. Finally, phone n-grams with fixed length n must be used for indexing of phone strings or lattices. The optimal length of phone n-grams was found to be 3 in [Ng00b]. In our prior work [SFL+08], we have also used sequences of overlapped 3-grams for search. However, out-of-vocabulary words shorter than 3 phones were dropped.

The disadvantage of the fixed length sequences is that the frequencies of phone sequences are not taken into account. Some phone trigrams are more frequent than the others. Variable length sequences can be used to overcome this problem: a rare sequence is split into more frequent shorter sequences while a frequent sequence can be represented as the whole unit. The other advantage is that variable length phone units can reflect word sequences and compensate for missing word language model.

6.1 Definition of multigrams

Variable length sequences of phones are denoted as phone multigrams. The multigram language model was proposed by Deligne et al. [DB95]. Multigram model is a statistical

[1] All phone acoustic models must be evaluated in the phone loop approach, while words and word language model reduce the search space significantly. This leads to evaluation of limited set of phone acoustic models and lower computational requirement.

model having sequences with variable number of units. The definition of multigram model and its parameter estimation follows:

Let $\mathbf{w} = \{w_1, w_2, \ldots, w_N\}$ denote a string of N units, and let $\mathbf{s}$ denote a possible segmentation of $\mathbf{w}$ into q sequences $q \leq N$ of units $\mathbf{s} = \{s_1, s_2, \ldots, s_q\}$. The n-multigram model computes the joint likelihood $L(\mathbf{w}, \mathbf{s})$ of the corpus $\mathbf{w}$ associated to segmentation $\mathbf{s}$ as the product of the probabilities p of the successive sequences, each of them having a maximum length of n:

$$L(\mathbf{w}, \mathbf{s}) = \prod_{i=1}^{q} p(s_i) \tag{6.1}$$

Denoting as $\mathcal{S}$ the set of all possible segmentations of $\mathbf{w}$ into sequences of units, the likelihood of $\mathbf{w}$ is:

$$L_{mgr}^{best}(\mathbf{w}) = \max_{\mathbf{s} \in \mathcal{S}} L(\mathbf{w}, \mathbf{s}) \tag{6.2}$$

A n-multigram model is fully defined by a set of parameters $\mathcal{P}$ consisting of the probability of each unit sequence $s_i \in \mathcal{D}$ in a dictionary $\mathcal{D} = \{s_1, s_2, \ldots s_m\}$ that contains all the sequences which can be formed by combination of $1, 2, \ldots, n$ units:

$$\mathcal{P} = (p_i)_{i=1}^{m} \quad \text{where} \quad p_i = p(s_i) \quad \text{and} \quad \sum_{i=1}^{m} p_i = 1 \tag{6.3}$$

Maximum likelihood estimates of $\mathcal{P}$ can be computed through Viterbi algorithm iteratively. Let $\mathbf{s}^{*(k)}$ denote the most likely segmentation of $\mathbf{w}$ with given parameters $\mathcal{P}^k$ at iteration k:

$$\mathbf{s}^{*(k)} = \arg\max_{\mathbf{s} \in \mathcal{S}} L(\mathbf{s} | \mathbf{w}, \mathcal{P}^k) \tag{6.4}$$

According to [DB95], the re-estimation formula of i^{th} parameter (sequence) at iteration $k+1$ is intuitive:

$$\mathcal{P}_i^{k+1} = \frac{c(s_i, \mathbf{s}^{*(k)})}{c(\mathbf{s}^{*(k)})}, \tag{6.5}$$

where $c(s_i, \mathbf{s})$ is the number of occurrences of sequence s_i in segmentation $\mathbf{s}$ and $c(\mathbf{s})$ is the total number of sequences in $\mathbf{s}$.

The set of parameters $\mathcal{P}$ is initialized with the relative frequencies of all occurrences of units up to length n in the training corpus. To avoid overlearning, it is advantageous to discard low probable sequences: by setting $p_i = 0$ to all $c(s_i) \leq c_0$. The c_0 parameter is denoted as **multigram pruning parameter**. Sequences of length $n = 1$ are excluded from pruning to ensure that each sequence is segmentable. If a unit with length $n = 1$ has 0 occurrences in $\mathbf{s}$, then it's probability is set to a very low number.

The following definition of multigram model and parameter estimation using Maximum Likelihood estimation and Viterbi algorithm is adopted from [DB95]. The training algorithm can be both Viterbi or EM (Expectation-Maximization). The use of a Viterbi training instead of an EM training does not have a large impact on the performances according to [DB95]: here the Viterbi algorithm built a unit dictionary about 10% larger than EM algorithm and the perplexity of "Viterbi" units was about 1% relative higher for both training and test data. We decided to use Viterbi algorithm due to easier implementation and nearly the same perplexity.

The principle of building-up the multigram model is the following:

1. Initial statistics of all possible multigrams are collected on training corpora.
2. Probabilities $\mathcal{P}^1$ of units are calculated.
3. Training corpora is segmented $\mathbf{s}^{*(k)}$ according to parameters $\mathcal{P}^k$.
4. Multigram statistics are collected on segmented training corpora $\mathbf{s}^{*(k)}$.
5. Pruning of less frequent units is applied.
6. Probabilities $\mathcal{P}^{k+1}$ of units are calculated.
7. If no convergence, go to step 3, else finish.

When the set of parameters $\mathcal{P}$ is estimated, any phone string can be segmented into sequence of phone multigrams. The process of segmentation is straightforward. All possible segmentations, according to the inventory of phone multigrams, are created. Then, probability of each segmentation is evaluated according to the probabilities of multigram units. The best (most probable) segmentation is considered as the segmentation of given phone string by the set of phone multigrams. The process of phone string segmentation to phone multigrams is implemented also by the Viterbi algorithm.

6.2 Multigram training data

The phone multigrams are trained on phone strings. The training procedure has 2 steps. Multigram dictionary and unit probabilities are estimated in the first step. Standard n-gram language model is estimated in the second step. The multigram language model training data are the same as used for phone language models (*LnoOOV* and *LOOV*), see section 5.1.1. Due to size of *LOOV* and the iterative multigram training procedure, the data used for estimation of multigram dictionary is reduced to 3.75M phones to achieve reasonable training time (several hours). This corpus is denoted as *MOOV*. The corresponding corpus derived from *LnoOOV* is also created and denoted as *MnoOOV*. The sizes of above mentioned corpora are summarized in table 6.1.

Notation	# of utterances	# of phones (incl. sil)	# of phones (excl. sil)
ctstrain04[HBD$^+$05]	300.5k	11.21M	9.97M
LnoOOV	237.2k	6.40M	5.60M
LOOV	169.1k	6.26M	5.60M
MnoOOV	143.5k	3.82M	3.35M
MOOV	106.4k	3.75M	3.35M

Table 6.1: Comparison of corpora used for multigram dictionary (M*) and language model (L*) training.

Word or subword recognizer produces lattices where terms appear as sequences of words or subwords. However, the recognizer (*SVite* decoder) can be switched to produce phone lattices even if it is word/multigram recognizer. We can evaluate phone accuracy

and detection of search terms as phone strings even if such phone lattice is produced by an LVCSR system. Beside *Word UBTWV* and *Phone UBTWV*, we define *Mgram UBTWV* which means the UBTWV in case terms are searched in multigram form.

Our multigram system is evaluated by the following metrics:

- *PAC* Phone 1-best accuracy – The decoder is switched to produce phone lattices instead of multigram lattices. It is similar to phone lattices generated by the LVCSR described in section 5.2.

- *Mgram UBTWV* – Spoken term detection task is applied on multigram lattices. Terms are converted from word form to multigram form. Units in lattices are multigrams.

- *Phone UBTWV* – Spoken term detection task is applied on phone lattices generated by multigram system. Terms are converted from word form to phone form. Units in lattices are phones.

- *wrdSIZE / phnSIZE* Lattice size – Size of lattices is measured by the number of indexed tokens. See section 3.4.1 for more precise description. Multigram unigrams are indexed in case of multigram lattices (denoted as *wrdSIZE*), which is similar to indexing word lattices. Phone trigrams are indexed in case of phone lattices (denoted as *phnSIZE*).

Note, that searched terms must be converted from words to sequences of multigrams, too. Each term has appropriate phone sequence which is searched in phone lattices. This phone sequence is segmented into phone multigrams. The multigram sequence is only the most likely segmentation of given phone string in this thesis, otherwise it is stated. For more details about term conversion from words to multigrams see section 6.5. Produced sequence of multigram units is searched in multigram lattices.

6.3 Tuning of multigram parameters

Several parameters must be known to build the inventory of multigram phone sequences. The parameters are **maximal length of multigram** – l_{mgram} and **multigram pruning** – c_0. For the multigram-based subword system, we must also set the **order of n-gram language model** and decoding parameters as maximum active models MAM, beam pruning factor Pr, word insertion penalty and language model scaling factor. We have 7 parameters which must be tuned to find the best results. Some of these parameters depend on each other. The following procedure is used to find approximately the best setup.

The baseline multigram dictionary is estimated on *MOOV* corpus and the n-gram language model is trained on *LOOV* corpus. Multigram parameters are set to $l_{mgram} = 5$ and $c_0 = 2$. Order of n-gram language model over multigram units is set to 3. Similarly to previous experiments, word insertion penalty and language model scaling factor are tuned to the best accuracy.

1. MAM and Pr are tuned to find reasonably good results depending on the size of lattices in the first step. It is found that MAM and Pr have no effect on the phone accuracy. Both the Mgram UBTWV and Phone UBTWV saturate close to $Pr = 220$ and $MAM = 5000$. So these values are fixed for further experiments.

2. Different multigram pruning factors c_0 are tried in range between 2 and 200. Dependencies of PAC, Mgram UBTWV and Phone UBTWV are plotted in figure 6.1. Phone accuracy is not significantly (0.5%) affected by the pruning. PAC saturates for multigram pruning $c_0 = 20$. UBTWVs are affected more significantly, especially the Mgram UBTWV. Both UBTWVs saturate close to $c_0 = 50$. That is why the multigram pruning factor is chosen $c_0 = 50$ in the following experiments.

3. Maximal length of multigram units l_{mgram} is tested next. The accuracies are plotted in figure 6.2. We conclude that the PAC is not affected much and UBTWVs saturates for multigram length 5. The value $l_{mgram} = 5$ is fixed.

4. Finally, different orders of n-gram language model are used (see table 6.3 for details). The conclusion is that trigram language model brings usually the best accuracy.

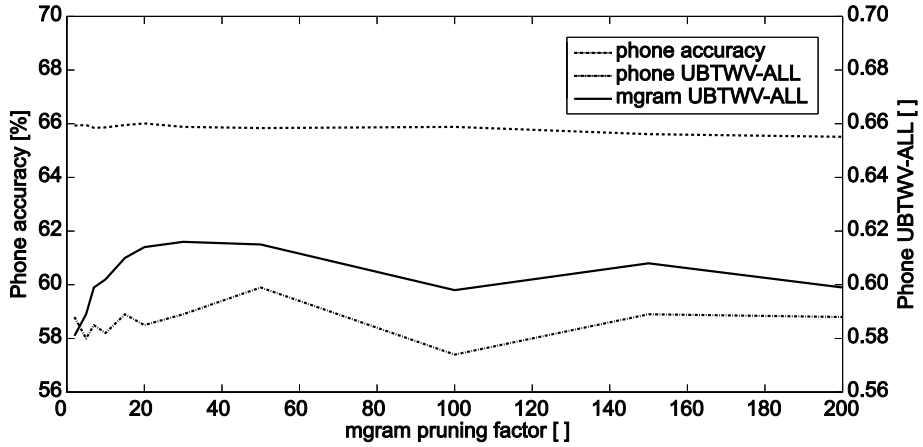


Figure 6.1: Comparison of phone accuracy (dashed), Mgram UBTWV-ALL (solid) and Phone UBTWV-ALL (dot-dashed) for multigrams of length $l_{mgram} = 5$ and different multigram pruning factors c_0.

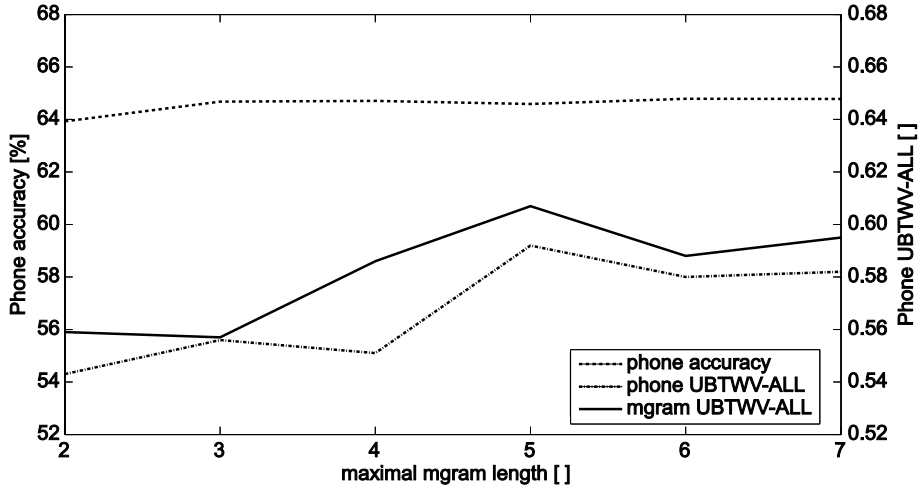


Figure 6.2: Comparison of phone accuracy (dashed), Mgram UBTWV-ALL (solid) and Phone UBTWV-ALL (dot-dashed) for multigrams of different length l_{mgram} and multigram pruning factor $c_0 = 50$.

word	sil YEAH I MEAN IT IS sil INTERESTING									
xwrd	sil-y-eh-ax		ay-m-iy-n		ih-t-ih-z-sil			ih-n-t-ax-r	eh-s-t-ih-ng	
nosil	sil	y-eh-ax	ay-m-iy-n		ih-t-ih-z		sil	ih-n-t-ax-r	eh-s-t-ih-ng	
noxwrd	*sil*	*y-eh-ax*	*ay*	*m-iy-n*	*ih-t*	*ih-z*	*sil*	*ih-n	t-ax-r-eh-s	t-ih-ng*

Table 6.2: Examples of different multigram segmentations. The first line is word transcript. The second line is unconstrained multigram segmentation. The third line is constrained multigram segmentation where silence is forbidden inside a multigram unit. The fourth line is constrained multigram segmentation where silence and word boundary `*` are forbidden inside a multigram unit.

6.4 Constrained multigram units

The baseline process of building multigram unit inventory is without any constraints (denoted **xwrd**). The corpus of phone strings is taken as is. An example of an utterance segmented by such unconstrained units is in table 6.2 line 2. A multigram unit can be placed across word boundaries and also across silences (`sil`). Incorporation of word boundaries (cross-word multigrams) into multigram units means, that multigrams also somehow reflect the word language model. The question is whether this is good or not. The same question can be asked about the silence `sil`. By incorporating silence into multigrams, the units are learned to remember parts of speech where silence is usual and where it is not. Two experiments with constrained training of multigram inventory are done to evaluate the influence of cross word multigrams and silence inside multigram units:

6.4.1 No silence in multigram

Inventories of multigram units which do not contain silence are trained in this experiment (denoted **nosil**). The unigram `sil` is the only one multigram unit which contains silence. This is needed to make utterances segmentable. An example of utterance segmented by this method is in table 6.2 line 3. Building of this nosil multigram inventory is done by a modification in the first step of multigram training procedure. After the statistics of all n-grams appearing in the training corpus are collected, all n-gram units containing `sil` are omitted (except the unigram `sil`). Then, the initial probabilities of units are re-normalized and the iterative training algorithm is run.

6.4.2 Non-cross-word multigrams

In this experiment, word boundaries are marked in the training corpus, and the following rule is incorporated into the training algorithm: word boundary will appear at most at the beginning or at the end of a multigram unit. Only two units with the word boundary marker can be put besides each other during the segmentation. If the first unit contains word boundary marker at the end, then the following boundary must contain the word boundary marker at the beginning. This system is denoted as **noxwrd**. An example of utterance segmented by noxwrd multigrams is in table 6.2 line 4. The word boundary marker is denoted by a star symbol.

6.4.3 Results

The multigram parameters for the following experiments are: maximum multigram length $l_{mgram} = 5$, multigram pruning $c_0 = 50$, multigram training corpus *MnoOOV* and language model training corpus *LnoOOV*. The decoder parameters are the same as for the previous multigram experiments.

The results comparing multigrams trained with different constraints are summarized in table 6.3. Three different orders of n-gram language model are trained for each of these three systems. The *"Phone"* column contains results when the multigram decoder is switched to produce phone lattices.

System	LM	Phone				Multigram	
	n-gram	PAC	PLAC	UBTWV-ALL	phnSIZE	UBTWV-ALL	wrdSIZE
xwrd	1	60.33	95.91	0.519	20.0	0.537	3.5
xwrd	2	63.13	95.54	0.533	9.4	**0.568**	2.0
xwrd	3	**65.25**	95.32	**0.547**	3.6	0.559	1.4
nosil	1	60.38	96.06	0.531	21.6	0.546	3.2
nosil	2	63.25	95.61	**0.551**	9.3	**0.592**	1.8
nosil	3	**65.42**	95.41	0.547	4.1	0.584	1.2
noxwrd	1	59.30	95.84	0.501	20.3	0.535	7.3
noxwrd	2	62.87	95.44	0.528	7.6	0.613	3.1
noxwrd	3	**65.10**	95.19	**0.541**	3.7	**0.630**	1.7

Table 6.3: Comparison of accuracies of xwrd, nosil and noxwrd multigram systems. All systems have maximal multigram length $l_{mgram} = 5$, multigram pruning $c_0 = 50$ and are trained on *MnoOOV* and *LnoOOV* corpora.

We conclude that multigrams perform better on the "multigram" level than on the phone level in the STD task. The best accuracy is achieved by noxwrd trained multigrams with trigram language model. The best phone accuracy is achieved by nosil trained multigrams and trigram language model. The constrains during the training have no significant effect on the phone accuracy. On the other hand, the nosil units perform slightly better. Significant impact is on the STD task, where the improvement of Mgram UBTWV-ALL is about 0.06 (10% relative).

Comparison of multigram systems trained with and without out-of-vocabulary words is in table 6.4. Excluding OOVs from training (which is the real scenario) causes a hit of 0.8% relative on the phone accuracy and up to 0.044 on the Mgram UBTWV-ALL.

System	trained on	Phone		Multigram
		PAC	UBTWV-ALL	UBTWV-ALL
xwrd	OOV	64.32	0.561	0.566
nosil	OOV	*65.84*	0.576	0.603
noxwrd	OOV	65.50	0.556	**0.674**
xwrd	noOOV	65.25	0.547	0.559
nosil	noOOV	*65.42*	0.547	0.584
noxwrd	noOOV	65.10	0.541	**0.630**

Table 6.4: Comparison of accuracy of xwrd, nosil and noxwrd multigram systems trained on corpora including (*MOOV*, *LOOV*) and excluding (*MnoOOV*, *LnoOOV*) the OOVs. All systems have maximal multigram length $l_{mgram} = 5$, multigram pruning $c_0 = 50$ and trigram language model.

6.5 Segmentation of searched terms

Terms must be converted from words to multigram sequences for multigram based STD. Only the most likely segmentation was used in all previous experiments. The following experiment evaluates the dependency of STD accuracy on the number of multigram segmentations of terms. The terms are converted from words to phones. These phone strings are next segmented to multigram sequences which are searched in multigram lattices. We modify the segmenting algorithm to produce n-best segmentations of terms' phone strings.

The multigram system with maximal multigram length $l_{mgram} = 5$, multigram pruning $c_0 = 50$ and noxwrd constraint is used for the evaluation. We also evaluate the dependency on the order of language model and training corpora (with OOVs or without OOVs). We generate $1, 2, 3, 5, 10$ and 20 best segmentations of the terms. The Mgram UBTWV-ALL are shown in figure 6.3.

Corpora	MOOV-LOOV			MnoOOV-LnoOOV				
LM order	1-gram	2-gram	3-gram	1-gram	2-gram	3-gram		
	UBTWV-ALL			UBTWV-ALL			IV	OOV
1-best	0.524	0.634	0.674	0.535	0.613	0.630	0.647	**0.593**
3-best	0.525	0.638	0.675	0.536	0.618	0.644	0.648	**0.636**

Table 6.5: Comparison of Mgram UBTWV-ALL for 1-best and 3-best segmentation of searched terms. The multigram system is trained on corpora including (*MOOV*, *LOOV*) and excluding (*MnoOOV*, *LnoOOV*) the OOVs. 3 different orders of n-gram language model are used. The multigram inventory has maximal length of $l_{mgram} = 5$, multigram pruning $c_0 = 50$ and noxwrd constraint. The Mgram UBTWV-IV and UBTWV-OOV is shown for 3-gram LM and "noOOV" trained multigrams.

We conclude that the Mgram UBTWV-ALL accuracy saturates for 3-best segmentations of the terms. The exact values are summarized in table 6.5. The largest improvement (0.014) caused by the n-best segmentation is reached for the system trained on corpora without OOVs (MnoOOV-LnoOOV) and trigram language model. The improvement is caused by improvement on OOV terms, Mgram UBTWV-OOV improved by 0.043. No significant improvement due to the number of segmentations is observed for systems trained

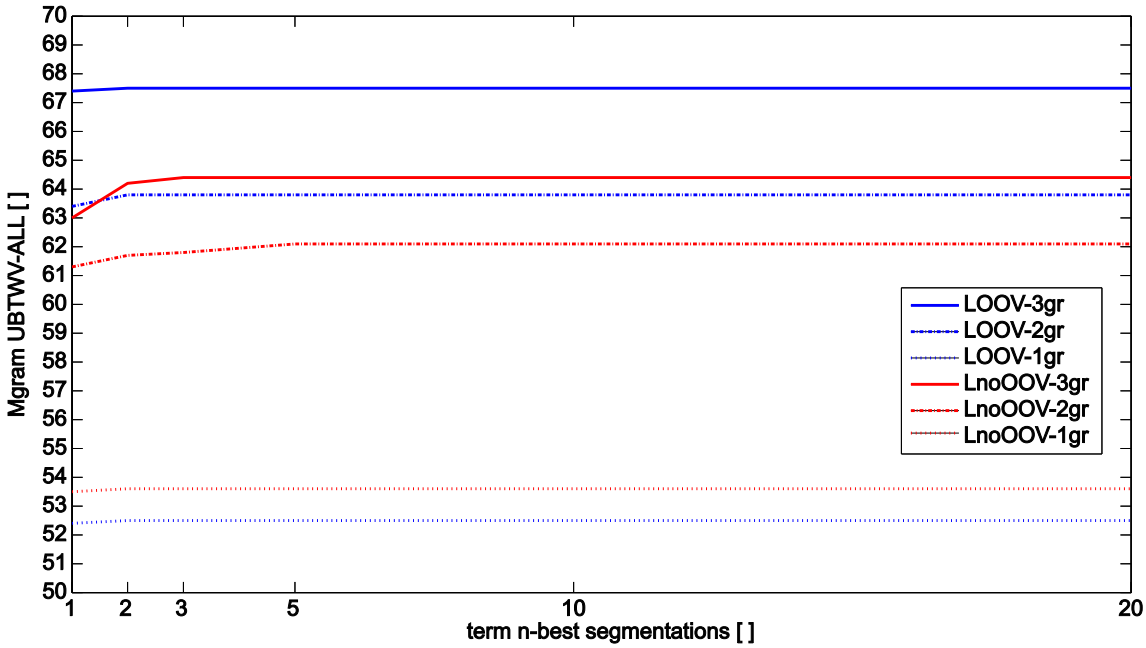


Figure 6.3: Comparison of Mgram UBTWV-ALL depending on n-best segmentations of searched terms for different multigram systems. The multigram systems are trained on corpora including (*MOOV*, *LOOV*) and excluding (*MnoOOV*, *LnoOOV*) the OOVs. 3 different orders of n-gram language models are used. The multigram inventory has maximal length of $l_{mgram} = 5$, multigram pruning $c_0 = 50$ and noxwrd constraint.

on OOVs. We do not see any improvement of accuracy on IV or OOV subsets. The search for the 3-best variants is slower on the other hand. Each term must be searched 3 times (without any optimization). That is why we concluded that searching for the 1-best segmentation is optimal for further experiments.

6.5.1 Comparison of confidence estimations for multigram system

In this section, we compare confidence estimation methods for multigram system in similar manner as we did in section 4.4 for word system. The confidence measures are compared for noxwrd system with trigram language model and decoding beam pruning lowered to $Pr = 300$. We lower the pruning to achieve larger lattices for better posterior probability and confidence estimation.

Evaluation metric	Mgram TWV			Mgram UBTWV		
Confidence measure	ALL	IV	OOV	ALL	IV	OOV
LP	0.178	0.234	0.294	0.658	0.667	0.637
SOLP	0.228	0.325	0.301	0.690	0.712	0.641
SCOLP	0.148	0.194	0.297	0.651	0.658	0.637
C_{max}	0.224	0.319	0.302	0.696	0.719	0.641

Table 6.6: Comparison of different term confidences and its influence on the Mgram TWV and UBTWV scores for the multigram noxwrd system with trigram LM. The multigram inventory has maximal length $l_{mgram} = 5$, multigram pruning $c_0 = 50$. Beam pruning is set to $Pr = 300$.

Results summarized in table 6.6 show, that the best accuracy is achieved again (see section 4.4) by SOLP and C_{max} confidence measures. On contrary to the word system, the confidence measures have significant impact on UBTWV. The difference between LP and SOLP is 0.032. It is interesting to note, that more "correct" confidence estimation has more significant influence on short terms. The IV subset contains large portion of one word terms in comparison to OOV subset. This is probably caused by the fact, that subword system produces more overlapped hypothesis of each term than the word system. In case of SOLP, these are taken into account of term confidence and improve the accuracy. Similar trend is seen also for multigram TWV.

6.6 Conclusion

Table 6.7 compares *word*, *phone* and *phone multigram* based systems from *phone* and *spoken term detection* accuracy point of view. The WRDREDtoPHN is the WRDRED LVCSR switched to produce phone lattices. The best phone accuracy is achieved by the multigram nosil constrained system. However, better STD accuracy is achieved by the noxwrd constrained multigrams. It is important to mention that multigram lattices are significantly smaller and the recognition network is approximately of the same size compared to phone system. The multigram system has maximal multigram length $l_{mgram} = 5$ and multigram pruning $c_0 = 50$. The terms are segmented only to 1-best multigram variant.

Unit	System	LM	PAC	UBTWV			SIZE
		n-gram		ALL	IV	OOV	
Word	WRDRED	2	-	0.514	0.734	0.000	0.56w
Word	WRDREDtoPHN	2	65.40	0.540	0.554	0.508	4.34p
Phone	LnoOOV	3	59.66	0.483	0.453	0.552	6.38p
Mgram	xwrd	3	65.25	0.559	0.552	0.577	1.4w/3.6p
Mgram	nosil	3	**65.42**	0.584	0.578	**0.597**	1.2w/4.1p
Mgram	noxwrd	3	65.10	**0.630**	**0.647**	0.593	1.7w/3.7p

Table 6.7: Comparison of word, phone and multigram systems from phone accuracy, lattice size and Word, Mgram and Phone UBTWVpoint of view. 0.56w means wrdSIZE and 4.34p means phnSIZE.

Chapter 7

Combined word-subword spoken term detection

We investigate into the use of different combination of word and subword STD systems. Let us have a term `"Igor Szöke"`. The term is first split into in-vocabulary (IV) and out-of-vocabulary (OOV) parts. Let us assume, that the name `Igor` is in-vocabulary word `igor` and the surname `Szöke` is an out-of-vocabulary word. If we choose phones as the subword units, the out-of-vocabulary part is decomposed into sequence of phones `s eh k eh`. The combination of a word and subword based spoken term detection is needed to spot both, in-vocabulary and out-of-vocabulary parts of the term.

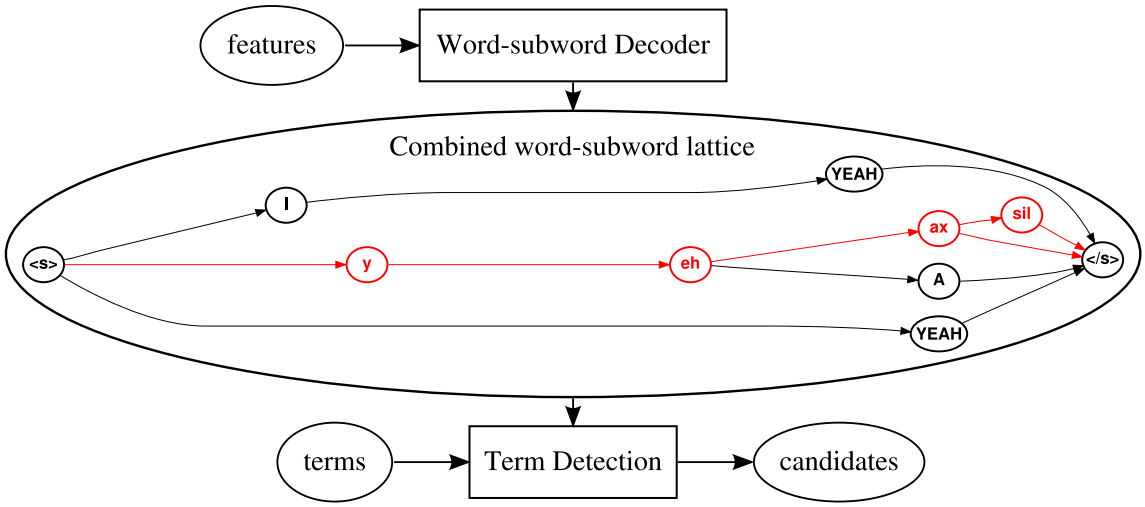


Figure 7.1: Scheme of word-subword recognition. The decoder produces hybrid lattice which is searched for the terms.

The combination of word and subword STD can be done in two ways. The *first way* of word-subword combination is a **hybrid word-subword decoding**[1], where the combined word-subword recognition is done directly during the decoding. A scheme of the word-subword combination during the recognition (decoding) is in figure 7.1. The output of the recognizer is a hybrid word-subword lattice, which is searched for terms [Baz02, YS04a]. An example of a hybrid word-subword lattice is in figure 7.2.

[1] Denoted as the prior combination in [YS04b].

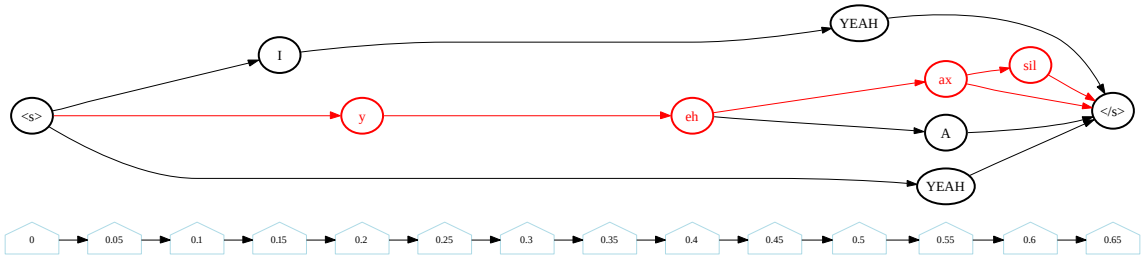


Figure 7.2: Example of word-subword lattice.

The *second way* is **combination after the decoding**[2]. The "after the decoding" combination can be done in several approaches. One possible approach is the combination in the **lattice level**. The word and subword lattices are generated separately and, then combined together [YS04b] to a hybrid lattice (scheme in figure 7.3).

In both *hybrid word-subword decoding* and *lattice level combination* approaches, in-vocabulary (IV) and out-of-vocabulary (OOV) terms are directly searched in the lattices. In our example, the term `"igor s eh k eh"` is directly searched in the hybrid lattice. The drawback of the lattice level combination after the decoding (reported by Yu et. al. [YS04b]) is in the need to balance scores between words and subwords. This can happen particularly in case of largely different systems (GMM-based word recognizer and neural net-based subword recognizer).

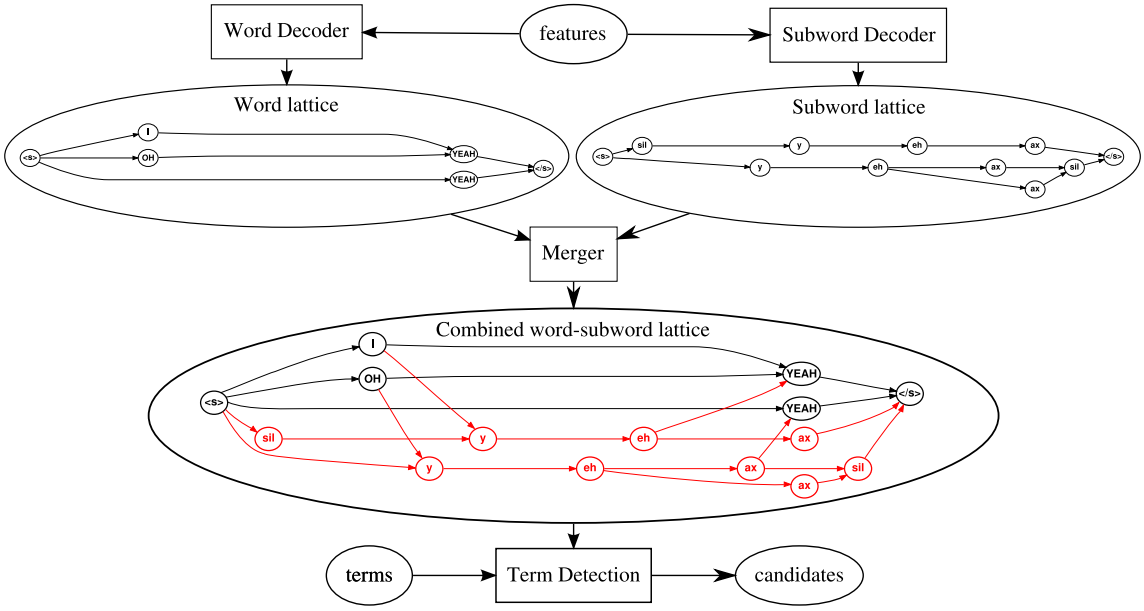


Figure 7.3: Scheme of combination after the decoding on the lattice level, where word and subword lattices are generated separately and then combined into a hybrid word-subword lattice. The hybrid lattice is then searched for the terms.

The *combination after the decoding* may be done later, on the **search level**. The term's word and subword parts are search separately in the appropriate lattice (scheme in figure 7.4). We used this approach in NIST STD 2006 evaluations [ČSF+07]. The IV part of term (`igor`) is searched in word lattice and the OOV part of term (`s eh k eh`)

[2]Denoted as the posterior combination in [YS04b].

is searched in the subword lattice. Last, the candidates are combined. Only pairs of candidates where word and subword parts satisfy time constrains (`s eh k eh` directly follows `igor`) are returned as the term detection. The drawback of this approach is that we need two standalone systems.

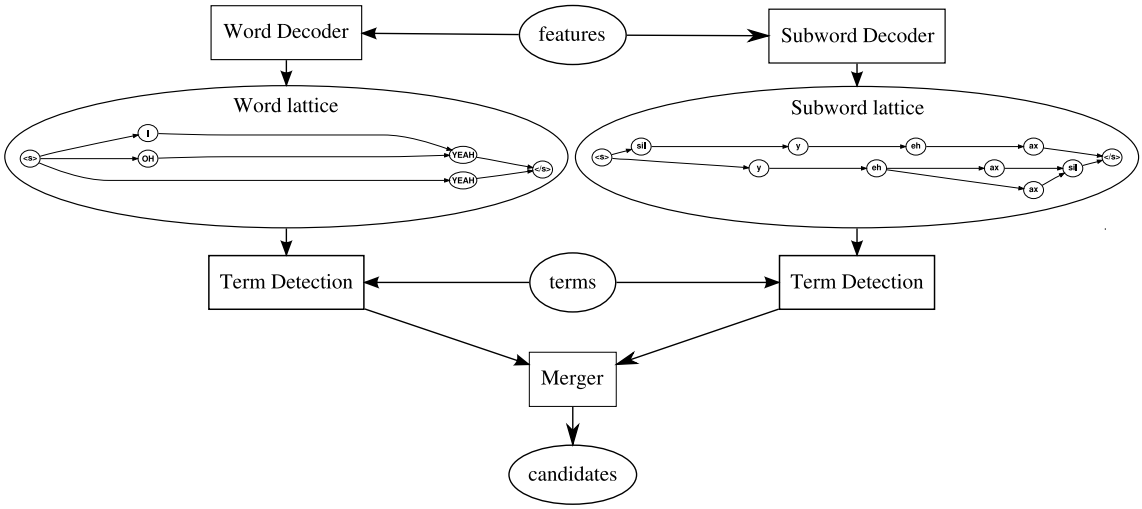


Figure 7.4: Scheme of combination after the decoding on the level of term detection, where word and subword lattices are generated separately. The term is split into IV and OOV parts and these are searched separately. Finally, the IV and OOV part candidates are combined.

7.1 Combined word-subword recognition via hybrid recognition network

Doing the combination of word and subword STD during the recognition (decoding) is the most straightforward approach. A hybrid word-subword language model is the only thing which is needed for the decoding.

The *word recognizer* is considered as a strong recognizer. It has strong acoustic model (word models) and language model (word bigrams). The *subword recognizer* is considered as a weak recognizer. It has weak acoustic model (phone or phone multigram units) and relatively weak language model (phone n-grams).

The combination of word and subword recognizer should allow to traverse between words and subwords in any time. If traversing penalties and other parameters are set correctly, the word part of the recognizer should well represent in-vocabulary speech. Out-of-vocabulary parts of speech may be highly unlikely for the strong word recognizer. However these OOV parts are not so unlikely for the subword part of the recognizer. This leads the recognizer to switch from the word part to the subword part. The result is the hybrid word-subword lattice where OOV parts of speech are represented by phone sequences and IV parts of speech by word sequences.

We decided to use an approach similar to [Baz02]. Bazzi investigated the **out-of-vocabulary detection**[3], its impacts on word recognition and word confidence score. He used hybrid word-subword recognition network. The word language model (network) contained a symbol substituting an OOV word. The OOV word was modeled by a phone sequence (bigram phone LM). He also proposed variable length phone sequences (multigrams), which brought significant improvement of OOV detection accuracy. This work was not aimed at using/post-processing the output of OOV (subword) model. Only phone accuracy of recognized phone sequences was evaluated.

On contrary to Bazzi [Baz02], we aim at the *investigation of STD accuracy* and the *practical application* for searching in spoken documents. We fully use the information produced by the OOV (subword) model for search of the OOV terms. We evaluate the accuracy of STD and word accuracy. We investigate in more depth "which subword model should be the best":

- The impact of subword model and hybrid network scaling parameters to the accuracy.
- The speed of the system and size of the index.
- Search for the system configuration suitable for practical use.
- Evaluation of the hybrid system in conjunction with indexing and search engine for spoken term detection.

7.1.1 Building combined word-subword hybrid recognition network

We use the same decoder (*SVite* from STK toolkit) as is used for the baseline experiments in word and phone recognition (chapters 4, 5, 6). Because the *SVite* is a static decoder, the hybrid decoding is possible by **modification of recognition network** (figure 7.5). No other changes are needed in the decoder.

Brief description of how a word recognition network is built is given in section 4.2. The hybrid word-subword recognition network is built in similar way. Only the language model automaton G and the lexicon L are modified in the composition (equation 4.3 in section 4.2):

$$H \circ C \circ L \circ G, \tag{7.1}$$

The word language model represented by WFSA G is created as open vocabulary language model and contains an `<unk>` symbol. The `<unk>` symbol represents any out-of-vocabulary word, see figure 7.6. The new open-vocabulary language model represented by WFSA in denoted as G_{word}. This `<unk>` symbol is substituted by a subword language model (figure 7.7). The subword language model is converted to WFST $G_{subword}$. The hybrid "language model" is created by composition of word and subword language models

$$G_{subword} \circ G_{word}. \tag{7.2}$$

The substitution is illustrated in figure 7.8. The red part of network is the `<unk>` in figure 7.6 substituted by the subword model in figure 7.7.

The word dictionary L mapping words to phones is joint with the subword dictionary mapping subword labels to phones. Then this dictionary is converted to WFST representing the hybrid lexicon. Modified composition of the hybrid recognition network is written as:

$$H \circ C \circ (L_{word} \cup L_{subword}) \circ G_{subword} \circ G_{word}, \tag{7.3}$$

[3]The OOV detection aims at detection of parts of speech containing out-of-vocabulary words. But it usually does not deal with the content (what was the OOV word).

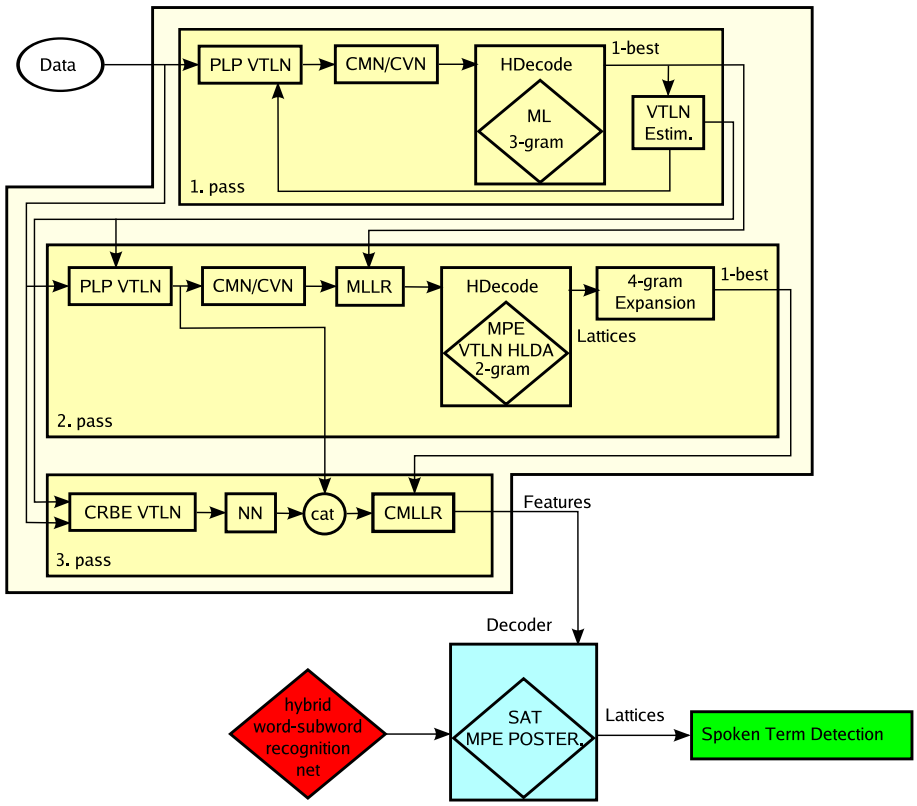


Figure 7.5: Schema of 3-pass recognition system used for hybrid word-subword lattice production.

where H represents the HMM (tied-list) and C represents the mapping from context-dependent to context-independent phonetic units, L_{word} is the pronunciation dictionary mapping phones to words, $L_{subword}$ maps phones to subword units (eg. syllables, multi-grams or phones). $G_{subword}$ is a weighted transducer created from the subword language model and G_{word} represents the word language model (weighted acceptor).

The `<unk>` and `<silsp>` nodes in the hybrid network (figure 7.8) produce an output label. The `<unk>` node produces symbol `<unk>` which is used as a marker of the beginning of subword section in the output. The `<silsp>` node produces symbol `<silsp>` which is used as a marker of the end of subword section in the output and also represents a *sil/sp* model[4].

Parameters such as *word insertion penalty* and *acoustic or language model scaling factors* are tuned to control the recognition accuracy and output of the LVCSR system. However, the hybrid network is considered as one object by the *SVite* decoder. The same penalty and scaling factor apply to both word and subword parts. That is why three different parameters are incorporated into the combined network during its building. The first parameter is **subword language model scaling factor** *SLMSF*. This parameter exponentiates the likelihood[5] assigned to the subword LM transitions. The second parameter is **subword word insertion penalty** *SWIP*. It is a constant which multiplies each transition's likelihood[6] value leading to a word node. The last parameter is **subword cost**

[4]The *silence/short pause* is attached to each word model by default in each word network. This is used for modeling of possible silences following the words.

[5]In practice, log likelihoods are used. So the *SLMSF* multiplies the log likelihoods of subword part.

[6]In practice, log likelihoods are used. So the *SWIP* is added to the log likelihoods of subword part.

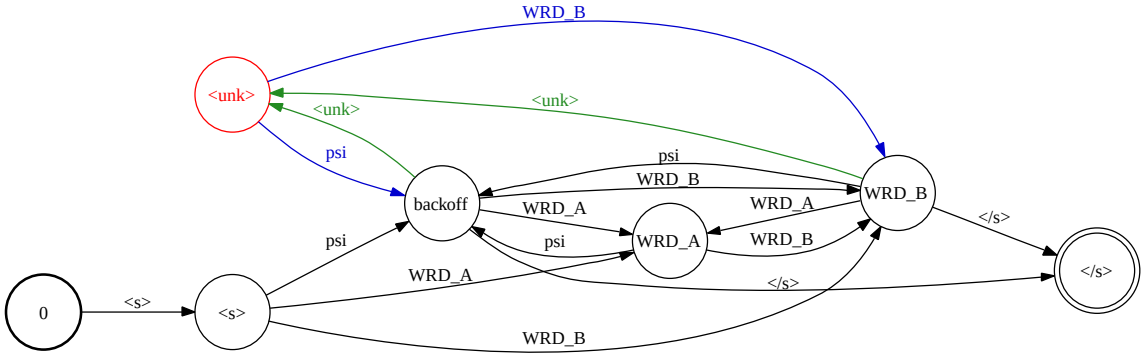


Figure 7.6: Example of open vocabulary language model. The `<unk>` states for the out-of-vocabulary words.

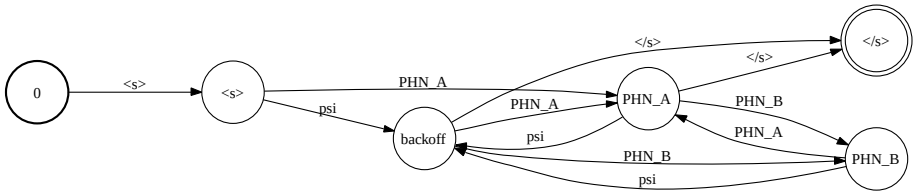


Figure 7.7: Example of a subword (phone) language model.

SC. It is a constant which multiplies the `<unk>` symbol likelihood[7] and represents a cost of going to the whole subword model. Figure 7.9 illustrates example of the above mentioned parameters.

Important note: From practical point of view, the *SLMSF* scales the probabilities of subword LM to corresponds to word LM, while *SWIP* modifies word insertion penalty set in the decoder to be applicable also in the subword part. "Word" insertion penalty can differ largely for systems with different type of units (words, phones or multigrams). *SLMSF* and LM are factors which should scale and shift the subword LM to have comparable probabilities to the word LM. The *SC* penalizes only entering the subword model. This can be understood as a factor which modifies the prior probability of observation of OOV word (represented by `<unk>` symbol in the word LM).

The LM log probabilities (negative values) are represented as positive values in the WFST. But we did not change sign of *SWIP* in our implementation, so the meaning of *SWIP* value is reversed. This constant is subtracted in the decoder. So, if the reported *SWIP* value is negative (-1.6), it means that positive value ($+1.6$) is added to accumulated log likelihood in each token passing through the subword unit label.

7.2 Baseline systems

In comparison to previous experiments with word recognizer (chapter 4), we modify the language model and the dictionary. The language model must contain the OOV symbol for correct modeling of speech with possible OOVs. We took several corpora (section 4.1.4) used for language model estimation (section 4.1.3) and trained two new open vocabulary language models:

- The first one is used as reference baseline LM (denoted **WRDunk**). It is an open

[7] In practice, log likelihoods are used. So the *SC* is added to the `<unk>` symbol log likelihood.

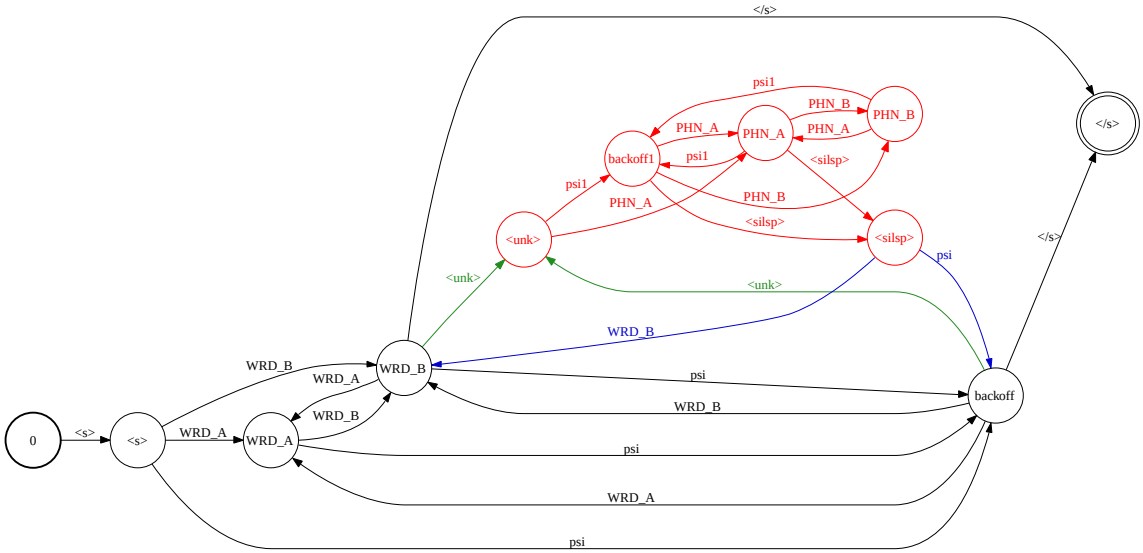


Figure 7.8: Example of hybrid word-subword language model.

vocabulary LM with full 50k words vocabulary. The `<unk>` symbol represents all words appearing in the corpora beyond the 50k vocabulary.

- Then the vocabulary is reduced by the 880 words (see section 3.1.1). The second open vocabulary LM is estimated with the reduced vocabulary (denoted **WRDREDunk**). So the OOV set of words is enlarged by the omitted 880 words.

The baseline experiments are summarized in table 7.1. The beam pruning is set to $Pr = 220$ and maximum active models is set to $MAM = 5000$.

System	WAC	Word UBTWV			wrdSIZE
		ALL	IV	OOV	
WRDforHYB	69.04%	0.738	0.734	0.746	0.252
WRDunk	69.20%	0.724	0.727	0.715	0.190
WRDREDunk	66.50%	0.486	0.694	0.000	0.190

Table 7.1: Comparison of closed vocabulary baseline word recognizer *WRDforHYB* with full *WRDunk* and reduced *WRDREDunk* open vocabulary language model. OOVs caused significant deterioration of the accuracy in the *WRDREDunk* system.

Obviously both word and STD accuracy dropped. Dropping of STD accuracy is caused by the out-of-vocabulary words, which deteriorate (-0.033) also the STD accuracy of the in-vocabulary words. Negative influence of OOVs has significant impact of -2.7% on WAC.

Besides the baseline system for word accuracy and in-vocabulary term detection, we need also baseline system for out-of-vocabulary term detection. The best accuracy (UBTWV-OOV) on the OOV terms was achieved by phone multigram based systems (see section 6.6). Multigram systems giving the best Mgram UBTWV-OOV accuracy proposed in section 6.4 are summarized in table 7.2. The best Mgram UBTWV-OOV accuracy is achieved by unigram language model for all three systems. The UBTWV-OOV accuracy is similar across different constrained multigram system. According to UBTWV-OOV and

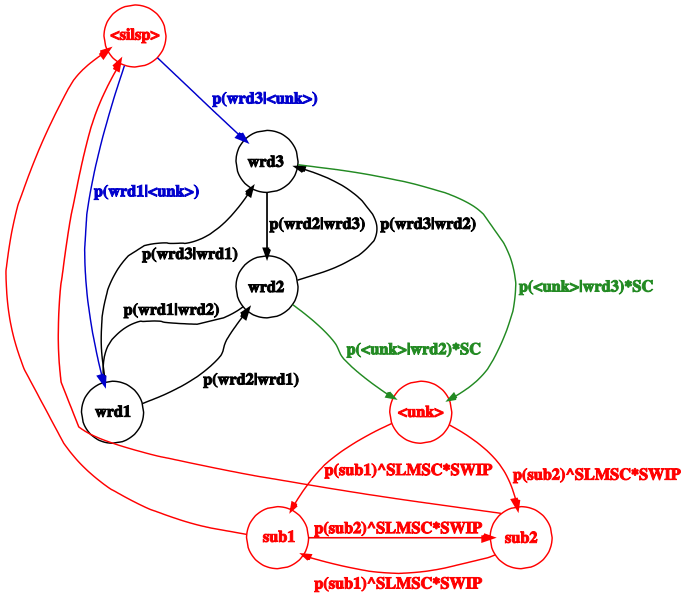


Figure 7.9: Example of parameters used for scaling the subword part in the hybrid network.

lattice size, we decided to use the xwrd multigram system with unigram language model as baseline system for OOV term detection.

System	LM order	Mgram UBTWV			wrdSIZE
		ALL	IV	OOV	
xwrd	1	**0.537**	**0.492**	**0.642**	**3.5**
nosil	1	0.546	0.508	0.636	3.2
noxwrd	1	0.535	0.492	0.637	7.3

Table 7.2: Comparison of multigram based systems giving the best UBTWV-OOV. The multigram inventory is estimated on *MnoOOV* and the unigram language model is trained on *LnoOOV* (see section 6.2 for more details).

7.3 Oracle hybrid system

An oracle hybrid word-subword system is built to estimate hybrid system upper-bound accuracy. The goal is to estimate deterioration of the accuracy (compared with WRDunk system) by using the (`<unk>`) symbol for modeling the "out-of-vocabulary" parts of speech. The subword part is replaced by the 880 OOV words with unigram language model. The unigram probabilities are taken from the WRDunk language model.

Two different systems are built for the estimation of upper-bound word accuracy. An example of 1-best output of these two systems is in table 7.3 (the word `EXAMPLE` is chosen as the OOV word).

- The **HybridOracle** system produces word transcripts without marking any of OOV parts. The word accuracy of *HybridOracle* shows the upper-bound word accuracy deterioration (0.48% absolutely against baseline WRDunk system, see table 7.4)

System	1-best output example
WRDunk	THIS IS AN EXAMPLE OF RECOGNIZER OUTPUT
WRDREDunk	THIS IS AMEX APPLE OF RECOGNIZER OUTPUT
HybridOracle	THIS IS AN EXAMPLE OF RECOGNIZER OUTPUT
HybridOracleunk	THIS IS AN <unk> OF RECOGNIZER OUTPUT

Table 7.3: Example of the 1-best output of the standard and hybrid systems. The word EXAMPLE is set as OOV (omitted from vocabulary) in WRDREDunk and added into the OOV model in HybridOracle and HybridOracleunk systems. The HybridOracle system recognizes and outputs EXAMPLE, the HybridOracleunk system recognizes EXAMPLE but outputs <unk>.

when the OOV words are modeled by <unk> and unigram probability in comparison to standard "in-vocabulary" bigram modeling.

- The **HybridOracleunk** system produces word transcripts with marking the OOV parts by the <unk> symbol. So each time the decoder passed through OOV subpart, only the <unk> is written out, although the OOV subpart contains the 880 OOVs. This should lead to one substitution error in the evaluation of word accuracy. The word accuracy of HybridOracleunk shows the upper-bound word accuracy improvement (1.07% absolutely against WRDREDunk, see table 7.4) when the OOV words are modeled.

The optimal scaling parameters must be estimated for subword (OOV model) part of the hybrid system. All three parameters (*SLMSF*, *SWIP* and *SC*) are tuned separately. The *SWIP* and *SLMSF* have the greatest influence on the word accuracy. The word accuracy (WAC) depending on the *SLMSF*, *SWIP* and *SC* parameters is plotted in figure 7.10 for the oracle systems. The Word UBTWV depending the *SLMSF*, *SWIP* and *SC* parameters is plotted for in-vocabulary words in figure 7.11. The conclusion is that the *SWIP* and *SLMSF* parameters have the greatest influence.

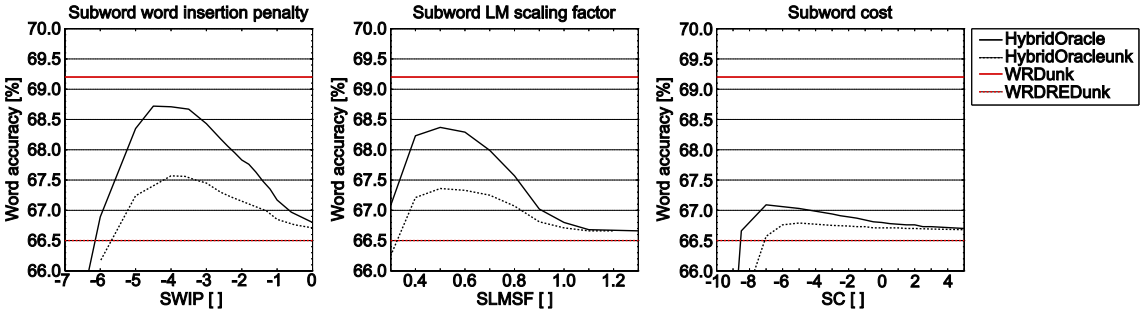


Figure 7.10: Dependency of the HybridOracle (solid black) and HybridOracleunk (dashed black) systems word accuracy on the *SWIP*, *SLMSF* and *SC* parameters.

After tuning *SLMSF*, *SWIP* and *SC* parameters separately, these parameters are tuned together to found the best accuracy. However, we find that this has no significant impact on the accuracy. The word accuracy improved by 0.05% and the Word UBTWV-IV improved by 0.005. The Word UBTWV-IV improvement can be seen significant, but the lattice size wrdSIZE increased by 10%, which negates the improvement. We decided to tune only the *SWIP* parameter for further experiments. This gives nearly the same results, while the system tuning for the following experiments is simpler. In table 7.4, the best word and

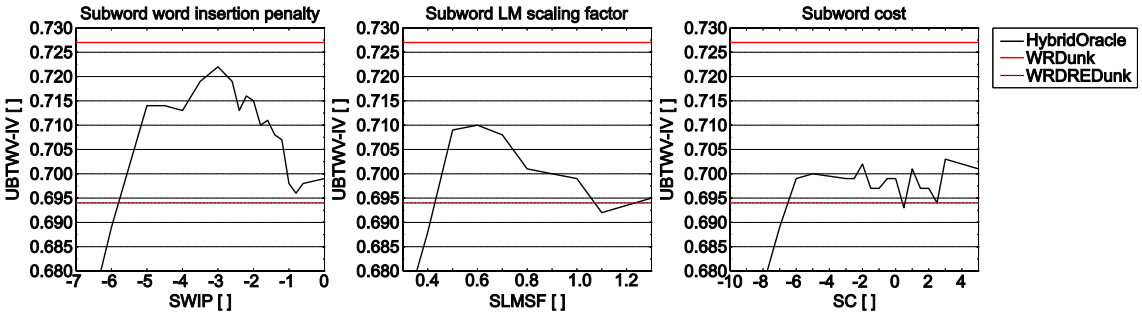


Figure 7.11: Dependency of the HybridOracle systems Word UBTWV-IV accuracy on the *SWIP*, *SLMSF* and *SC* parameters.

UBTWV-IV accuracies achieved by tuning all three parameters (marked by blue) and only the *SWIP* parameter (marked by red) are shown.

System	WAC	Word UBTWV IV	*SLMSF*	*SWIP*	*SC*	wrdSIZE
WRDunk	69.20%	0.727	-	-	-	0.190
WRDREDunk	66.50%	0.694	-	-	-	0.190
HybridOracle	**68.72%**	0.714	1.0	4.5	0.0	0.216
HybridOracleunk	**67.57%**	0.719	1.0	4.0	0.0	0.196
HybridOracle	68.43%	**0.722**	1.0	3.0	0.0	0.202
HybridOracle	**68.77%**	0.718	1.1	5.0	0.0	0.215
HybridOracle	68.53%	**0.727**	1.0	4.5	1.0	0.228

Table 7.4: Comparison of accuracies of baseline word recognizers with full and reduced open vocabulary language model, and hybrid oracle recognizers on different settings of tuned parameters *SLMSF*, *SWIP* and *SC*. Value in bold means that this accuracy is maximized by tuning of parameters. Systems where only *SWIP* parameter was tuned are marked in color red. Systems where all three parameters were tuned are marked in color blue.

The HybridOracle system suggests an easy way of inserting new words into the recognizer. A minimalistic LVCSR based spoken term detector can be built on small LVCSR ($5k$ words for example). The searched terms which are OOV can be added on system start-up into the recognition network in the same way as is in the HybridOracle. This can not be used in document indexing and search. However, it can be used in the security domain, where lots of data is searched once or few times and the processing time must be fast. This can be also an interesting alternative to the acoustic keyword spotting.

7.3.1 Accuracy depending on the lattice size

The saturation of accuracy is visualized for certain values of the parameters for the HybridOracle system in figure 7.10. But as the "weight" of the OOV part of the network changes, the lattice size varies. Because the size of the lattice is one of important parameter in the spoken term detection task, we analyze the accuracy related to the lattice size. Note however, that lattice size and accuracy depend on system parameters (beam pruning, penalties, scaling parameters, *SLMSF*, *SWIP*, *SC* etc.).

In this section, the independent variables are *SLMSF*, *SWIP* or *SC* parameters for the hybrid systems (HybridOracle). Other parameters are fixed. Pruning is fixed to the values $Pr = 220$, $MAM = 5000$. On the other hand, there are no *SLMSF*, *SWIP* or *SC* parameters in WRDunk and WRDREDunk systems. We must modify the lattice size by setting the decoder pruning *Pr* in this case. The *Pr* is the independent variable, and accuracy and lattice size depend on the *Pr* for WRDunk and WRDREDunk systems.

The dependency of WAC or Word UBTWV-IV accuracy and lattice size wrdSIZE on the *SLMSF*, *SWIP* or *SC* parameters is plotted in figures 7.12 and 7.13 respectively. The baseline WRDunk and WRDREDunk systems are plotted in red. So both, the accuracy on Y-axis and the lattice size on X-axis are dependent variables on the *SLMSF*, *SWIP* or *SC* (or beam pruning *Pr* for the baseline).

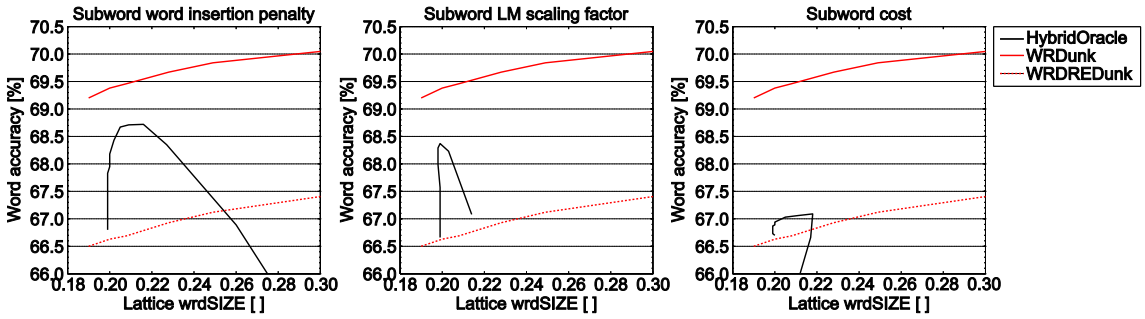


Figure 7.12: Dependency of the WAC and the lattice size wrdSIZE of HybridOracle systems while parameters *SWIP*, *SLMSF* and *SC* are tuned. The accuracy and lattice size of baseline systems WRDunk and WRDREDunk are modified by pruning factor Pr compared to the HybridOracle system.

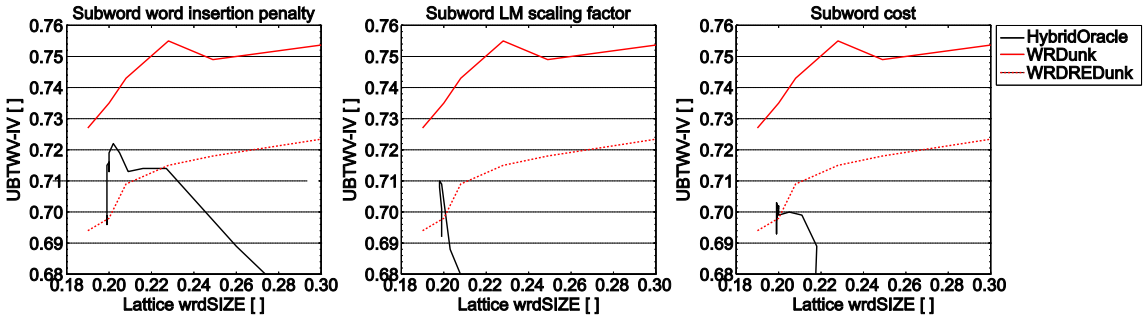


Figure 7.13: Dependency of the Word UBTWV-IV and the lattice size wrdSIZE of HybridOracle systems while parameters *SWIP*, *SLMSF* and *SC* are tuned. The lattice sizes of baseline systems WRDunk and WRDREDunk are modified by pruning factor Pr compared to the HybridOracle system.

The conclusion is, that if the lattices produced by baseline system (WRDunk and WRDREDunk) and the hybrid system are of the same size, better accuracy can be achieved by the baseline system rather than by the hybrid system for certain values of *SLMSF*, *SWIP* or *SC* parameters. If the *SC* parameter is set to 5, the Word UBTWV-IV is 0.700 for the HybridOracle system and 0.694 for the WRDREDunk system (0.006 improvement). On the other hand, if the WRDREDunk system is tuned to produce comparable size of the lattices, then the Word UBTWV-IV is 0.705 and the HybridOracle system is 0.005 worse. The improvement of HybridOracle system is valid only for *SLMSF*, *SWIP* or

SC parameter ranges where accuracy curves lie above the baseline WRDREDunk curve (figures 7.12 and 7.13).

7.4 Hybrid system with subword out-of-vocabulary model

While previous section described a "cheating" oracle system that was used to investigate the influence of different parameters, here we start with a real hybrid system containing subword OOV model.

We use two different subword models for modeling the out-of-vocabulary words. The first is the **phone loop** (chapter 5) and the second is the **phone multigrams** (chapter 6). The phone loop is evaluated just to validate the results published in [Baz02]. Only the *SWIP* parameter is used for tuning in the hybrid subword experiment. We did the experiment also with *SLMSF* and *SC* parameters, but the *SWIP* is found to be the most important.

The out-of-vocabulary terms should be searched in a word-multigram form (IV words as words, OOV words as subwords). Let us suppose that term "`TERM EXAMPLE`" is an OOV term, where word `EXAMPLE` is the OOVword. The OOV term should be searched correctly in form "`TERM ih g z ae m p el`". Unfortunately, the **LatticeSTD** tool (description in section 3.5.5) does not support the word-subword forms. So the OOV terms must be searched in full subword form "`t er m ih g z ae m p el`". Experiments with correct word-subword forms of OOV terms are possible with our **LSE – indexing and search engine** (description in section 3.5.6). Spoken term detection experiments using *LSE* are summarized later in chapter 8.

However, using of the indexing and search engine for all experiments is not optimal due to large number of experiments. Therefore we decided to use the *LatticeSTD* tool for all following experiments and the OOV terms are searched entirely in the subword form. The UBTWV-OOV accuracy is negatively influenced by this decision. The deterioration is 0.008 (see table 8.1 in section 8 for more details) and is not so large, but it should be taken in mind.

The in-vocabulary terms are searched in two variants. Expect that term "`TERM EXAMPLE`" is IV term. All words are let in word forms and also are converted to subword form: `TERM EXAMPLE`, `t er m ih g z ae m p el`. The reason for the subword form is in errors caused by OOV word on surrounding IV words. Influence of this decision is also evaluated more deeply later in chapter 8.

The following experiment is done as a proof of hypothesis, that the IV terms are searched in both variants (word and subword). Phone multigram hybrid system used in this section will be fully described later. The subword part of the hybrid system contains phone multigram units trained on LVCSR pronunciation vocabulary (*HybridMgramDictLVCSR* system). More detailed description of this multigram system is in section 7.7.2 below.

We compare the upper bound term weighted value for in-vocabulary terms (UBTWV-IV) in **only word** (HybridMgram-Wrd), **only subword** (HybridMgram-Swrd) and **both word and subword forms** (HybridMgram-Wrd&Swrd). The comparison is shown in figure 7.14. The dotted black line denotes a system where IV terms are converted to multigram sequences (HybridMgram-Swrd) and detected by the OOV subsystem. The dashed black line denotes system where IV terms are kept in word forms (HybridMgram-Wrd) and detected by the LVCSR subsystem. The solid black line denotes combined system (HybridMgram-Wrd&Swrd). The terms were in both variants, word and multigram. The combined HybridMgram-Wrd&Swrd approach is slightly better for reasonable range of *SWIP*. The highest improvement 0.014 caused by the combination is achieved

for $SWIP = 1.3$. That is why we use combined word and subword form for search of IV terms in the following experiments.

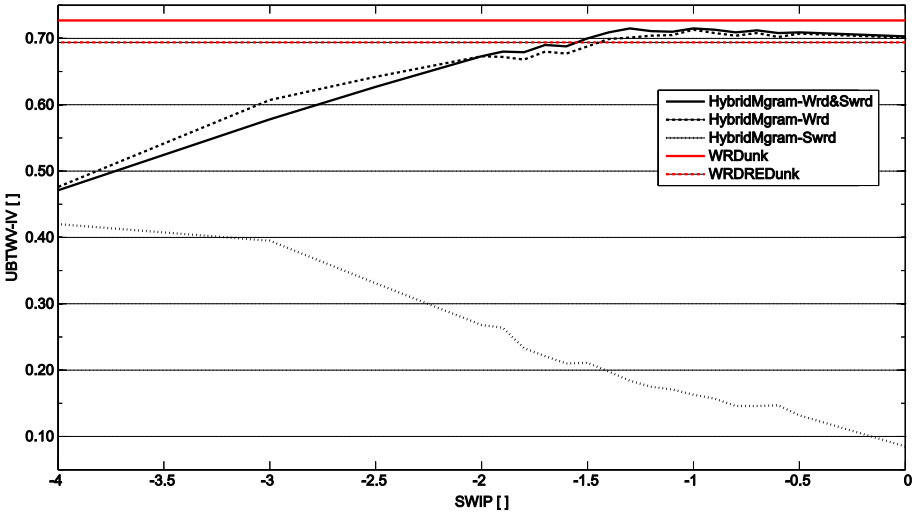


Figure 7.14: Comparison of UBTWV-IV accuracy for in-vocabulary terms searched in word (dashed black), subword (dotted black) and combined (word and subword) form (solid black). The baseline is in red color.

7.5 Phone hybrid system

The out-of-vocabulary part is modeled by the **phone loop** in this system. We experimented with the use unigram and bigram language models over the phones (see chapter 5). The unigram LM is significantly better. The hybrid phone system with unigram LM is denoted as **HybridPhn**. Dependencies of word accuracy and UBTWV on the *SWIP* are plotted in figure 7.15. Dependencies of UBTWV on the lattice size for different *SWIP* parameter values are plotted in figure 7.16.

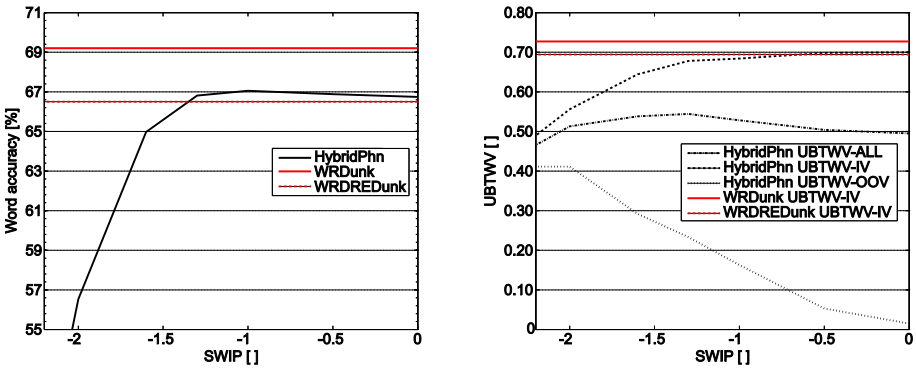


Figure 7.15: Dependency of word accuracy and UBTWV for different *SWIP* parameter values for HybridPhn system. The red color denotes baseline systems.

The word accuracy of HybridPhn is improved by 0.55% (see table 7.5). Disadvantage of HybridPhn system is the size of produced subword index. This system produces phones, so it should be indexed as phone trigrams to fulfill optimal conditions for storing in index. The UBTWV-OOV phnSIZE ratio is significantly lower than for the baseline (section 7.2)

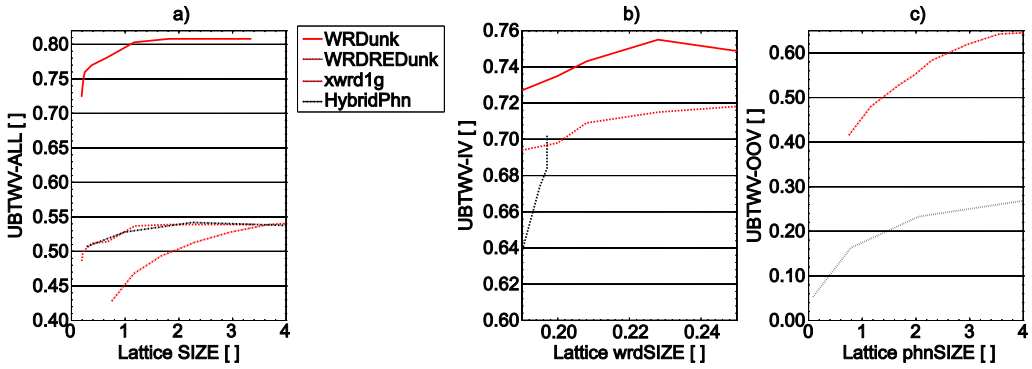


Figure 7.16: Dependency of UBTWV accuracies and lattice sizes depending on different *SWIP* parameter values for HybridPhn system. The red color denotes baseline systems.

System	WAC	UBTWV			*SWIP*	wrdSIZE	phnSIZE
		ALL	IV	OOV			
HybridPhn	**67.05**	0.542	0.674	0.233	1.3	0.195	2.05
WRDREDunk	66.50	0.486	**0.694**	–	–	0.190	–

Table 7.5: Comparison of baseline word recognizer with reduced vocabulary WRDREDunk and hybrid system with the unigram phone LM as subword model HybridPhn. The HybridPhn is tuned to provide best WAC and the best UBTWV-ALL.

xwrd system with unigram language model (see figure 7.16c). The overall UBTWV-ALL accuracy is comparable to the WRDREDunk system (figure 7.16a), which means that the HybridPhn is not better than a system that has no OOV handling at all.

7.6 Phone multigram hybrid system

Variable length phone sequences are the second type of subword units we tested in the hybrid recognition system. Two categories of multigram units are trained. The first one is taken from the subword STD experiments using multigrams (chapter 6). These are trained on a large set of phone sequences.

The second category of multigrams is trained only on the pronunciation vocabulary. This approach is similar to that one proposed in Bazzi's thesis [Baz02], where LVCSR pronunciation dictionary is used for training the phone multigrams.

7.6.1 Hybrid system using constrained multigrams

The "**constrained multigrams**" which provide the best results on the multigram based subword STD chosen for combination into hybrid system. Multigram systems giving the best Mgram UBTWV-OOV accuracy proposed in section 6.4 are summarized in table 7.6. These systems were also considered as candidates for baseline system in section 7.2.

Results

The results of hybrid constrained multigram systems are summarized in figures 7.17 and 7.18. The systems are comparable from the word accuracy point of view. Slightly

System	LM order	Mgram UBTWV			wrdSIZE
		ALL	IV	OOV	
xwrd	1	0.537	0.492	**0.642**	3.5
nosil	1	0.546	0.508	**0.636**	3.2
noxwrd	1	0.535	0.492	**0.637**	7.3

Table 7.6: Comparison of multigram based systems giving the best UBTWV-OOV. The multigram inventory is estimated on *MnoOOV* and the unigram language model is trained on *LnoOOV* (see section 6.2 for more details).

better UBTWV-ALL accuracy is achieved by the xwrd system. The xwrd also achieves the best Mgram UBTWV-OOV accuracy depending on the subword lattice size. The conclusion is, that constrained multigrams are not significantly better. Only the nosil system achieves slightly better word accuracy and UBTWV-IV depending on the lattice size. However, the xwrd system is chosen as the best according to the Mgram UBTWV-OOV accuracy.

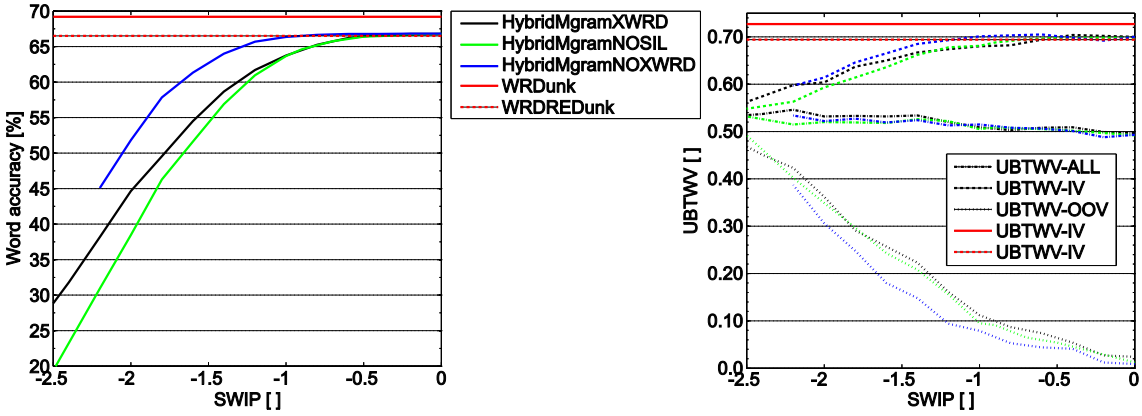


Figure 7.17: Dependency of the HybridMgram systems word accuracy and UBTWV accuracy on the parameter *SWIP*. The red color denotes the baseline systems WRDunk and WRDREDunk. The xwrd, nosil and noxwrd subword models are used for OOV modeling.

7.7 Hybrid system using dictionary multigrams

Compared to "*constrained multigrams*" (Chapter 6), the "**dictionary multigrams**" are trained only on the pronunciation dictionary. The multigram training data are pronunciations of words. Each pronunciation variant is taken as one utterance (phone string). The language model built over the multigrams is also estimated on the segmented (by multigrams) pronunciation dictionary. The difference between "*constrained multigrams*" and "*dictionary multigrams*" is in the size of training data and especially in the prior information of frequency of word occurrences. This should lead to a better hybrid model. In our opinion, word frequencies should be modeled by the word language model. The subword model should model only the pronunciations of unknown words. The subword model will not work properly if the information of word frequencies is incorporated in the subword model, however this happens in "*constrained multigrams*" hybrid systems and *phone hybrid* system.

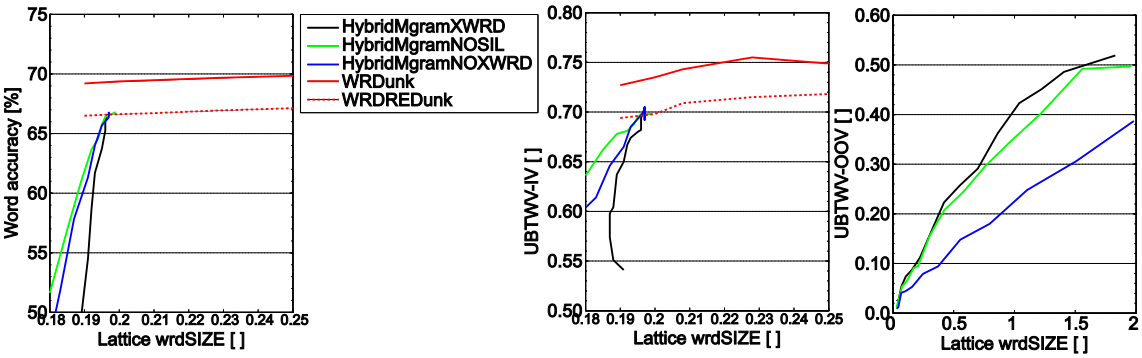


Figure 7.18: Dependency of the HybridMgram systems word accuracy and UBTWV accuracy on the lattice size while parameter *SWIP* is tuned. The red color denotes the baseline systems WRDunk and WRDREDunk. The xwrd, nosil and noxwrd subword models are used for OOV modeling.

We trained the subword system on several different dictionaries and with different parameters. Our baseline for these experiments is similar to [Baz02]. The "dictionary multigrams" are trained on the *WRDRED* dictionary.

We used also a dictionary automatically created by **grapheme to phoneme** (*G2P*) conversion. The word labels are collected from the training corpora of open vocabulary language model (section 4.1.4). Our goal is to evaluate which parameter significantly influences the hybrid system accuracy.

7.7.1 Grapheme to phoneme conversion

Our *grapheme to phoneme* (**G2P**) system was briefly introduced in section 3.5.7, deeper description is attached as appendix B. The *G2P* rules are trained on 90% of randomly selected of the 50k WRD pronunciation dictionary. The rest 10% is used for evaluation of *G2P* accuracy. Only the best pronunciations are considered in the test. 73% of test words are correctly converted to phones and 16% of test words have one phone incorrectly converted in the generated pronunciation.

7.7.2 Multigrams trained on hand-made LVCSR dictionary

Multigrams trained on the LVCSR dictionary are used in [Baz02]. We did the same experiment for better comparison. The WRDRED pronunciation dictionary is taken and multigrams (maximal multigram length $l_{mgram} = 5$, multigram pruning $c_0 = 5$) are trained on the word pronunciations. Hybrid system using the WRDRED dictionary trained multigrams is denoted as **HybridMgramDictLVCSR**. The advantage of WRDRED dictionary is in its correctness, the pronunciations are carefully hand-checked.

We also process the WRDRED dictionary word labels by the *G2P* system. Hybrid system using this subword model (denoted as **HybridMgramDictG2P**) evaluates the influence of *G2P* conversion accuracy on the word or STD accuracy. Comparison of these two systems is in figures 7.19 and 7.20.

We conclude, that the *G2P* conversion has no significant negative influence on the accuracy. UBTWV-IV is influenced slightly negatively, on the other hand the Mgram UBTWV-OOV is influenced positively for certain values of *SWIP* parameter. The HybridMgramDictLVCSR will be used in several further experiments because we want to be comparable to Bazzi [Baz02].

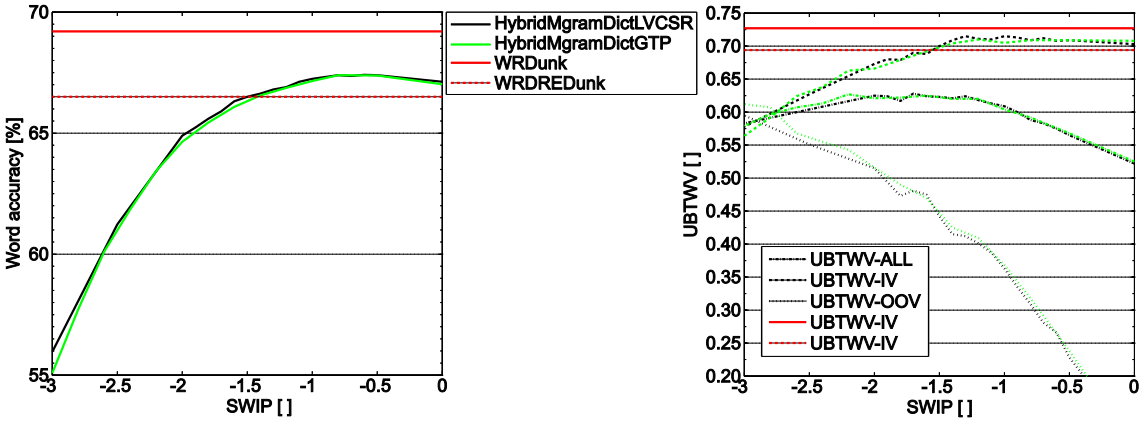


Figure 7.19: Dependency of the HybridMgram systems WAC and UBTWV on the parameter *SWIP*. The red color denotes the baseline systems WRDunk and WRDREDunk. Pronunciations of words of HybridMgramDictLVCSR system are taken from the WRD dictionary. Pronunciations of words of HybridMgramDictG2P system are generated automatically by the *G2P* tool.

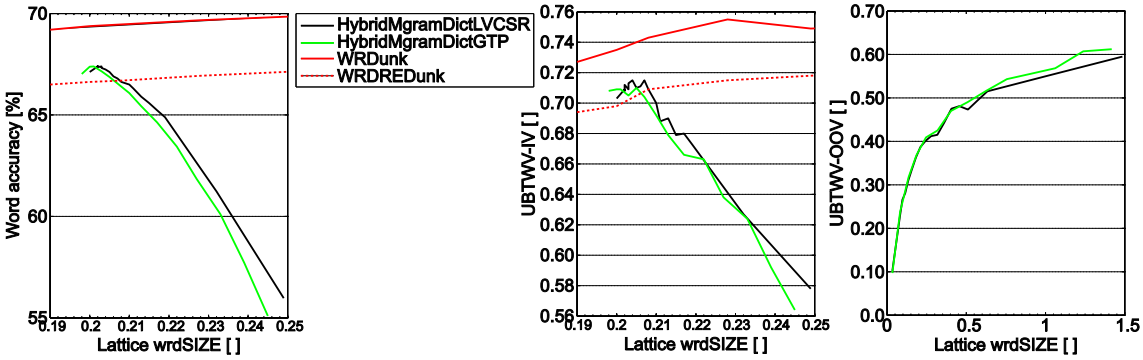


Figure 7.20: Dependency of the HybridMgram systems WAC, UBTWV and lattice size on the parameter *SWIP* is tuned. The red color denotes the baseline systems WRDunk and WRDREDunk. Pronunciations of words of HybridMgramDictLVCSR system are taken from the WRD dictionary. Pronunciations of words of HybridMgramDictG2P system are generated automatically by the *G2P* tool.

7.7.3 Multigrams trained on automatically built LVCSR dictionary

In the previous section, we concluded that the *G2P* used for generating of pronunciation variants has no significant effect on the accuracy of the hybrid system. In this section, we try to answer the question, whether the number of unseen low frequent words has the impact on the accuracy.

System denoted as **HybridMgramDictLarge** is built using automatically generated pronunciation variants of large number of words. A large set of word label is collected from the corpora used for open vocabulary word language model estimation (section 4.3).

Dictionary parameters

The open vocabulary word language model corpora contains 1.2M unique word labels. A part of this word list is "garbage" (words "`AAAAHHHH, 92', ETC.`"), so we perform several steps to clean the word list:

- We omit the WRDunk dictionary. This leads to:
  1. Our OOV words become unseen words for the subword model.
  2. We omit also the IV words so the subword model should not be biased by the IV words.
- We omit words with counts lower or equal than C, where C is a parameter. This filters out the senseless "garbage" words.
- We omit words which have label length shorter or equal than L characters, where L is a parameter. We want to model longer words. Also, the short words are usually "garbage" abbreviations.
- The automatically trained *G2P* tool (section 3.5.7) produces the pronunciation variant and the posterior probability is attached to the variant. We compare the posterior probability to the threshold VP and the probability has to be greater than VP, where VP is a parameter. This should filter out words with "uncertain" pronunciation.

The original size 1.2M of word list is reduced to value in range between 360k words for $L = 2$, $C = 2$, $VP = 0.0$ down to 14k words for $L = 4$, $C = 50$, $VP = 0.9$ depending on the set of parameters L, C and VP.

We "tune" parameters C, L and VP (denoted as *dictionary parameters*) separately and observe their influence on the accuracy. The overall effect of *dictionary parameters* on the accuracy is small (tenths of percent). Therefore, we decided to put figures and brief description related to this experiment to appendix A. The best accuracy is achieved with dictionary parameters $C = 2$, $L = 4$ and $VP = 0.8$.

Multigram pruning parameter

In this experiment, we tune the pruning parameter of the multigram units trained on the dictionary. We set multigram pruning to values $P = 2, 5, 10, 20, 50$. Dependencies of the accuracy on *SWIP* and lattice size are plotted in figures 7.21 and 7.22. The multigram pruning parameter has greater influence on the accuracy than the dictionary parameters from the previous section. We conclude that the UBTWV-OOV accuracy saturates for multigram pruning 5 and 10. We decided to use multigram pruning $P = 5$ in further work.

Our conclusion of this section aimed to the **HybridMgramDictLarge** system is, that the best accuracy of the hybrid system estimated on the large dictionary is achieved with parameters $C = 2$ (word counts), $L = 4$ (word length), $VP = 0.8$ (pronunciation variant posterior probability) and $c_0 = 5$ (multigram pruning) for the hybrid system where the phone multigrams are estimated on the large dictionary. This system (denoted as **HybridMgramDictLarge**) uses parameters tuned in this section for further experiments.

7.7.4 Hybrid systems based on manually versus automatically built dictionary

We compare the accuracies for hybrid system based on the LVCSR dictionary (HybridMgramDictLVCSR) and hybrid system based on the large dictionary (HybridMgramDictLarge). Both multigram systems are trained with multigram pruning parameter $c_0 = 5$.

Our conclusion is, that both systems are comparable. The HybridMgramDictLVCSR system is slightly better on UBTWV-IV and word accuracy, the HybridMgramDictLarge

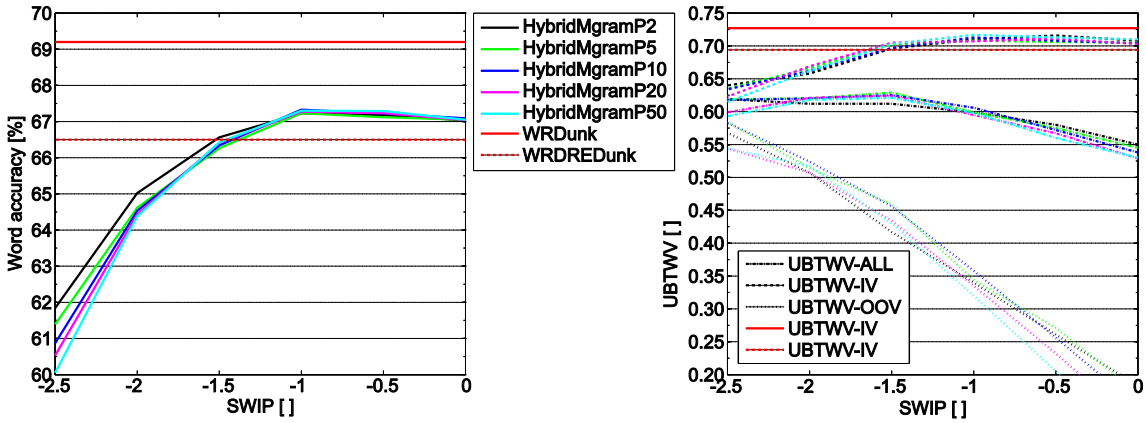


Figure 7.21: Dependency of the HybridMgram systems word accuracy and UBTWV accuracy on the parameter *SWIP*. The red color denotes the baseline systems WRDunk and WRDREDunk. The multigram pruning parameter of trained multigrams c_0 is set from 2 up to 50. Fixed parameters: maximal length of multigrams is $l_{mgram} = 5$, dictionary parameters $C = 2$, $L = 4$ and $VP = 0.8$.

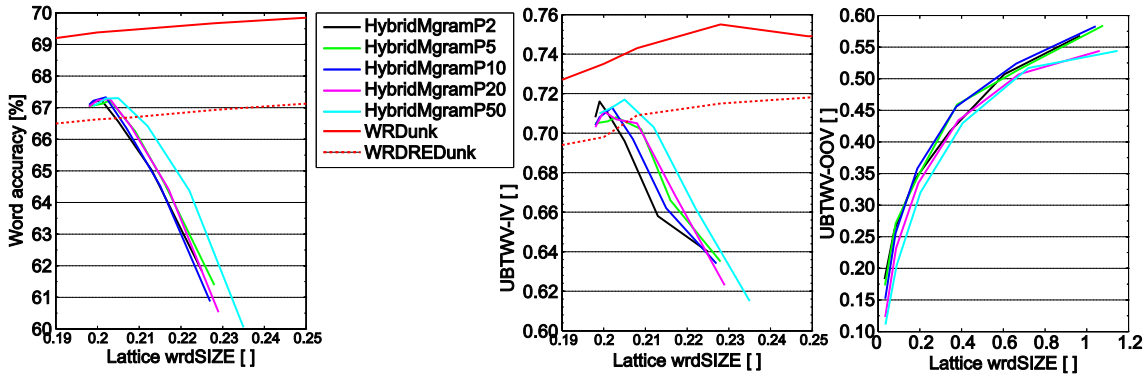


Figure 7.22: Dependency of the HybridMgram systems word accuracy and UBTWV accuracy on the lattice size while parameter *SWIP* is tuned. The red color denotes the baseline systems WRDunk and WRDREDunk. The multigram pruning parameter of trained multigrams c_0 is set from 2 up to 50. Fixed parameters: maximal length of multigrams is $c_0 = 5$, dictionary parameters $C = 2$, $L = 4$ and $VP = 0.8$.

is slightly better on the Mgram UBTWV-OOV and achieves smaller size of the lattices for comparable word accuracy and UBTWV-IV. Figures 7.23 and 7.24 compare these systems.

7.7.5 Subword model with bigram language model

Our approach to build hybrid word-subword recognition networks allows to use subword language model with higher order than unigrams. We tested bigram subword language model[8]. We compare both, the HybridMgramDictLVCSR and the HybridMgramDictLarge systems. The results are summarized in figures 7.25 and 7.26. From the word accuracy point of view, the HybridMgramDictLVCSR bigram system has slightly lower maximum accuracy than the unigram system. The HybridMgramDictLarge bigram system is slightly better than the unigram system on the word accuracy.

The UBTWV-ALL of bigram systems are comparable. Only UBTWV-OOV of Hybrid-

[8]We are unable to compile the network for the trigram model due the memory limitations.

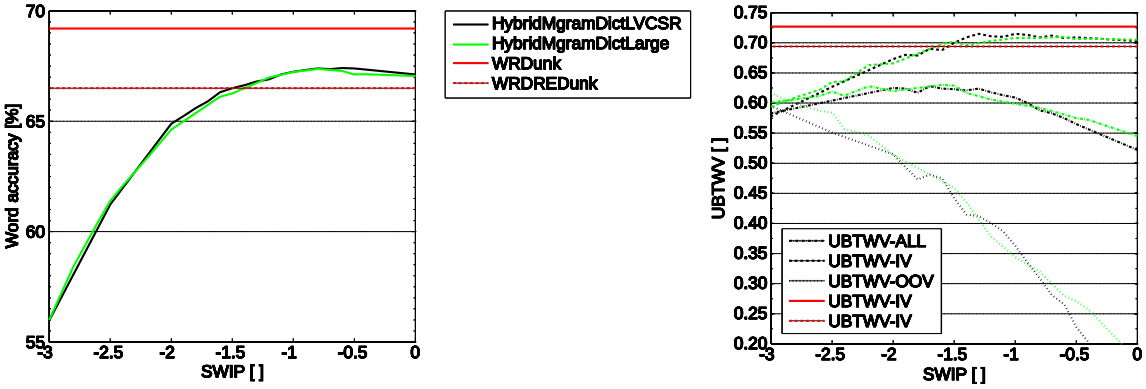


Figure 7.23: Dependency of the hybrid multigram systems WAC and UBTWV accuracy on the *SWIP* parameter for two different subword models. The HybridMgramDictLVCSR is trained on the hand made LVCSR WRDRED dictionary. The HybridMgramDictLarge is trained on dictionary created by *G2P* tool automatically trained on large word list derived from the word corpora. The red color denotes the baseline systems WRDunk and WRDREDunk. The multigram pruning parameter of trained multigrams is $c_0 = 5$ for both systems.

MgramDictLarge is significantly higher for certain range of *SWIP* parameter. UBTWV-IV of bigram systems are slightly lower for higher values of *SWIP* compared to the unigram systems. On the other hand, UBTWV-OOV are significantly higher. The greatest differences between unigram and bigram subword systems are seen on accuracy related to the same lattice size (figure 7.26). We can see, that the bigram systems have lattice sizes significantly reduced. The lattice size does not grow so fast with increasing accuracy while the *SWIP* is tuned.

Although HybridMgramDictLVCSR2gr and HybridMgramDictLarge2gr are comparable, we decide to select **HybridMgramDictLarge2gr** according to the higher Mgram UBTWV-OOV accuracy.

7.7.6 Beam pruning in the decoder

The following experiment compares the behavior of the selected hybrid system (HybridMgramDictLarge2gr) while beam pruning parameter of the decoder changed. We try to soften the beam pruning from baseline value $Pr = 220$ to $\{250, 300, 350, 400\}$. We look at the changes of accuracy and lattice size for several values of *SWIP*. Results of this experiment are summarized in figure 7.27.

The hybrid system word accuracy follows the WRDREDunk baseline as the lattice size grows. Softening the beam pruning has no negative influence on the improvement of WAC achieved by the hybrid system.

Changes of beam pruning have greater and more complex influence to the UBTWV accuracy. The choice of *SWIP* parameter depends on the beam pruning. The highest improvement of UBTWV-IV is achieved for beam pruning $Pr = 300$. The improvement on the Mgram UBTWV-OOV for the same pruning is about 0.130 absolute. On the other hand, it must be mentioned, that the best accuracy of standalone subword system is not reached. The dash-dotted red line denotes the best standalone subword system according to Mgram UBTWV-OOV in the right part of figure 7.27. It is the xwrd system with unigram language model (denoted xwrd1gr).

As the decoder beam pruning is softened (lattice size grows), the UBTWV-IV accuracy does not increase so fast compared to the WRDREDunk baseline. The accuracy is worse

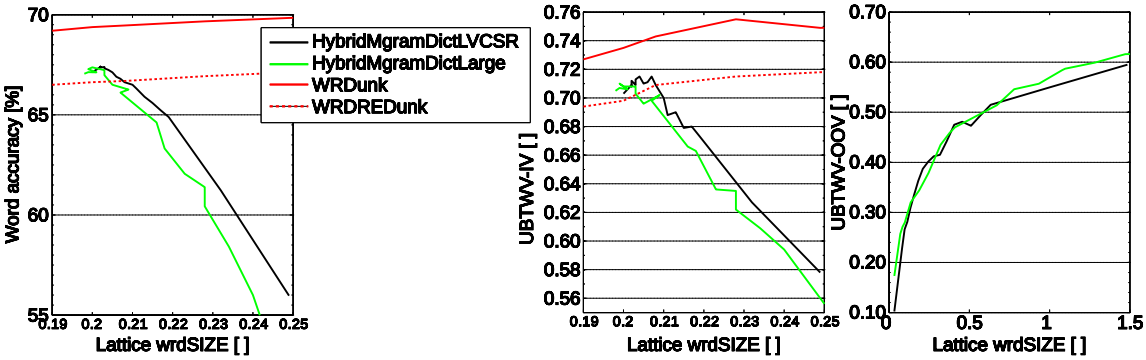


Figure 7.24: Dependency of the hybrid multigram systems WAC, UBTWV accuracy and lattice size on the *SWIP* parameter for two different subword models. The *HybridMgramDictLVCSR* is trained on the hand made LVCSR WRDRED dictionary. The HybridMgramDictLarge is trained on dictionary created by *G2P* tool automatically trained on large word list derived from the word corpora. The red color denotes the baseline systems WRDunk and WRDREDunk. The multigram pruning parameter of trained multigrams is $c_0 = 5$ for both systems.

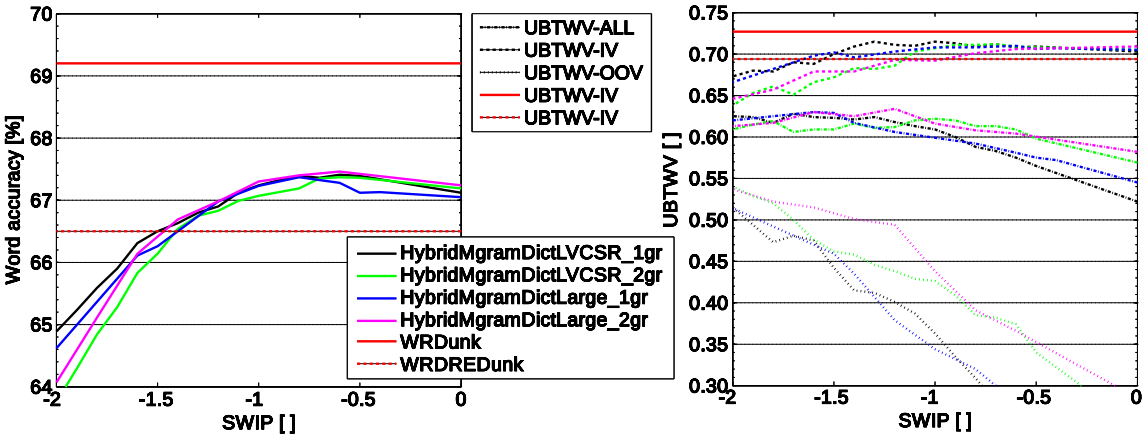


Figure 7.25: Dependency of the hybrid multigram systems word accuracy and UBTWV accuracy on the subword model. The HybridMgramDictLVCSR and the HybridMgramDictLarge systems are evaluated also with bigram language model in the subword part (**_2gr*). The red color denotes the baseline systems WRDunk and WRDREDunk.

than the baseline for beam pruning values softer than $Pr = 350$.

The Mgram UBTWV-OOV accuracy was better with softer beam pruning (increasing lattice size) compared to the WRDREDunk baseline. The saturation tendency can be seen in figure 7.27. The saturated value of hybrid system Mgram UBTWV-OOV accuracy is around 0.630, but the saturation of the xwrd1gr baseline is about 0.650. This means that hybrid system looses about 0.020 of the best possible accuracy compared to the baseline.

From the STD accuracy point of view, it is important to note, that by tuning the *SWIP* parameter, we obtained better UBTWV-IV at the price of decreasing Mgram UBTWV-OOV and vice-verse. Two system configurations marked with square and circle are plotted in figure 7.27. While the "circle" system is tuned to provide better UBTWV-IV, the Mgram UBTWV-OOV is worse. The "square" system is tuned to achieve better Mgram UBTWV-OOV.

The other point of view – the index size – indicates that hybrid system achieves a significant improvement. If our goal is to be better than the IV baseline (WRDREDunk) in

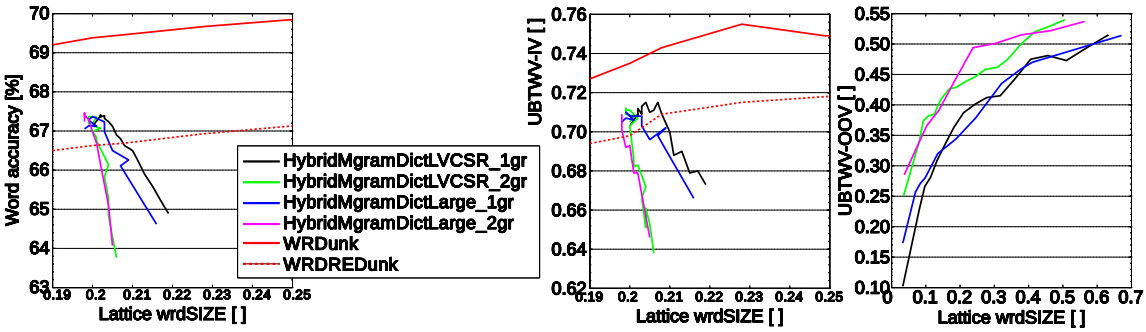


Figure 7.26: The dependency of the hybrid multigram systems word accuracy and UBTWV accuracy on the lattice size while parameter *SWIP* is tuned. The HybridMgramDictLVCSR and the HybridMgramDictLarge systems are evaluated also with bigram language model in the subword part (**_2gr*). The red color denotes the baseline systems WRDunk and WRDREDunk.

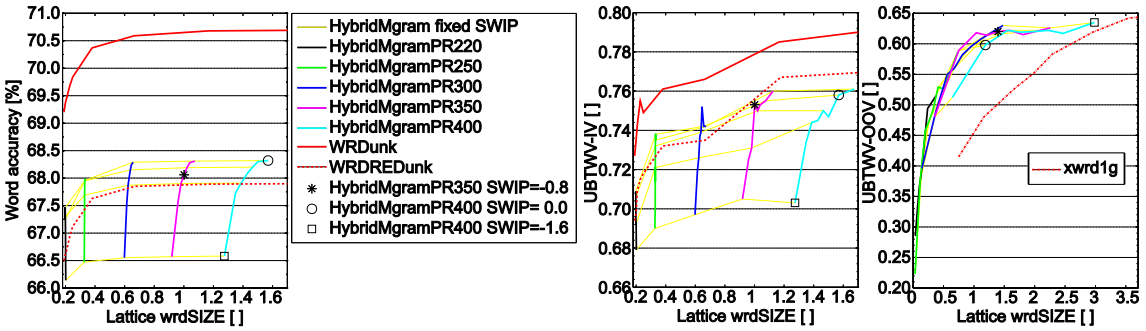


Figure 7.27: The dependency of the HybridMgramDictLarge2gr systems WAC UBTWV accuracies and lattice size on the parameter *SWIP* while decoder beam pruning changed. The red color denotes the baseline systems WRDunk (solid), WRDREDunk (dashed) and xwrd with unigram LM (dash-dotted). The color (black, green, blue, magenta and cyan) lines correspond to the beam pruning factors. The yellow lines connect "points" with the same *SWIP* parameter value (for better orientation).

the UBTWV-IV/wrdSIZE ratio, we achieve also significant improvement in the UBTWV-OOV/wrdSIZE ratio. We choose the "star" system as the best hybrid system.

The "star" system configuration is **HybridMgramDictLarge2gr** (HybridMgramDictLarge with bigram subword LM), decoder beam pruning $Pr = 350$, subword scaling parameters $SLMSF = 1.0$, $SWIP = -0.8$ and $SC = 0.0$. This system achieves UBTWV-IV accuracy close to the baseline WRDREDunk having the same word index size. It improves the Mgram UBTWV-OOV accuracy by 0.110 keeping the same subword index size, or reduces subword index size to one half keeping the same UBTWV-OOV accuracy.

Finally, we also tried to tune decoder language scaling factor and word insertion penalty, but these did not improve the "star" hybrid system significantly.

7.8 Memory and speed

Memory consumption and system speed (decoding time) are important factors for practical use. We evaluate the **real-time factor**[9] – *RT factor* and memory allocated by the decoder after loading the recognition network and acoustic model. Real-time factors

[9]Proportion of the time of 1 CPU core needed to decode a portion of acoustic data to the time length of the portion of acoustic data.

are measured without feature extraction which has a constant RT factor and is the same in all experiments. Also, time consumed by spoken term detection algorithm is not included into RT factor, because it represents only fraction of the time. Both RT factor and allocated memory depend on the implementation of the decoder, so they can vary for different decoders. Decoding speed is tested on Intel® Xeon® CPU, model E5345 at frequency 2.33GHz processor with sufficient size of RAM.

Table 7.7 compares hybrid systems to the baseline systems from memory and index size, accuracy and speed points of view. The first part of the table compares baseline systems with fixed beam pruning $Pr = 220$. We see that both multigram systems (xwrd and noxwrd) are significantly (up to 8 times) slower and produce significantly larger indexes (up to 18 times) compared to WRDunk. On the other hand, the multigram xwrd system with unigram LM consumes only tenth of RAM compared to the other systems. The multigram xwrd system was chosen as the baseline system of OOV terms detection because it reaches the highest UBTWV-OOV accuracy.

The baseline hybrid systems (HybridMgramDictLarge and HybridMgramDictLarge2gr) with beam pruning $Pr = 200$ run relatively fast ($RT = 2$) and provide UBTWV-IV close to the WRDREDunk system. The UBTWV-OOV accuracy is about the half of xwrd system, but the index size is one tenth.

A **combined baseline system** is combination of baseline systems (WRDREDunk and xwrd) after the decoding, on the search level (see figure 7.4). "Combined" UBTWV-IV accuracy is the accuracy of WRDREDunk system, UBTWV-OOV accuracy is the accuracy of xwrd system. RT factor and index size are sums of word and sub-word systems.

Bottom three parts of table 7.7 present more precise comparison of hybrid and combined baseline system. We took 3 hybrid systems (HybridMgramDictLarge2gr, HybridMgramDictLarge and HybridMgramDictLVCSR2gr) and tuned the combined baseline systems to produce comparable UBTWV accuracy. Then these systems can be compared from speed and index size point of view.

The first system is the "star" system from the previous section. It is HybridMgramDictLarge2gr system with decoder beam pruning $Pr = 350$, subword scaling parameters $SLMSF = 1.0$, $SWIP = -0.8$ and $SC = 0.0$. This system was chosen as the system which produces the best UBTWV-OOV having UBTWV-IV and word index size comparable with the baseline WRDREDunk in the previous section. Here we see, that apart from the best accuracy, this system is not much usable in practice. The real time factor is $RT = 28.4$ which is two time slower than the *combined baseline system* with real time factor sum $RT = 14$. One advantage of this system is the index size which is about 40% smaller.

HybridMgramDictLarge system with the same subword scaling parameters is chosen as the second hybrid system. The beam pruning $Pr = 250$ is set in this case. The real time factor $RT = 5.25$ of the combined baseline improves to $RT = 4.15$, which is 20% faster. Also, the hybrid index size is 38% smaller than the sum index size of the combined baseline system.

The last system is HybridMgramDictLVCSR2gr system with $Pr = 270$ and subword scaling parameters $SLMSF = 1.0$, $SWIP = -1.3$ and $SC = 0.0$. Compared to HybridMgramDictLarge2gr, this hybrid system with bigram subword language model consumes about 500MB less RAM. The UBTWV accuracy is close to saturated accuracies of the combined baseline system. If the baseline systems are tuned to produce comparable accuracy, this hybrid system is 9% faster and achieves only 34% of the index sizes of the baseline systems. The UBTWV-OOV accuracy deterioration is 0.027 against the "reasonably" saturated xwrd multigram baseline 0.647. The UBTWV-IV accuracy deterioration is 0.031 against the "reasonably" saturated WRDREDunk baseline 0.754.

System	LM	# WRDn-grams		# SWRD n-grams		RAM	Pr	RT	UBTWV			wrdSIZE		
	order	1	2	1	2 (3)				ALL	IV	OOV	sum	wrd	swrd
WRDunk	2	50.0k	2.0M	–	–	550MiB	220	1.36	0.724	0.727	0.715	0.190	0.190	–
WRDREDunk	2	49.2k	1.6M	–	–	470MiB	220	1.32	0.486	0.694	–	0.190	0.190	–
xwrd	1	–	–	3.0k	–	22MiB	220	7.11	0.537	0.492	0.642	3.540	–	3.540
noxwrd	3	–	–	3.0k	451k (161k)	315MiB	220	10.75	0.630	0.647	0.593	1.740	–	1.740
HybridMgramDictLarge	$2+1$	49.2k	1.6M	7.8k	–	685MiB	220	2.00	0.592	0.708	0.320	0.335	0.200	0.135
HybridMgramDictLarge2gr	$2+2$	49.2k	1.6M	7.8k	136.8k	2310MiB	220	2.40	0.608	0.701	0.391	0.337	0.198	0.138
WRDREDunk	2	49.2k	1.6M	–	–	470MiB	330	7.79	0.528	0.754	–	0.950	0.950	–
xwrd	1	–	–	3.0k	–	22MiB	210	6.19	0.528	0.489	0.619	2.960	–	2.960
HybridMgramDictLarge2gr	$2+2$	49.2k	1.6M	7.8k	136.8k	2310MiB	350	28.37	0.713	0.753	0.620	2.400	1.000	1.400
WRDREDunk	2	49.2k	1.6M	–	–	470MiB	226	1.67	0.499	0.714	–	0.210	0.210	–
xwrd	1	–	–	3.0k	–	22MiB	160	3.58	0.481	0.470	0.500	1.400	–	1.400
HybridMgramDictLarge	$2+1$	49.2k	1.6M	7.8k	–	685MiB	250	4.15	0.651	0.715	0.501	1.000	0.340	0.660
WRDREDunk	2	49.2k	1.6M	–	–	470MiB	260	3.24	0.512	0.723	–	0.380	0.380	–
xwrd	1	–	–	3.0k	–	22MiB	210	6.19	0.528	0.489	0.619	2.960	–	2.960
HybridMgramDictLVCSR2gr	$2+2$	49.2k	1.6M	4.0k	42.3k	1855MiB	270	8.62	**0.691**	**0.723**	**0.615**	1.100	0.440	0.700

Table 7.7: Comparison of memory and CPU requirements of hybrid systems. Column *order* denotes the order of used language models. $2+1$ means bigram word and unigram subword LM. The following 4 columns contain the numbers of particular unigrams/bigrams. The number in brackets is the number of trigrams for the noxwrd system. Occupied memory after the recognition network is loaded by the decoder is in column *RAM*. Acoustic model is not included. Its size is constant for all experiments: 190MB. *Pr* denotes chosen beam pruning. RT is the estimated real time factor. Columns UBTWV and wrdSIZE denote the accuracies and index sizes of particular systems. The first part of the table (the first 6 rows) compares baseline and hybrid systems having beam pruning $Pr = 220$. The following 3 parts of the table show three different hybrid systems and appropriate baseline systems having comparable UBTWV accuracy. The real time factor is estimated on Intel® Xeon® CPU, at frequency 2.33GHz.

The analysis of CPU and memory or disk consumptions of hybrid systems shows that hybrid systems are faster and reduce needed disk space for storage of the indexes. However, a hybrid system tuned to achieve accuracy comparable to "saturated" combined baseline system is about two times slower. If the accuracy is not the most important quality of STD system, hybrid system can provide very good performance. The needed decoding time can be reduced by 10% and the index size by 66% at the cost of 5% deterioration of UBTWV accuracy (HybridMgramDictLVCSR2gr system).

We conclude, that hybrid system based on LVCSR vocabulary (HybridMgramDictLVCSR2gr) is faster and consumes less memory than the HybridMgramDictLarge2gr system. This is caused by about two times smaller subword part. Also, the decoder can influence the RT factor. We noticed that the RT of HybridMgramDictLarge2gr system for higher beam pruning increased significantly more than linearly. This was not observed in case of WRDREDunk system.

7.9 Comparison of confidence estimations for hybrid system

This section deals with the influence of confidence measures on the accuracy of the hybrid system. Confidence measures for standalone word and multigram systems were evaluated in sections 4.4 and 6.5.1. We found that $SOLP$[10] confidence measure provides the best accuracy. So the following experiments investigate only the $SOLP$ and LP[11] confidence measures. The LP has been used in all hybrid experiments.

In this experiment, the hybrid system configuration is the large dictionary trained with bigram subword LM, and decoder beam pruning $Pr = 350$: *HybridMgramDictLarge2gr*. The configurations of baseline systems (word and subword) are taken from the previous section 7.8 (table 7.7). The word system WRDREDunk has beam pruning set to $Pr = 330$ and the subword system xwrd1gr (with unigram LM) has beam pruning set to $Pr = 210$ to achieve comparable accuracy on UBTWV. We also evaluate the subword system with bigram LM (denoted xwrd2gr) to see the influence of bigram LM to the TWV accuracy. The beam pruning is set to $Pr = 240$ in the case of xwrd2gr system.

The TWV accuracies for different confidence estimators are summarized in table 7.8. Terms are searched *in word and subword part* (denoted **HybridWRDMGRAM**), *only in word part* (denoted **HybridWRD**) and *only in subword part* (denoted **HybridMGRAM**).

The choice of confidence measure has significant impact also in case of hybrid system. The SOLP provides significant improvement in comparison to LP confidence for both UBTWV and TWV.

If the hybrid HybridMgramDictLarge2gr – wrdmgram and baseline WRDREDunk systems are compared from the IV terms point of view, it is interesting that we loose TWV-IV accuracy with the hybrid system. The UBTWV-IV are comparable, baseline LP is 0.756 against hybrid LP 0.759. However, the baseline WRDREDunk LP TWV-IV 0.471 is about 0.030 higher than the hybrid LP UBTWV-IV which is 0.442. The deterioration of hybrid system is 0.020 for SOLP compared to WRDREDunk.

On the other hand, the hybrid system improves the TWV-OOV accuracy. The baseline system is xwrd1gr in this case. The UBTWV-OOV are nearly comparable, baseline LP is 0.619 against hybrid LP 0.624. The baseline LP TWV-OOV is only 0.135 and is about 0.217 lower that the hybrid LP UBTWV-OOV which is 0.352. The SOLP confidence

[10]SOLP: Sum of Overlapped Link Posterior probabilities
[11]LP: Link Posterior probability

System		CM	TWV			UBTWV			wrdSIZE
			ALL	IV	OOV	ALL	IV	OOV	
WRDREDunk		LP	0.330	0.471	–	0.529	0.756	–	0.96
WRDREDunk		SOLP	0.388	0.554	–	0.542	0.775	–	0.96
xwrd1gr		LP	0.065	0.094	0.135	0.528	0.489	0.619	2.96
xwrd1gr		SOLP	0.099	0.138	0.137	0.533	0.495	0.620	2.96
xwrd2gr		LP	0.162	0.189	0.235	0.578	0.562	0.615	2.37
xwrd2gr		SOLP	0.193	0.236	0.250	0.588	0.577	0.613	2.37
HybridMgramDictLarge2gr	WRD	LP	0.307	0.439	–	0.531	0.758	–	1.00
	WRD	SOLP	0.350	0.501	–	0.533	0.762	–	1.00
	mgram	LP	0.099	0.025	0.352	0.452	0.378	0.624	1.40
	mgram	SOLP	0.106	0.026	0.363	0.449	0.372	0.630	1.40
	wrdmgram	LP	0.365	0.442	0.352	0.718	0.759	0.624	2.40
	wrdmgram	SOLP	0.417	0.533	0.363	0.732	0.777	0.630	2.40

Table 7.8: Comparison of two different term confidence estimators. LP – Link Posterior, SOLP Sum of Overlapped Link Posteriors. Beam pruning of HybridMgramDictLarge2gris set to $Pr = 350$, beam pruning of baseline systems is set to achieve comparable UBTWV accuracy. "WRD" means search only in word part, "mgram" means search only in subword part and "wrdmgram" means search in both word and subword parts.

improvement is 0.226.

If we compare the subword baseline with bigram LM – xwrd2gr the difference is about 0.117, which is very large.

The improvement of TWV-IV caused by incorporation of IV terms searched also in subword variant (HybridMgramDictLarge2gr – mgram) is only 0.003 in case of LP measure. If the confidence is SOLP, the TWV-IV improves by 0.032 (from 0.501 to 0.533). This is larger improvement than in the case of UBTWV-IV where the difference is 0.015. This means that the IV term detected also in subword form contains information important for better estimation of robust confidence allowing for comparison of terms according to one global threshold.

The SOLP outperforms the LP term confidence estimation in most experiments. From the TWV point of view, hybrid system deteriorates about 0.030 on IV terms but improves the TWV accuracy on OOV terms by about 0.117 (compared to xwrd2gr system). This points out, that the word model provides more robust term confidence across different terms.

Similar trend was seen also in other configurations of hybrid system.

7.10 Conclusion

Conclusions of hybrid word-subword systems are given in this section. Theoretically, we should obtain better IV and OOV accuracy with equal or smaller lattice size (index size) with the hybrid system than is achieved by the combination of standalone systems at the

level of term detection. Our experiments have however shown, that this can be achieved only for a certain range of system parameter settings (beam pruning, hybrid network penalties and scales). If standalone systems are tuned to the best accuracy (separately), this combination is not overcome by the hybrid system presented in this thesis. We can only make it smaller and faster, but still with little deterioration of accuracy.

It is interesting to notice, that the UBTWV-OOV of pure subword multigram system (xwrd) is not outperformed by the hybrid systems, while for UBTWV-IV the hybrid systems show superiority. This can be explained by the inaccuracy of estimated place of out-of-vocabulary words (represented by `<unk>`).

Our explanation, why UBTWV-OOV is not better for the hybrid than for pure subword multigram system, is the following: the strong word-model in the hybrid system can cause that for an OOV, the sub-word model is not activated at all – the system does not enter the `<unk>` part of the recognition network. In this case, the system actually misses the OOV without any chance to recover it, resulting in a miss. This problem does not occur in subword systems where the whole utterance is recognized in subwords (so no misses are produced).

On the other hand, a false alarm of OOV occurrence does not necessarily cause that the IV term (overlapped with the OOV false alarm) is not detected. In this case, the IV can be still converted to subwords and searched in the subword form.

This leads to conclusion, that it is important not to miss the OOV parts of utterances for hybrid systems. The position of `<unk>` must be estimated accurately.

Chapter 8

Indexing and search with hybrid system

In the previous section, we searched parameters and configurations of hybrid system with the best accuracy as criterion. According to these experiments, we chose the *HybridMgramDictLarge2gr* system with beam pruning $Pr = 350$, $SLMSF$[1] $= 1.0$, $SWIP$[2] $= -0.8$ and SC[3] $= 0.0$. In this section, the system is indexed by our indexing and search system to evaluate how to convert terms to word-subword form.

Our **Lattice Search Engine** – *LSE* (section 3.5.6) was used in the NIST STD evaluation in 2006. Detailed description of the engine was published in [Fap07] and it is out of the scope of this thesis. Only the most important features of LSE are presented in the following paragraphs. The search engine indexes lattices produced by a decoder. Terms defined by user are searched after the lattices are indexed. The term is a sequence of "word" labels appearing in the lattices. The sequence of words can be searched as "quoted" ("`Igor Szöke`") or "unquoted" (`keyword spotting thesis`). Only the *quoted* variant is taken into account in this thesis (according to NIST STD 2006). Another feature is the option of search with or without a verification in the lattice.

The following steps are performed in the case of search **without verification**:

1. Groups of overlapped units (words or phone multigrams) are identified in the lattice, these are substituted by the best candidate and indexed.

2. The input term is decomposed into sequence of units and each unit is searched in the index.

3. Then term candidates are collected from unit candidates by applying time constraints on the unit candidates. The time constraint stipulates that two adjacent words must be closer than 0.5 second in time (according to NIST STD 2006). Term candidates which satisfy time constraint are finally filtered for overlapping candidates (only the best one is taken). The term posterior probability is approximated from posterior probabilities of units.

The "**with verification**" mode includes one more step in the processing of a term. The existence of term is verified in the lattice. Sequence of links and nodes corresponding to the term must exist in the lattice, and "precise" term posterior probability is estimated.

[1] *SLMSF*: Subword language scale factor.
[2] *SWIP*: Subword word insertion penalty.
[3] *SC*: Subword cost (global penalty of going to subwords).

The disadvantage of verification is the need of storing not only the index but also the lattice. That is why the search speed and size of index are negatively affected by the verification step. In NIST STD 2006 evaluations, we found, that the verification has no significant impact on the accuracy.

8.1 Term in a hybrid form

Because the hybrid system produces hybrid word-subword lattices, we need to know which word-subword variant of term should be searched to achieve the best accuracy. It is also good to remind that in our case, the *IV term* contains *only IV words*. The *OOV term* contains *at least one OOV word* (but it can contain also some IV words).

The term list is built in several different ways for the hybrid word-subword spoken term detection task:

- All words are searched only as word forms (denoted **IVwrd**) in the IV term. "`TERM EXAMPLE`"
- All words are converted and searched only as subword forms (denoted **IVmgram**) in the IV term. "`t-er-m ih-g-z-ae-m p-el`"
- IV term is searched in two variants (denoted **IVwrdmgram**). All words are let in word forms and also are converted to subword form. This approach is used only in one baseline experiment. "`TERM EXAMPLE`", "`t-er-m ih-g-z-ae-m p-el`"
- IV term is searched in all possible word-subword variants (denoted **IVhybrid**). "`TERM EXAMPLE`", "`t-er-m ih-g-z-ae-m p-el`", "`TERM ih-g-z-ae-m p-el`", "`t-er-m EXAMPLE`"
- All words (IV and OOV) are converted and searched only as subword forms (denoted **OOVmgram**) in the OOV term. "`t-er-m ih-g-z-ae-m p-el`" – `EXAMPLE` is OOV
- IV words are searched as word forms and OOV words are converted and searched as subword form (denoted **OOVwrdmgram**) in the OOV term. "`TERM ih-g-z-ae-m p-el`" – `EXAMPLE` is OOV
- OOV term is searched in all possible word-subword variants (denoted **OOVhybrid**). IV words are searched as word or subword forms, OOV words are searched only as subword forms. "`TERM ih-g-z-ae-m p-el`", "`t-er-m ih-g-z-ae-m p-el`" – `EXAMPLE` is OOV

We built several term lists with different forms of IV and OOV words in IV and OOV terms. The results are summarized in table 8.1. The tested system is HybridMgramDictLarge2gr with $Pr = 350$, $SLMSF = 1.0$, $SWIP = -0.8$ and $SC = 0.0$.

The *baseline* results are produced by the **LatticeSTD** tool. This tool was used for all past STD experiments in this thesis.

The LSE denotes results are produced by the *Lattice Search Engine*. Lattices produced by the hybrid system are first indexed by LSE. The list of terms is built and processed by a search module in LSE. The confidence is computed as LP – link posterior because C_{max} is not implemented in LSE. Produced results are scored in the same manner as the baseline.

In table 8.1, the rows *LSE IVwrd* and *LSE IVmgramOOVmgram* compare the *LSE* system to the *baseline IVwrd* and *baseline IVmgramOOVmgram*. The termlists are the

System	Verif.	TWV			UBTWV		
		ALL	IV	OOV	ALL	IV	OOV
baseline IVwrd	yes	0.305	0.436	-	0.527	0.753	-
baseline IVmgramOOVmgram	yes	0.103	0.029	0.340	0.453	0.381	0.620
baseline IVwrdmgramOOVmgram	yes	0.363	0.441	0.345	0.713	0.753	0.620
LSEIVwrd	no	0.313	0.447	-	0.530	0.757	-
LSEIVwrd	yes	0.303	0.433	-	0.523	0.748	-
LSEIVmgramOOVmgram	no	0.113	0.030	0.383	0.434	0.343	0.646
LSEIVmgramOOVmgram	yes	0.105	0.030	0.351	0.452	0.380	0.620
LSEIVwrdOOVwrdmgram	no	0.368	0.447	0.436	0.735	0.757	0.682
LSEIVwrdOOVwrdmgram	yes	0.355	0.433	0.377	0.712	0.748	0.628
LSEIVwrdOOVhybrid	no	0.370	0.447	0.453	0.735	0.757	0.682
LSEIVwrdOOVhybrid	yes	0.358	0.433	0.395	0.712	0.748	0.628
LSEIVhybridOOVhybrid	no	**0.378**	**0.459**	**0.453**	**0.737**	**0.761**	**0.682**
LSEIVhybridOOVhybrid	yes	0.363	0.441	0.395	0.717	0.755	0.628

Table 8.1: Summary of different forms of IV and OOV words in IV and OOV terms. The baseline systems are produced by *LatticeSTD*. Systems below denote standard experiments with the LSE.

same for both experiments. The results are similar for both *LSE* and *baseline* systems. Theoretically, the results should be exactly the same, because both systems are based on the same formula. However the implementation is different which leads to small differences in floating point numbers.

It is interesting to notice, that the search without verification achieves better accuracy in all experiments. Explanation of this fact is given in the following section.

Using hybrid form of OOV terms (remember that here, IV words from OOV terms are in both word and multigram forms) brings no further improvement of UBTWV-OOV. The aim of this analysis was to see if the presence of an OOV does not hurt the recognition of preceding IV word, when a hybrid recognizer is used. Decomposition of IV into multigrams would then probably allow to correct some errors caused by an OOV (for example, recognizer entering the OOV part of the recognition network too soon). The fact that we did not see any improvement documents that the OOV model is activated correctly only in OOV parts of speech.

A slight improvement of accuracy of about 0.004 is achieved, when the IV terms are searched in their hybrid form. This improvement is not significant to make explicit conclusion, but could be explained by activation of OOV model in some sentences that are unlikely for the word language model.

We conclude that this experiment using real indexing & search system verified results achieved by the *LatticeSTD*. If OOV term is searched with verification, expanding the OOV term from multigram form *OOVmgram* (the whole term in multigram form) to word–multigram form *OOVwrdmgram* (IV words in word form and OOV words in multigram form) brings only small improvement in accuracy.

8.2 N-best multigram term variants

The second experiment done with LSE indexing and search engine aimed at the expansion of 1-best to n-best multigram variants of OOV words. Similar experiment was done in section 6.5, where each word of the term list is segmented by multigrams to n-best variants. The conclusion was that the accuracy saturates for 3-best variants with an improvement of 0.014 for UBTWV-OOV.

The *LSE IVwrdOOVwrdmgram* system is used. OOV words in OOV terms are segmented to 1 (baseline), 2, 3 and 5-best variants. The results are summarized in table 8.2. Significant improvement can be seen in experiments with verification. The n-best variants have no influence to systems without verification.

This clarifies why systems with verification are worse than without verification. The multigram subword model is not 100% accurate. It produces several possible multigram sequences representing the underlying speech. An OOV part of speech contains searched multigrams (representing an OOV word), but these multigrams are not properly connected. That is why systems without verification found this OOV while systems with verification failed. If several multigram segmentation variants are generated and searched, the probability that a different variant exists in the lattice increases and gives better chance that the term will be verified.

Similarly to the experiment in section 6.5, the accuracy saturates for 3-best variants. On the other hand, n-best variants slow the search times. If a term has 2 OOV words and we search for 3-best variants, we must search for 6 variants of the term which can be very time expensive.

System	Mgram n-best variants	Verif.	UBTWV		
			ALL	IV	OOV
LSE IVwrdOOVwrdmgram	1	no	0.735	0.757	0.682
LSE IVwrdOOVwrdmgram	1	yes	0.712	0.748	**0.628**
LSE IVwrdOOVwrdmgram	2	no	0.735	0.757	0.682
LSE IVwrdOOVwrdmgram	2	yes	0.719	0.748	0.654
LSE IVwrdOOVwrdmgram	3	no	0.735	0.757	0.682
LSE IVwrdOOVwrdmgram	3	yes	0.722	0.748	**0.663**
LSE IVwrdOOVwrdmgram	5	no	0.735	0.757	0.682
LSE IVwrdOOVwrdmgram	5	yes	0.722	0.748	0.663

Table 8.2: Dependency of the UBTWV accuracy on the n-best multigram segmentation of OOV words. Positive impact on the accuracy of more segmentations of an OOV word to sequences of multigrams can be seen on systems with verification.

8.2.1 Out-of-vocabulary conversion by grapheme-to-phoneme

In all experiments up to now, the OOV word pronunciation variants have been taken from the full 50k WRD dictionary. This is done to obtain the results not influenced by possible errors in generated OOV pronunciations. Here, we applied the *G2P tool* (section 3.5.7) on the OOV word labels and produced the pronunciations which corresponds to real deployment of the system. Only the best pronunciation variant is taken. Then the generated pronunciations are segmented to multigrams (also only the 1-best multigram

segmentation is taken), and searched by the LSE indexing and search tool. The results are summarized in table 8.3. With pronunciations generated by *G2P*, the OOV term accuracy deteriorates by 0.008. We also did an experiment with generating 2-best pronunciation variants per word by the G2P tool. However, this had no effect on the accuracy.

System	Mgram n-best variants	Verif.	UBTWV		
			ALL	IV	OOV
LSE IVwrdOOVwrdmgram LVCSR	1	no	0.735	0.757	0.682
LSE IVwrdOOVwrdmgram LVCSR	1	yes	0.712	0.748	0.628
LSE IVwrdOOVwrdmgram G2P	1	no	**0.733**	**0.757**	**0.675**
LSE IVwrdOOVwrdmgram G2P	1	yes	0.709	0.748	0.620

Table 8.3: Comparison of hybrid system where OOV pronunciations are taken from the LVCSR dictionary (*LSE IVwrdOOVwrdmgram LVCSR*) or generated automatically by *G2P* (*LSE IVwrdOOVwrdmgram G2P*).

Chapter 9

Conclusion and discussions

9.1 Summary

The thesis deals with spoken term detection. The corner stone of this thesis is search of out-of-vocabulary terms which are not present in dictionary of word-based speech recognizer. We investigate into combination of word and subword approaches to get the best search accuracy (especially for out-of-vocabulary words) having the highest search speed and the lowest memory consumptions.

Several systems were described and tested in this thesis. We aimed at evaluation of spoken term detection accuracy. The accuracy was evaluated on 3h of conversational telephone speech. We searched for nearly 400 terms (having up to 4 words) where about one third contains at least one out-of-vocabulary word.

The Upper-Bound-Term-Weighted-Value (UBTWV) was used as the primary evaluation metric. We derived this metric from Term-Weighted-Value (TWV) defined by NIST. The difference is in calibration of terms' scores to one global threshold. The UBTWV shifts the terms confidences to maximize term's TWV for threshold 0. Terms are then pooled and average upper-bound TWV is calculated. By this, we can effectively bypass the calibration of scores and concentrate on the actual system's accuracy.

We also evaluated word accuracy and size of output produced by systems. The size of the output is important from the practical point of view.

The baseline LVCSR system was used to demonstrate the effect of missing words in the vocabulary. We have shown a deterioration of spoken term detection and word recognition accuracy. The baseline recognizer was also used to demonstrate tuning the recognizer to "reasonably best" accuracy. Term confidence estimation techniques were studied as well. We found that *sum of link posteriors* (SOLP) and C_{max} confidences are the best. This fact was also validated for subword and hybrid systems. However, we chose *link posterior* (LP) confidence for majority of experiments due to its simplicity (large difference between LP and SOLP was observed only for phone multigram system). SOLP confidence estimation was however used to validate the important conclusions in this thesis.

The first set of subword systems were the phone systems. We trained the phone system up to phone trigram, but there was no large difference between bigram and trigram language models. We primarily evaluated the approach of producing phone lattices from word lattices published by Witbrock and Hauptmann [WH97]. The conclusion is, that this approach has no advantage for spoken term detection of out-of-vocabulary terms.

The second set of subword systems were phone multigram systems. We adopted the approach of phone multigrams published by Deligne and Bimbot [DB95]. First, we found

the optimal configuration of phone multigram model according to spoken term detection accuracy. We proposed two new multigram models with constraints (nosil and noxwrd). These constrained phone multigrams were found superior to the baseline multigrams. Beside the evaluation of term confidences, we evaluated also the influence of number of out-of-vocabulary term segmentations to multigrams. It was found, that this number of segmentations has significant impact only on the accuracy of out-of-vocabulary term detection. The conclusion is that constrained multigrams significantly overcome standard multigrams and phones. This system is more accurate, faster and produces smaller lattices.

The third set were hybrid word-subword systems. A framework based on WFST was defined for construction of hybrid word-subword language models. We first evaluated "oracle" systems (subword model was substituted by out-of-vocabulary words). We investigated also the dependency of size of lattices and accuracy on parameters of hybrid language model. A hybrid system can achieve higher accuracy than a word system having comparable size of produced lattices.

The first two subword systems "plugged" into hybrid system framework were the phones and constrained multigrams. It was shown, that these hybrid systems did not bring any interesting improvement. The explanation of this was that these subword models were trained on whole sentences and contain "word" priors. Plugged into hybrid, subword model competes with the word model, and decreases the accuracy on in-vocabulary words.

Therefore, we trained phone multigram model only on pronunciation vocabulary. The baseline hybrid system was based on pronunciation multigrams derived from LVCSR dictionary (similarly as proposed by Bazzi [Baz02]). We extended this baseline system further by training the multigram model on large dictionary of out-of-vocabulary words. This system achieved slightly better accuracy. We tested the influence of automatic grapheme-to-phoneme production of pronunciations of out-of-vocabularies on the spoken term detection accuracy. The influence was not significant.

The second extension was incorporation of bigram language model over multigrams in the subword part of the hybrid recognizer. The effect of stronger subword model becomes evident on the accuracy of out-of-vocabulary terms and smaller lattice size.

The hybrid systems were also evaluated with different beam pruning in the decoder. We found performance of in-vocabulary term detection to be instable in this experiment. The hybrid system started to deteriorate compared to word baseline for low beam pruning. Also, we were not able to overcome the best out-of-vocabulary term detection accuracy achieved by the baseline multigram system. The hybrid system was evaluated from the lattice size and computational speed points of view. This should ensure practical applicability of the proposed system. We found that hybrid system can achieve slightly worse accuracy with significant reduction of lattice size and the same speed compared to the combined word and multigram systems. From pure accuracy point of view, this could be considered a failure of the proposed approach, but in our opinion, this drawback is largely compensated by its simplicity and efficiency – keep in mind that in the combination of word and multigram systems, the data must be processed separately by both systems which is much more complicated.

The final test of the hybrid system was in combination with a real indexing and search engine. The advantage of proposed word-multigram hybrid system is that the output hybrid lattices can be indexed by the existing system. There is no need for overlapped phone n-grams that are usually used in phone-based indexing systems.

We have also investigated the verification of terms. The verification is a two step process, where existence of candidates hypothesized by the indexing and search engine is validated in the lattice and more precise confidence is estimated. Our conclusion is

that the search is more accurate without verification. The explanation is that even if the requested path of subword units does not exist in the lattice, the indexing and search engine (that is looking only for sequence of subword units properly ordered in time) has a chance to detect it. We have proven this in experiments searching for more than one possible multigram segmentations of terms – in this case, turning verification in lattice on did not deteriorate the results significantly.

9.2 Future work

The main problem of our hybrid approach is in the requirement of correct scaling of word and subword language models. In the case of incorrectly scaled language models, produced lattices can contain too high or too low number of subword units. The accuracy will not be optimal.

An approach published by Yazgan and Saraclar [YS04a] based on estimation of hybrid language model on hybrid textual corpora could be more robust. There is no need of any scaling of word-subword parts of the LM. The "scaling" parameters are directly estimated on the data. On the other hand, our approach can be used in cases where we do not have much LM training data, or we want to adapt the system to a different domain.

We could build an LM containing several different types of OOV symbol as `<unk-name>`, `<unk-street>`, `<unk-city>` and `<unk-other>` for each domain. Then, each domain-dependent subword language model could be estimated separately. `<unk-name>` on names of people, `<unk-street>` on names of streets, etc. In a new scenario, where the set of names radically differs from the set used for training of the previous `<unk-name>` model, only the new `<unk-name>` would need to be be trained and hybrid network recompiled. As the corpora for the subword language models are in "dictionary" format, building them is easy. The approach based on hybrid textual corpora can not be so easily adapted to the new set of names. The names appearing in the hybrid corpora would need to be substituted by the new names and the whole hybrid language model would need to be retrained.

This work could be also highly beneficial for applications, where adding new words to the system is requested. Having the `<unk-name>`, `<unk-street>`, `<unk-city>`, `<unk-other>` etc. symbols in the recognition network would actually create the necessary place-holders, where appropriate new words could be added off- or even on-line.

It would be also very interesting to evaluate the phone multigram model proposed by Bisany and Ney [BN08] and compare it to our approach. Their multigram model is defined theoretically more correctly and should achieve better results. It would be also interesting to assess the improvement in the hybrid system and in a standalone multigram system.

In the proposed constrained multigrams, we could also improve dealing with the silence, that is currently considered as an independent unit. The silence models (`sil` and `sp`) could be placed to the end of each multigram unit in the same way as it is done in LVCSR.

A big step forward would be to implement a "hybrid-friendly" decoder. It can be difficult to precisely tune hybrid language model parameters to achieve balanced word and subword performance. In the current setup, there is a danger that the hybrid system switches completely to the word or sub-word modes. The decoder should guarantee, for example, at least one token surviving in each part all the time.

From the experimental point of view, it would be interesting to investigate the performance of our approach on out-of-language (OOL) parts of speech – such parts can appear for example in meeting data or broadcast news.

Another future work is linked with using the hybrid system as OOV detector and comparing its performances with the detector based on strong vs. weak posteriors, as it was proposed at JHU workshop in 2007[1] by Hermansky et al. [HBS+07]. Efforts in this direction are already running within the European DIRAC project [ČH10].

[1] http://www.clsp.jhu.edu/ws2007/groups/rmimsr/

Appendix A

Dictionary parameters

In section 7.7.3, we tried to find out, if the accuracy of hybrid system was influenced by the size of the dictionary. A large set of 1.2M word labels was collected from the corpora used for open vocabulary word language model estimation (section 4.3). We performed several steps to clean the word list of "garbage" (words "`AAAAHHHH, 92', etc.`"):

- We omit the WRDunk dictionary. This leads to:
  1. Our OOV words become unseen words for the subword model.
  2. We omit also the IV words so the subword model is not be biased by the IV words.
- We omit words with counts lower or equal than C, where C is a parameter. This filters out the senseless "garbage" words.
- We omit words which have label length shorter or equal than L characters, where L is a parameter. We want to model longer words. Also, the short words are usually "garbage" abbreviations.
- The automatically trained *G2P* tool (section 3.5.7) produces the pronunciation variant and the posterior probability is attached to the variant. We compare the posterior probability to the threshold VP, where VP is a parameter. Pronunciations with probability less than VP are omitted. This should filter out words with "uncertain" pronunciations.

The original size 1.2M of word list is reduced to value in range between 360k words for $L = 2$, $C = 2$, $VP = 0.0$ down to 14k words for $L = 4$, $C = 50$, $VP = 0.9$ depending on the set of parameters L, C and VP.

We "tune" parameters C, L and VP (denoted as *dictionary parameters*) separately and observe their influence on the accuracy. The overall effect of *dictionary parameters* on the accuracy is small (tenths of percent).

The C parameter (minimum count) is set to values 2, 10 and 50 (figures A.1 and A.2). Small accuracy improvement is observed for word accuracy and UBTWV-IV for $C = 50$, on the other hand, significant improvement of UBTWV-OOV is observed for $C = 2$. We selected minimum count parameter $C = 2$ for further experiments according to better accuracy on OOV words.

The L parameter (minimum length) is set to values 2, 3 and 4 (figures A.3 and A.4). No significant accuracy improvement is observed. UBTWV-OOV is slightly better for $L = 4$. We selected minimum length parameter $L = 4$ for further experiments.

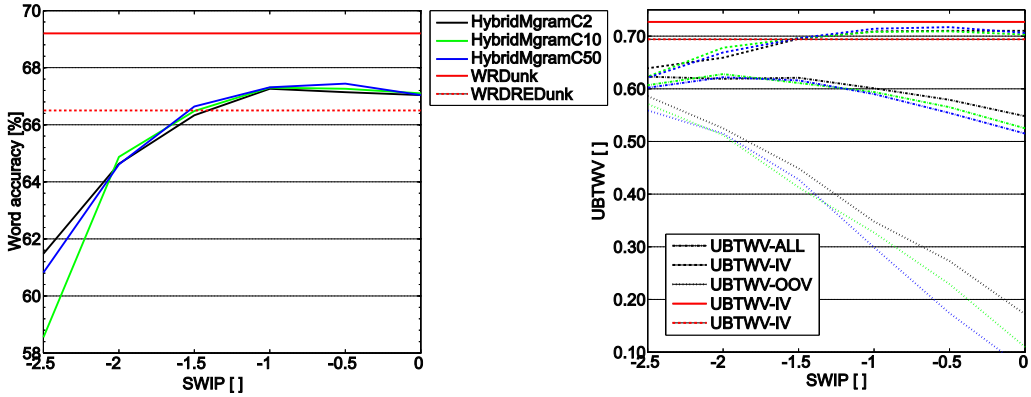


Figure A.1: The dependency of the HybridMgram systems WAC and UBTWV accuracy on the parameter SWIP. The red color denotes the baseline systems WRDunk and *WRDREDunk*. The dictionary parameter C (minimum count) is set to 2, 10 and 50. Fixed parameters: maximum length of multigrams is $l_{mgram} = 5$, multigram pruning $c_0 = 5$, dictionary parameters $L = 3$ and $VP = 0.8$.

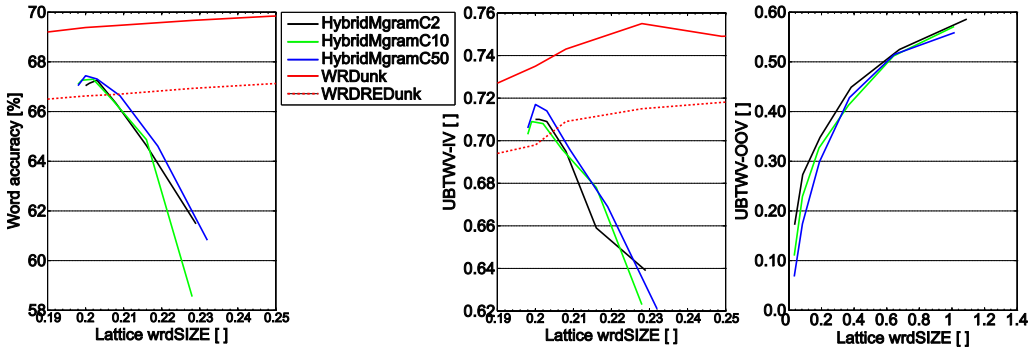


Figure A.2: The dependency of the HybridMgram systems WAC, UBTWV accuracy and the lattice size on parameter SWIP. The red color denotes the baseline systems *WRDunk* and *WRDREDunk*. The dictionary parameter C (minimum count) is set to 2, 10 and 50. Fixed parameters: maximum length of multigrams is $l_{mgram} = 5$, multigram pruning $c_0 = 5$, dictionary parameters $L = 3$ and $VP = 0.8$.

The VP parameter (minimum posterior probability of variant) is set to values 0.1, 0.2 up to 0.9 (figures A.5 and A.6). No consistent and significant improvement is observed. We selected the minimum posterior probability of pronunciation variant $VP = 0.8$ for further experiments.

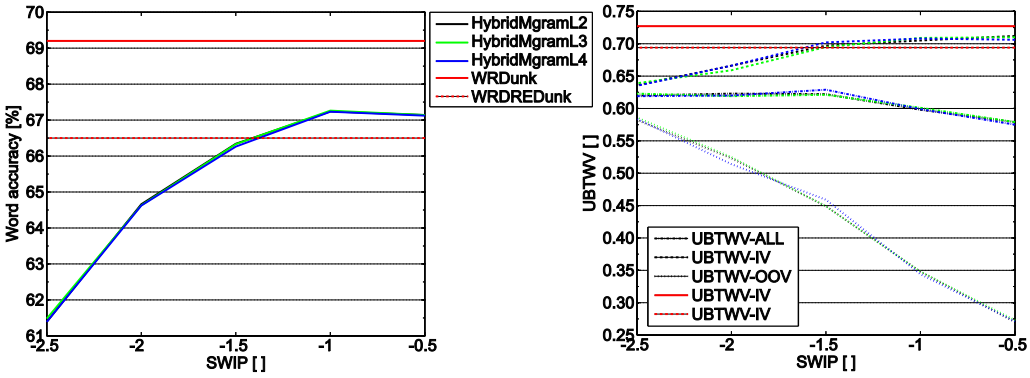


Figure A.3: The dependency of the HybridMgram systems WAC and UBTWV accuracy on the parameter SWIP. The red color denotes the baseline systems WRDunk and WRDREDunk. The dictionary parameter L (minimum length) is set to 2, 3 and 4. Fixed parameters: maximum length of multigrams is $l_{mgram} = 5$, multigram pruning $c_0 = 5$, dictionary parameters $C = 3$ and $VP = 0.8$.

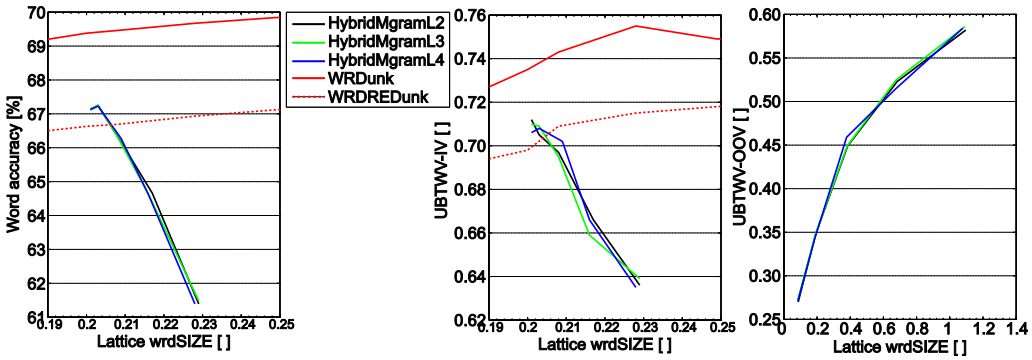


Figure A.4: The dependency of the HybridMgram systems WAC, UBTWV accuracy and the lattice size on parameter SWIP. The red color denotes the baseline systems *WRDunk* and *WRDREDunk*. The dictionary parameter L (minimum length) is set to 2, 3 and 4. Fixed parameters: maximum length of multigrams is $l_{mgram} = 5$, multigram pruning $c_0 = 5$, dictionary parameters $C = 3$ and $VP = 0.8$.

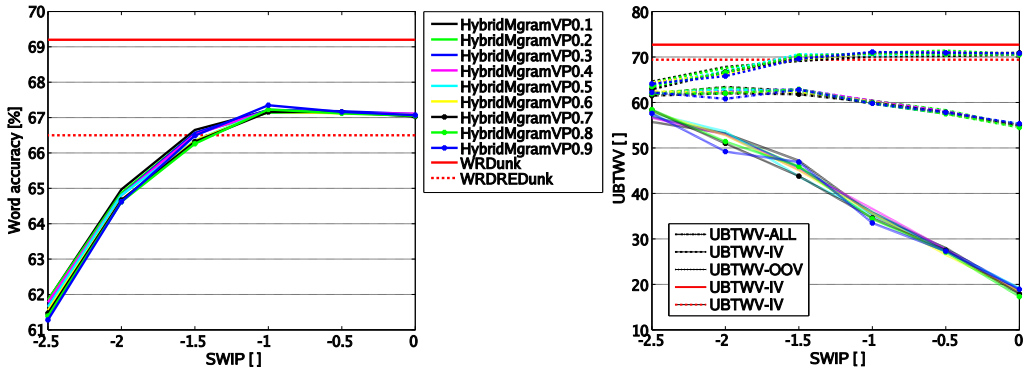


Figure A.5: The dependency of the HybridMgram systems WAC and UBTWV accuracy on the parameter SWIP. The red color denotes the baseline systems WRDunk and WRDREDunk. The dictionary parameter VP (minimum posterior probability of variant) is set to 0.1, 0.2 up to 0.9. Fixed parameters: maximum length of multigrams is $l_{mgram} = 5$, multigram pruning $c_0 = 5$, dictionary parameters $C = 2$ and $L = 4$.

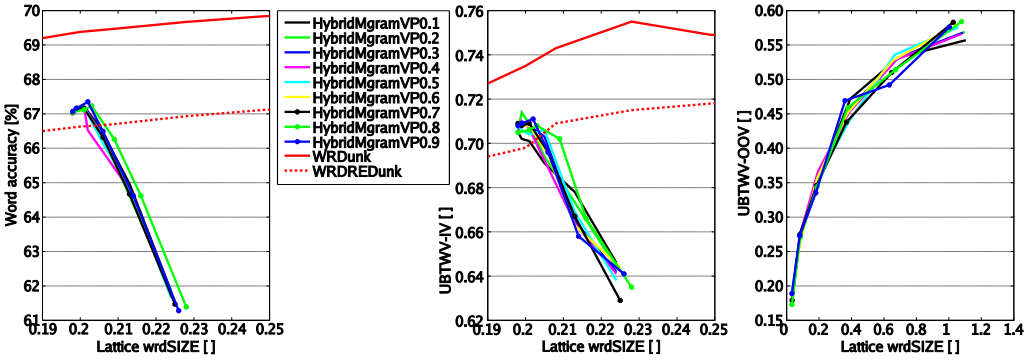


Figure A.6: The dependency of the HybridMgram systems WAC, UBTWV accuracy and the lattice size on parameter SWIP. The red color denotes the baseline systems *WRDunk* and *WRDREDunk*. The dictionary parameter VP (minimum posterior probability of variant) is set to 0.1, 0.2 up to 0.9. Fixed parameters: maximum length of multigrams is $l_{mgram} = 5$, multigram pruning $c_0 = 5$, dictionary parameters $C = 2$ and $L = 4$.

Appendix B

Grapheme-to-phoneme converter

The grapheme to phoneme system used in this thesis works in the following steps. In the training, it takes pronunciation vocabulary (words mapped to phoneme strings) and it produces set of rules. When pronunciation variants of words are needed, the input is word labels (graphemes) and a set of rules. The output is generated pronunciation vocabulary with probabilities assigned to produced variants.

Phoneme grouping

At first, it is expected that every grapheme can be mapped at most to one phoneme. If some words have more graphemes than phonemes, two phonemes are mapped to one pseudo phoneme: `"X"` → `"k s"`, `"U"` → `"y uh"`. So it is necessary to find these phoneme groups. The fastest way to perform this step for any language is to find words that have more phonemes than graphemes. We look at them and derive common phoneme groups.

```
INSECURITY -> ih n s ih k "y uh" r ax t iy
INTRAMURAL -> ih n t r ax m "y uh" r ax l
JANUARY -> jh ae n "y uw" eh r iy

INTOXICATED -> ih n t aa "k s" ax k ey t ax d
KLUX -> k l ah "k s"
LARYNX -> l ae r ih ng "k s"
```

New pseudo phonemes are made using character `*`. The new pseudo phonemes look as: `"k s"` → `"k*s"` or `"y uw"` → `"y*uw"`.

Creating variants

In the second step, all possible mappings of graphemes to phonemes (pseudo phonemes) are produced for every word in the dictionary.

```
D  A  Y  T  O  N
-  -  d  ey t  en
-  d  -  ey t  en
d  -  -  ey t  en
-  d  ey -  t  en
d  -  ey -  t  en
d  ey -  -  t  en
-  d  ey t  -  en
d  -  ey t  -  en
d  ey -  t  -  en  <= We would like to get this in next step
d  ey t  -  -  en
-  d  ey t  en  -
d  -  ey t  en  -
d  ey -  t  en  -
d  ey t  -  en  -
d  ey t  en  -  -
```

Creating rules

In the third step, all possible rewriting rules are generated. A probability is attached to each rule according to how many times it was used. The probabilities of rules are normalized, so that the probability of rules for one grapheme sums to 1.0.

```
A->-     0
A->aa    0.0385445688875768
A->ae    0.129970293558684
A->ah    0.00129158440504087
...
A->w     0.0140782700149455
A->y     0.0010148163182464
A->y*ax  0.00276768086794472
A->y*uw  0.00079340184881082
A->z     0.0116427108511541
A->zh    0.00112552355296419
...
```

Selecting the best variants

In the fourth step, all mapping variants are evaluated for each word. This is done with consideration of having as small rule set as possible. Probabilities of rules are sums for each variant, and the one with the highest score is used. The most probable mapping rules are estimated in this step.

```
D  A  Y  T  O  N
d  ey -  t  -  en
```

In this case, the correct (the most probable) variant is chosen.

Table of rules

We obtained correct grapheme to phoneme mappings in the previous step. These will be used as the training set. Final rewriting rules are derived from the mappings. The probability of a rule is the probability of applying this grapheme to phoneme mapping conversion according to right and left contexts of the given grapheme.

```
|A| => - 0.142501
|A| => aa 0.078082
|A| => ae 0.269577
...
|B| => b 0.955423
...
|A|A => aa 0.450000
|A|A => ae 0.200000
...
|A|D => aa 0.050602
...
O|N|G => - 0.528736
O|N|G => n 0.103448
O|N|G => ng 0.333333
...
+++S|A|DIST+ => ey 1.000000
+++S|A|DISTI => ax 1.000000
...
```

The character + marks the end of a word. This table of rules is stored and can be used for estimation of pronunciation variants.

Using table of rules

This step aims at estimation of pronunciations from input words labels. For every grapheme in the word, we can find the longest possible context in the table of rules. All pronunciation variants (phoneme strings) are generated for each word label (grapheme string). Score is calculated for each variant by multiplying the probabilities of each used rule. The best variant (the highest score) is presented as the estimated pronunciation of the given word label. It is also possible to produce n-best pronunciation variants. The last step is the deletion of the * character defining the pseudo phonemes.

```
GARFIELD        1.0                 g aa r f iy l d
BATTLESHIP      1.0                 b ae t el sh ih p
STANISLAV       0.2371969778976     s t ae en ih s l aa v
STANISLAV       0.1778974221024     s t ae n ih s l aa v
```

Bibliography

[AES01] A. Amir, A. Efrat, and S. Srinivasan. Advances in phonetic word spotting. In *Proceedings of International Conference on Information and Knowledge Management*, pages 580–582, Atlanta, Georgia, USA, 2001.

[AMS04] C. Allauzen, M. Mohri, and M. Saraclar. General indexation of weighted automata – application to spoken utterance retrieval. In *Proceedings of HLT*, 2004.

[AVS08] M. Akbacak, D. Vergyri, and A. Stolcke. Open-vocabulary spoken term detection using graphone-based hybrid recognition system. In *Proceedings of ICASSP*, pages 5240–5243, April 2008.

[Baz02] I. Bazzi. *Modelling Out-of-vocabulary Words for Robust Speech Recognition.* PhD thesis, Massachusetts Institute of Technology. Dept. of Electrical Engineering and Computer Science., 2002.

[BFHC03] Y. Benayed, D. Fohr, J. P. Haton, and G. Chollet. Confidence measures for keyword spotting using support vector machines. In *Proceedings of ICASSP*, volume I, pages 588–591, April 2003.

[BFJ+96] M. G. Brown, J. T. Foote, Gareth J. F. Jones, K. S. Jones, and S. J. Young. Open-vocabulary speech indexing for voice and video mail retrieval. In *Proceedings of ACM Multimedia*, pages 307–316, 1996.

[BG00] I. Bazzi and J. Glass. Modeling out-of-vocabulary words for robust speech recognition. In *Proceeding of ICSLP*, Beijing, 2000.

[Bis06] C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics).* Springer, August 2006.

[BJM83] L. R. Bahl, F. Jelinek, and R. L. Mercer. A maximum likelihood approach to continuous speech recognition. In *IEEE Trans. Pattern Analysis and Machine Inteligence, PAMI-5(2)*, pages 179–190, March 1983.

[BN05a] M. Bisani and H. Ney. Open vocabulary speech recognition with flat hybrid models. In *Proceedings of Interspeech*, pages 725–728, Lisbon, Portugal, September 2005.

[BN05b] M. Bisani and H. Ney. Open vocabulary speech recognition with flat hybrid models. In *Proceedings of Interspeech*, pages 725–728, 2005.

[BN08] M. Bisani and H. Ney. Joint-sequence models for grapheme-to-phoneme conversion. *Speech Communication*, 50(5):434–451, 2008.

[Bri73] J. S. Bridle. An efficient elastic-template method for detecting given words in running speech. In *British Acoustical Society Spring Meeting*, pages 1–4, London, UK, April 1973.

[CCW06] T. H. Chen, B. Chen, and H. M. Wang. On using entropy information to improve posterior probability-based confidence measures. In *Proceedings of ICSLP*, December 2006.

[ČH10] J. Černocký and H. Hermansky. Report on identification of repeatedly occurring out-of-vocabulary words, deliverable no: D2.12, DIRAC project. Technical report, Brno University of Technology, Czech Republic, 2010.

[ČSF+07] J. Černocký, I. Szöke, M. Fapšo, M. Karafiát, L. Burget, J. Kopecký, F. Grézl, P. Schwarz, O. Glembek, I. Oparin, P. Smrž, and P. Matějka. Search in Speech for Public Security and Defense. In *Proceedings of IEEE Workshop on Signal Processing Applications for Public Security and Forensics, SAFE*, pages 1–7. IEEE Signal Processing Society, 2007.

[DB95] S. Deligne and F. Bimbot. Language Modeling by Variable Length Sequences: Theoretical Formulation and Evaluation of Multigrams. In *Proceedings of ICASSP*, pages 169–172, Detroit, MI, USA, 1995.

[DFR98] S. Dharanipragada, M. Franz, and S. Roukos. Audio-indexing for broadcast news. In *Proceedings of Text REtrieval Conference*, pages 63–67, 1998.

[DP73] P. B. Denes and E. N. Pinson. *The Speech Chain: Physics and Biology of Spoken Language*. W. H. Freeman & Co Ltd, 2nd revised edition (27 april 1993) edition, 1973.

[FAD06] J. Fiscus, J. Ajot, and G. Doddington. The spoken term detection (STD) 2006 evaluation plan. Technical report, National Institute of Standards and Technology (NIST) USA, September 2006.

[FAGD07] J. Fiscus, J. Ajot, J. Garofolo, and G. Doddington. Results of the 2006 spoken term detection evaluation. In *The 2007 Special Interest Group on Information Retrieval (SIGIR-07) Workshop in Searching Spontaneous Conversational Speech*, July 2007.

[Fap07] M. Fapšo. Search in speech data. Master's thesis, Brno University of Technology, Faculty of Information Technology, Czech Republic, June 2007.

[Gal03] L. Galescu. Recognition of out-of-vocabulary words with sub-lexical language models. In *Proceedings of Eurospeech*, pages 249–252, Geneva, Switzerland, September 2003.

[GAV00] J. S. Garofolo, C. G. P. Auzanne, and E. M. Voorhees. The TREC spoken document retrieval track: A success story. Technical report, National Institute of Standards and Technology (NIST) USA, 2000.

[GKKČ07] F. Grézl, M. Karafiát, S. Kontár, and J. Černocký. Probabilistic and bottleneck features for LVCSR of meetings. In *Proceedings of ICASSP*, pages 757–760. IEEE Signal Processing Society, 2007.

[GLV00] J. Garofolo, J. Lard, and E. M. Voorhees. 2000 TREC-9: Spoken document retrieval track. Technical report, National Institute of Standards and Technology (NIST) USA, November 2000.

[GMRO99] C. Garcia-Mateo, W. Reichl, and S. Ortmanns. On combining confidence measures in HMM-based speech recognizers. In *Proceedings of ASRU*, pages 201–204, 1999.

[GNR92] H. Gish, K. Ng, and J. R Rohlicek. Secondary processing using speech segments for an HMM word spotting system. In *Proceedings of ICSLP*, pages 17–20, Banff, Canada, 1992.

[Gré07] F. Grézl. *TRAP-based Probabilistic Features for Automatic Speech Recognition.* PhD thesis, Brno University of Technology, Faculty of Information Technology, Brno, Czech Republic, 2007.

[HAH00] X. Huang, A. Acero, and H.W. Hon. *Spoken Language Processing.* Microsoft, Redmond, WA, USA, 1 edition, October 2000.

[HBD+05] T. Hain, L. Burget, J. Dines, G. Garau, M. Karafiát, M. Lincoln, I. McCowan, D. Moore, V. Wan, R. Ordelman, and S. Renals. The 2005 AMI system for the transcription of speech in meetings. In *Proceedings of Rich Transcription 2005 Spring Meeting Recognition Evaluation Workshop*, Edinburgh, UK, July 2005.

[HBD+06] T. Hain, L. Burget, J. Dines, G. Garau, M. Karafiát, and V. Wan. The AMI Meeting Transcription System. In *Proceedings of NIST Rich Transcription 2006 Spring Meeting Recognition Evaluation Worskhop*, page 12. National Institute of Standards and Technology, 2006.

[HBS+07] H. Hermansky, L. Burget, P. Schwarz, P. Matějka, M. Hannemann, A. Rastrow, C. White, S. Khudanpur, and J. Černocký. Recovery from model inconsistency in multilingual speech recognition, report from JHU workshop 2007. Technical report, Johns Hopkins University, USA, 2007.

[HHZ+05] J. H. L. Hansen, R. Huang, B. Zhou, M. Seadle, Jr J. R. Deller, A. R. Gurijala, M. Kurimo, and P. Angkititrakul. Speechfind: Advances in spoken document retrieval for a national gallery of the spoken word. In *IEEE Transactions on Speech and Audio Processing*, volume 13, pages 712–727. IEEE, September 2005.

[HW85] A. L. Higgins and R. E. Wohlford. Keyword recognition using template concatenation. In *Proceedings of ICASSP*, pages 1233–1236, April 1985.

[HWB+07] T. Hain, V. Wan, L. Burget, M. Karafiát, J. Dines, J. Vepa, G. Garau, and M. Lincoln. The ami system for the transcription of speech in meetings. In *Proceedings of ICASSP*, pages 357–360. IEEE Signal Processing Society, 2007.

[Jam95] D. A. James. *The Application of Classical Information Retrieval Techniques to Spoken Documents.* PhD thesis, Downing College, Cambridge, United Kingdom, February 1995.

[JFJY96] G. J. F. James, J. T. Foote, K. Sparck Jones, and S. J. Young. Retrieving spoken documents by combining multiple index sources. In *Proceedings of SIGIR*, pages 30–38, Zurich, Switzerland, August 1996.

[Jia05] H. Jiang. Confidence measures for speech recognition: A survey. In *Speech Communication*, volume 45, pages 455–470, 2005.

[JRH97] J. Junkawitsch, G. Ruske, and H. Höge. Efficient methods for detectiong keywords in continuous speech. In *Proceedings of Eurospeech*, volume 1, pages 259–262, Rhodes, Greece, September 1997.

[KGS+06] M. Karafiát, F. Grézl, P. Schwarz, L. Burget, and J. Černocký. Robust heteroscedastic linear discriminant analysis and LCRC posterior features in meeting data recognition. *Lecture Notes in Computer Science*, 2006(4299):275–284, 2006.

[KL99] M. W. Koo and S. J. Lee. An utterance verification system based on subword modeling for a vocabulary independent speech recognition system. In *Proceedings of Eurospeech*, volume 1, pages 287–290, Budapest, September 1999.

[KLC02] J. Kim, J. Lee, and S. Choi. Hybrid confidence measure for domain-specific keyword spotting. In *Proceedings of The 15th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems*, pages 736–745, London, UK, 2002. Springer-Verlag.

[KLJ98] T. Kawahara, C. H. Lee, and B. H. Juang. Flexible speech understanding based on combined key-phrase detection and verification. *IEEE Transactions on Speech and Audio Processing*, 6(6):558–568, November 1998.

[KLJ01] M. W. Koo, C. H. Lee, and B. H. Juang. Speech recognition and utterance verification based on a generalized confidence score. *IEEE Transactions on Speech and Audio Processing*, 9(8):821–832, November 2001.

[KVBB06] H. Ketabdar, J. Vepa, S. Bengio, and H. Bourlard. Posterior based keyword spotting with a priori thresholds. In *Proceedings of ICSLP*, Pittsburgh, USA, 2006. IDIAP-RR 06-67.

[KY96] K. M. Knill and S. J. Young. Fast implementation methods for Viterbi-based word-spotting. In *Proceedings of ICASSP*, pages 1–4, 1996.

[Le04] A. Le. NIST's workshop presentation on STT, NIST RT 2004, November 2004.

[LMD02] B. Logan, P. Moreno, and O. Deshmukh. Word and sub-word indexing approaches for reducing the effect of OOV queries on spoken audio. In *Proceedings of Human Language Technology Conference*, pages 31–35, San Diego, California, USA, March 2002.

[LT02] B. Logan and J. M. Van Thong. Confusion-based query expansion for OOV words in spoken document retrieval. In *Proceedings of ICSLP*, September 2002.

[LTI05] S. Lee, K. Tanaka, and Y. Itoh. Combining multiple subword representations for open-vocabulary spoken document retrieval. In *Proceedings of ICASSP*, volume 1, pages 505–508, 2005.

[MJ99] N. Moreau and D. Jouvet. Use of a confidence measure based on frame level likelihood ratios for the rejection of incorrect data. In *Proceedings of Eurospeech*, volume 1, pages 291–294, Budapest, September 1999.

[MPR08] M. Mohri, F. Pereira, and M. Riley. *Speech Recognition with Weighted Finite-state Transducers*. Springer Handbook on Speech Processing and Speech Communication, Part E: Speech recognition. Springer-Verlag, Heidelberg, Germany, 2008.

[Ng00a] K. Ng. Information fusion for spoken document retrieval. In *Proceedings of ICASSP*, 2000.

[Ng00b] K. Ng. *Subword-Based Approaches for Spoken Document Retrieval*. PhD thesis, Massachusetts Institute of Technology, USA, February 2000.

[NIS91] *"The Road Rally Word-Spotting Corpora (RDRALLY1), NIST Speech Disc 6-1.1"*. September 1991.

[PČB+00] P. Polák, J. Černocký, J. Boudy, K. Choukri, H. van den Heuvel, K. Vicsi, A. Virag, R. Siemund, W. Majewski, P. Staroniewicz, and H. Tropf. Speechdat(e) – eastern European telephone speech databases. 2000.

[PHG05] A. Park, T. J. Hazen, and J. R. Glass. Automatic processing of audio lectures for information retrieval: Vocabulary selection and language modeling. In *Proceedings of ICASSP*, volume 1, pages 497–500, 2005.

[Pov03] D. Povey. *Discriminative Training for Large Vocabulary Speech Recognition*. PhD thesis, Cambridge University Engineering Dept., 2003.

[PS08] S. Parlak and M. Saraclar. Spoken term detection for Turkish broadcast news. In *Proceedings of ICASSP*, pages 5244–5248, April 2008.

[PSG98] C. Pao, P. Schmid, and J. Glass. Confidence scoring for speech understanding systems. In *These Proceedings, Syndey, Australia*, November 1998.

[PSPH08] J. Pinto, I. Szöke, S. R. M. Prasanna, and H. Heřmanský. Fast approximate spoken term detection from sequence of phonemes. In *The 31st Annual International ACM SIGIR Conference 20-24 July 2008, Singapore*, pages 28–33, 2008.

[RCAC96] Z. Rivlin, M. Cohen, V. Abrash, and T. Chung. A phone-dependent confidence measure for utterance rejection. In *Proceedings of ICASSP*, pages 515–518, Atlanta, GA, USA, 1996.

[RJ97] S. E. Robertson and K. S. Jones. Simple proven approaches to text retrieval. In *Tech. Rep. TR356, Cambridge University Computer Laboratory*, 1997.

[RLJ97] M. G. Rahim, C. H. Lee, and B. H. Juang. Discriminative utterance verification for connected digits recognition. *IEEE Transactions on Speech and Audio Processing*, 5(3):266–277, May 1997.

[Ros92] R. C. Rose. Discriminant wordspotting techniques for rejecting non-vocabulary utterances in unconstrained speech. In *Proceedings of ICASSP*, volume II, pages 105–108, 1992.

[RP90] R. C. Rose and D. B. Paul. A hidden Markov model based keyword recognition system. In *Proceedings of ICASSP*, volume 2, pages 129–132, Albuquerque, New Mexico, USA, April 1990.

[RRRG89] J. R. Rohlicek, W. Russell, S. Roukos, and H. Gish. Continuous hidden Markov modeling for speaker-independent word spotting. In *Proceedings of ICASSP*, volume 1, pages 627–630, Glasgow, UK, May 1989.

[SFL+08] I Szöke, M. Fapšo, M. Karafiát L., Burget, F. Grézl, P. Schwarz, O. Glembek, P. Matějka, J. Kopecký, and J. Černocký. Spoken term detection system based on combination of LVCSR and phonetic search. *Lecture Notes in Computer Science*, 2008(4892):237–247, 2008.

[Sie99] M. A. Siegler. *Integration of Continuous Speech Recognition and Information Retrieval for Mutually Optimal Performance*. PhD thesis, Carnegie Mellon University, USA, 1999.

[SL96] R. A. Sukkar and C. H. Lee. Vocabulary independent discriminative utterance verification for nonkeyword rejection in subword based speech recognition. *IEEE Transactions on Speech and Audio Processing*, 4(6):420–429, November 1996.

[SMČ04] P. Schwarz, P. Matějka, and J. Černocký. Towards lower error rates in phoneme recognition. *Lecture Notes in Computer Science*, 2004(3206):465–472, 2004.

[SS04] M. Saraclar and R. Sproat. Lattice-based search for spoken utterance retrieval. In *Proceedings of Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL)*, Boston, Massachusetts, USA, May 2004.

[SSB+05] I. Szöke, P. Schwarz, L. Burget, M. Karafiát, and J. Černocký. Phoneme based acoustics keyword spotting in informal continuous speech. In *Proceedings of RADIOELEKTRONIKA*, Brno, Czech Republic, May 2005.

[SSJ96] A. R. Setlur, R. A. Sukkar, and J. Jacob. Correcting recognition errors via discriminative utterance verification. In *Proceedings of ICSLP*, volume 2, pages 602–605, Philadelphia, PA, USA, 1996.

[SV05] M. C. Silaghi and R. Vargiya. A new evaluation criteria for keyword spotting techniques and a new algorithm. In *Proceedings of Interspeech*, Lisbon, Portugal, September 2005.

[TGT01] B. T. Tan, Y. Gu, and T. Thomas. Word level confidence measures using N-best sub-hypotheses likelihood ratio. In *Proceedings of Eurospeech*, September 2001.

[VH98] E. M. Voorhees and D. Herman. Overview of the seventh text retrieval conference (TREC-7. Technical report, National Institute of Standards and Technology (NIST) USA, November 1998.

[VH01] E. Voohrees and D. Harman. NIST special publication 500-249: The ninth text retrieval conference (TREC 9). August 2001.

[Wan10] D. Wang. *Out-of-vocabulary Spoken Term Detection*. PhD thesis, Institute for Communicating and Collaborative Systems, School of Informatics, University of Edinburgh, 2010.

[Wec98] M. Wechsler. *Spoken Document Retrieval Based on Phoneme Recognition*. PhD thesis, Swiss Federal Institute of Technology, Zurich, 1998.

[Wei95] M. Weintraub. LVCSR log-likelihood ratio scoring for keyword spotting. In *Proceedings of ICASSP*, pages 297–300, Detriot, USA, 1995.

[WH97] M. J. Witbrock and A. G. Hauptmann. Using words and phonetic strings for efficient information retrieval from imperfectly transcribed spoken documents. In *Proceedings of The Second ACM International Conference on Digital Libraries*, pages 30–35, Philadelphia, Pennsylvania, USA, 1997.

[WJJJ00] P. C. Woodland, S. E. Johnson, P. Jourlin, and K. S. Jones. Effects of out-of-vocabulary words in spoken document retrieval. In *Proceedings 18th Annual International ACM-SIGIR*, pages 372–374, Athens, Greece, 2000.

[WMN99] F. Wessel, K. Macherey, and H. Ney. A comparison of word graph and N-best list based confidence measures. In *Proceedings of Eurospeech*, volume 1, pages 315–318, Budapest, September 1999.

[WSMN01] F. Wessel, R. Schluter, K. Macherey, and H. Ney. Confidence measures for large vocabulary continuous speech recognition. *IEEE Trans. Speech and Audio Processing*, 9(3), March 2001.

[You99] S. Young. *The HTK Book*. Entropics Ltd., 1999.

[YS04a] A. Yazgan and M. Saraclar. Hybrid language models for out of vocabulary word detection in large vocabulary conversational speech recognition. In *Proceedings of ICASSP*, volume 1, pages 745–748, May 2004.

[YS04b] P. Yu and F. Seide. A hybrid word/phoneme-based approach for improved vocabulary-independent search in spontaneous speech. In *Proceedings of ICSLP*, volume 1, pages 895–898, 2004.

[YZS06] P. Yu, D. Zhang, and F. Seide. Maximum entropy based normalization of word posteriors for phonetic and LVCSR lattice search. In *Proceedings of ICASSP*, pages 965–968, May 2006.