# Interactive Approaches to Video Lecture Assessment

Der Technischen Fakultät der
Universität Erlangen–Nürnberg

zur Erlangung des Grades

DOKTOR–INGENIEUR

vorgelegt von

Korbinian Thomas Riedhammer

Erlangen — 2012

Deutscher Titel:
Interaktive Methoden zur Inhaltserschließung von Vorlesungsvideos

Als Dissertation genehmigt von der
Technischen Fakultät der
Universität Erlangen-Nürnberg

| | |
|---|---|
| Tag der Einreichung: | 06.07.2012 |
| Tag der Promotion: | 13.08.2012 |
| Dekanin: | Prof. Dr.-Ing. habil. M. Merklein |
| Berichterstatter: | Prof. Dr.-Ing. habil. E. Nöth |
| | Prof. Dr. N. Morgan |

## Acknowledgments

A number of people supported and accompanied me on this long journey to finish this thesis, and I would like to take the chance to thank them for their patience and effort.

I am deeply indebted to Prof. Dr.-Ing. habil. Elmar Nöth who mentored and advocated my scientific career from the early beginning on. In the second semester of my undergraduate studies, he eliminated my doubts if computer science was the right choice for me by infecting me with his enthusiasm about speech processing. He later supervised my student and diploma theses and introduced me to the scientific world. I was fortunate to learn how to do research, to publish the results, to present at conferences, and to close this scientific circle of life by writing new research proposals. I am, however, especially grateful that Elmar not only funded my work through several different projects but also supported me in exploring my personal research interests. Without that freedom, this thesis would not exist.

I would like to express my sincere thanks to Prof. Dr. Nelson Morgan from the University of California at Berkeley and head of the speech group at the International Computer Science Institute (ICSI). We first met in 2008 when I started to work on speech summarization at ICSI. He advised my research and further supported me by funding trips to present at various international conferences. In spring 2010, Morgan again welcomed me to his speech group and supervised my research on robust speech recognition. I am very grateful that he supported my application for a post-doctoral scholarship at ICSI that I was fortunate to receive and start this October.

I cannot be thankful enough for Elmar and Morgan to serve as reviewers to the committee and to provide their reviews on such short notice. Their timely response allowed me to get through the process in a ridiculous short amount of time and to have a seamless transition from Erlangen to my post-doctoral stay at ICSI.

I owe sincere and earnest thankfulness to Tobias Bocklet, a true friend with whom I share all the delights and frustrations in our academic and personal life since our undergraduate studies. The numerous joint publications and subsequent travels to conferences are an impressive testimony of the good teamwork we shared. I would like to thank him for the fruitful discussions and his comments that greatly benefited this thesis. I also wish to thank Benoit Favre for his comments on this thesis, especially on the chapters that cover natural language processing. When we first met in 2008 at ICSI, I did not foresee that our joint work would evolve into a warm friendship and long-term collaboration.

I would like to thank my colleagues in the speech group at the Pattern Recognition Lab for their comments and support, in particular Anton Batliner, Stefan Steidl, Tino Haderlein and Florian Hönig, as well as our student Martin Gropp. I am further indebted to Prof. Dr.-Ing. Joachim Hornegger for supporting me with various scholarship and grant applications, and for recording his lecture series. Lastly, I would like to thank my family and friends that help me to find my way and share the ups and downs enroute.

**Abstract**

A growing number of universities and other educational institutions record videos of regularly scheduled classes and lectures to provide students with additional resources for their study. However, the video alone is not necessarily the same than a carefully prepared educational video. The main issue is that they are typically not post-processed in an editorial sense. That is, the videos often contain longer periods of silence or inactivity, unnecessary repetitions, spontaneous interaction with students, or even corrections of prior false statements or mistakes. Furthermore, there is often no summary or table of contents of the video, unlike with educational videos that supplement a certain curriculum and are well scripted and edited. Thus, the plain recording of a lecture is a good start but far from a good e-learning resource.

This thesis describes a system that can close the gap between a plain video recording and useful e-learning resource by producing automatic summaries and providing an interactive lecture browser that can visualize automatically extracted key phrases and their importance on an augmented time line. The lecture browser depends on four tasks: automatic speech recognition, automatic extraction and ranking of key phrases, extractive speech summarization, and the visualization of the phrases and their salience. These tasks as well as the contribution to the state of the art are described in detail and evaluated on a newly acquired corpus of academic spoken English, the LMELectures. A first user study shows that students using the lecture browser can solve a topic localization task about 29 % faster than students that are provided with the video only.

**Kurzdarstellung**

Immer mehr Universitäten und andere Bildungseinrichtungen lassen Vorlesungen und ähnliche Veranstaltungen auf Video aufzeichnen um diese später den Studenten als zusätzliches Studienmaterial bereit zu stellen. Allerdings entspricht die bloße Aufzeichnung nicht notwendigerweise einem sorgfältig vorbereitetem Lehrfilm, vor allem da diese Videos in der Regel nicht redaktionell aufbereitet werden und daher oft längere Abschnitte von Stille oder Inaktivität, unnötige Wiederholungen, spontane Interaktion mit Studenten, oder Berichtigungen von vorangegangen Missverständnissen oder Fehlern enthalten. Weiterhin sind die Aufzeichnungen oft nicht mit Zusammenfassungen oder Inhaltsverzeichnissen versehen im Gegensatz zu Lehrfilmen welche Teil eines Lehrplans sind und oft aufwendig aufbereitet werden. Daher sind unbearbeitete Videoaufzeichnungen zwar ein guter Anfang, aber noch weit entfernt von sinnvoll verwendbaren E-learning Material.

Die vorliegende Arbeit beschreibt ein System das die Lücke zwischen einem nicht aufbereitetem Video und für E-learning geeignetem Material schließen kann indem es automatische Zusammenfassungen generiert und einen interaktiven Browser für Vorlesungsaufzeichnungen bereit stellt, der automatisch extrahierte Schlüsselphrasen sowie deren Relevanz grafisch darstellt. Der Browser greift bei der Datenanalyse auf vier Teilschritte zurück: Spracherkennung, Extraktion und Ranking von Schlüsselphrasen, extraktive Zusammenfassung, sowie die Visualisierung der Schlüsselphrasen. Diese vier Teilschritte werden erläutert sowie der Beitrag zum aktuellen Stand der Forschung aufgezeigt; die Validität der Verfahren wird anhand eines neu aufgezeichneten Korpus von gesprochenem Englisch aus dem akademischen Umfeld, den LMELectures, bestätigt. Eine erste Benutzerstudie zeigt, dass Studenten mit Hilfe des Browsers Themengebiete etwa 29 % schneller finden können als andere, welche nur die Aufzeichnung zur Verfügung haben.

# Contents

# Chapter 1

# Introduction

> *"Books will soon be obsolete in the public schools. Scholars will be instructed through the eye. It is possible to teach every branch of human knowledge with the motion picture. Our school system will be completely changed inside of ten years."*
>
> Thomas A. Edison, Summer 1913[1]

Edison's prediction did not (yet) come true, but much has changed since.

## 1.1 The Evolution of E-Learning

The term *e-learning* is often associated with futuristic ideas such as virtual class rooms with avatars or holographic instructors driven by artificial intelligence, something "with internet," or at least "something with computers." But for sure with something remotely digital. *E-learning* might be a relatively recent buzzword but the idea of distant learning using audio and video material goes back to the early 1900s. With the invention of the sound motion picture, educational films found their way into the class rooms, a movement labeled "visual instruction," "visual education," or later "audio visual instruction" [Saet 68].

In the United States, the first catalog of instructional films was published in 1910 [Reis 87]. In the 1920s, the early days of radio, stations also started to broadcast educational content. Along the great success of television broadcasting from the 1940s on, more and more institutions and universities began to offer classes and courses.

In post-World-War-II Germany, the distant learning movement picked up in 1966, when the State of Hessen (West Germany) secretary of education, Ernst Schütte, suggested that the public broadcast service offers a correspondence course to qualify teachers in social studies. Up to now, the *Hessischer Rundfunk* maintains a huge archive of educational audio recordings and still offers courses in the "Funkkolleg" program[2] [Grev 98]. Only a year later, in 1967, the public television of the State

[1] July 9, 1913. *The New York Dramatic Mirror*, The Evolution of the Motion Picture: VI – Looking into the Future with Thomas A. Edison by Frederick James Smith, Page 24, Column 3, New York.

[2] `http://www.funkkolleg.de`

of Bavaria (West Germany) began to broadcast educational films in the "Telekolleg" program[3], in the context of a distant learning high school diploma course. Beside the audio and video program which is accessible to everyone, enrolled students can obtain lecture notes and assignments. Study-days and exams are required to earn a degree.

Over the years, numerous programs have been established, covering both basic and higher education. In 1974, the Fernuniversität Hagen was established in Hagen, West Germany, which is up to now the first and only German public university offering distant teaching degree programs. In 2000, the Bavarian universities founded the "Virtuelle Hochschule Bayern"[4] (Virtual University of Bavaria), which offers online courses that are accredited at every Bavarian university. The offered material includes also audio and video material in addition to lecture notes and assignments. These distant learning materials are prepared and designed for that specific purpose, and are, in case of degree programs, maintained by teachers that also supervise the students. But following technological advances such as high-definition digital cameras, affordable mass-storage, omnipresent high-speed internet connections and cellular phones more powerful than a 2000's desktop computer, there is a new trend to blur the border between distant and regular teaching, especially at universities: some professors have their lectures recorded on audio or video and make them available to the students. While this is convenient for students that missed a class or want to recap parts of it, a mere recording of a lecture is not necessarily the same as a carefully prepared educational video.

The main issue with these recordings is that they are typically not post-processed in an editorial sense. That is, the videos often contain longer periods of silence or inactivity, *e.g.*, if the teacher writes or cleans the blackboard, unnecessary repetitions, or even corrections of prior false statements or mistakes. Furthermore, there is often no summary or table of contents of the video, unlike with educational videos that supplement a certain curriculum. Thus, the plain recording of a lecture is a good start — but far from a good e-learning resource.

This thesis describes a system that can close this gap between a plain recording and a useful e-learning resource by producing automatic summaries and showing key phrases and their importance on an augmented time line.

## 1.2 Existing Video Lecture Platforms

Before going into details of the system that is described in this thesis, this section gives an overview of existing video lecture platforms, most of them publicly accessible. This is, of course, not a complete list but a compilation of *types* of platforms to illustrate the current state of the art. Although most of the examples are hosted by universities and often result from or are linked to research projects, there is also a growing number of commercial vendors that market all-inclusive solutions including recording, storage and distribution of the videos. All platforms have a basic catalog function and a video display, but differ greatly in terms of design, usability and additional features.

[3] `http://www.br.de/telekolleg`
[4] `http://www.vhb.org/`

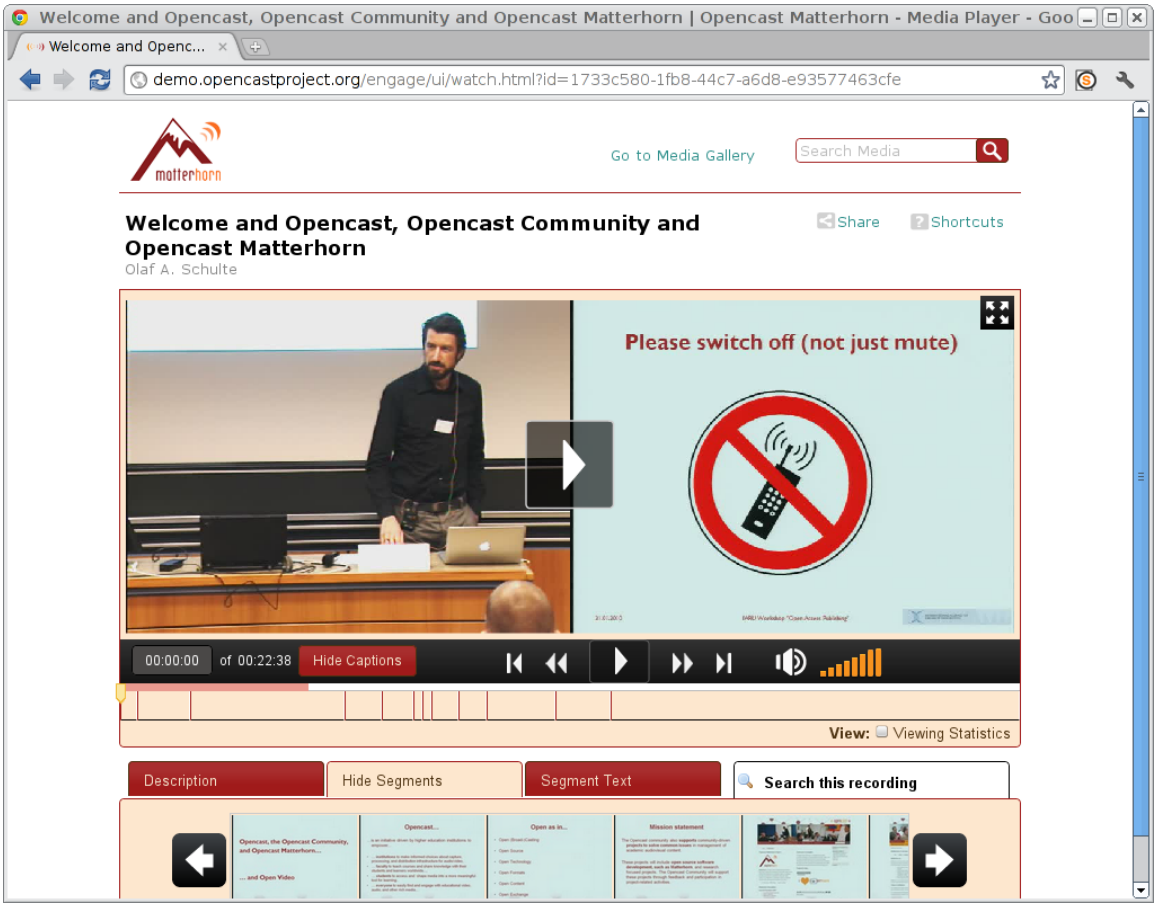



Figure 1.1: Screenshot of the OpenCast Matterhorn video lecture interface (retrieved Feb. 14, 2012).

**OpenCast Matterhorn** – `http://opencast.org/matterhorn` The *OpenCast Matterhorn* platform is a free, open-source, community-driven project that provides an infrastructure and tools to record, manage and provide video lectures. The videos can be manually annotated with categories, subtitles, tags and alike. If PDF slides are available, the video can be segmented into respective chunks and accompanied by text extracted from the slides. On the consumer side, the user interface (*cf.* Fig. 1.1) is primarily a video player with some displays for the annotations. Possible interactions include clicking on the slides to seek the time in the video where it appears, or a text search in the presentation slides.

The OpenCast community was initiated in 2007 by the University of California at Berkeley and is supported by numerous institutions and companies. Currently, 13 universities from all over the world including the University of Erlangen-Nuremberg adopted Matterhorn to host their video lectures. At this stage, the Matterhorn project is focused on providing a solid and efficient infrastructure which is a challenging task on its own: The acquisition, post-processing and storage of hundreds of gigabytes of data needs to be put on a solid base. Due to its openness, Matterhorn may be extended by automatic annotations such as speech recognition, topic and key word extraction.


Figure 1.2: Screenshot of the video lecture catalog of the University of Erlangen-Nuremberg (retrieved Feb. 14, 2012).

**FAU Videoportal** – `http://video.uni-erlangen.de` The Regionales Rechenzentrum Erlangen (RRZE), closely affiliated with the University of Erlangen-Nuremberg (FAU), maintains a growing archive of video lectures. The lectures are recorded either in a special multi-media studio using several cameras and microphones, or using portable equipment in the class room. Each video can be annotated with a description, key words and topics.

The infrastructure is implemented on top of OpenCast Matterhorn, however, the presentation component is reduced to only the video without any further material (*cf.* Fig. 1.2). The recordings are grouped in courses, and indexed by course, semester and instructor. The lecturer controls whether the content access is partly open or restricted to enrolled students of the FAU. In some cases, the videos are only available to students that signed up for that specific class.

Up to now, the platform is limited to video data. The inclusion of further material such as assignments or lecture notes is not available. The videos are offered in different quality and media encoding. Depending on the recording location, the slides may or may not be visible. If the lecture was recorded in the multi-media studio, a video is available showing both the presenter and a magnified version of the currently displayed slide.

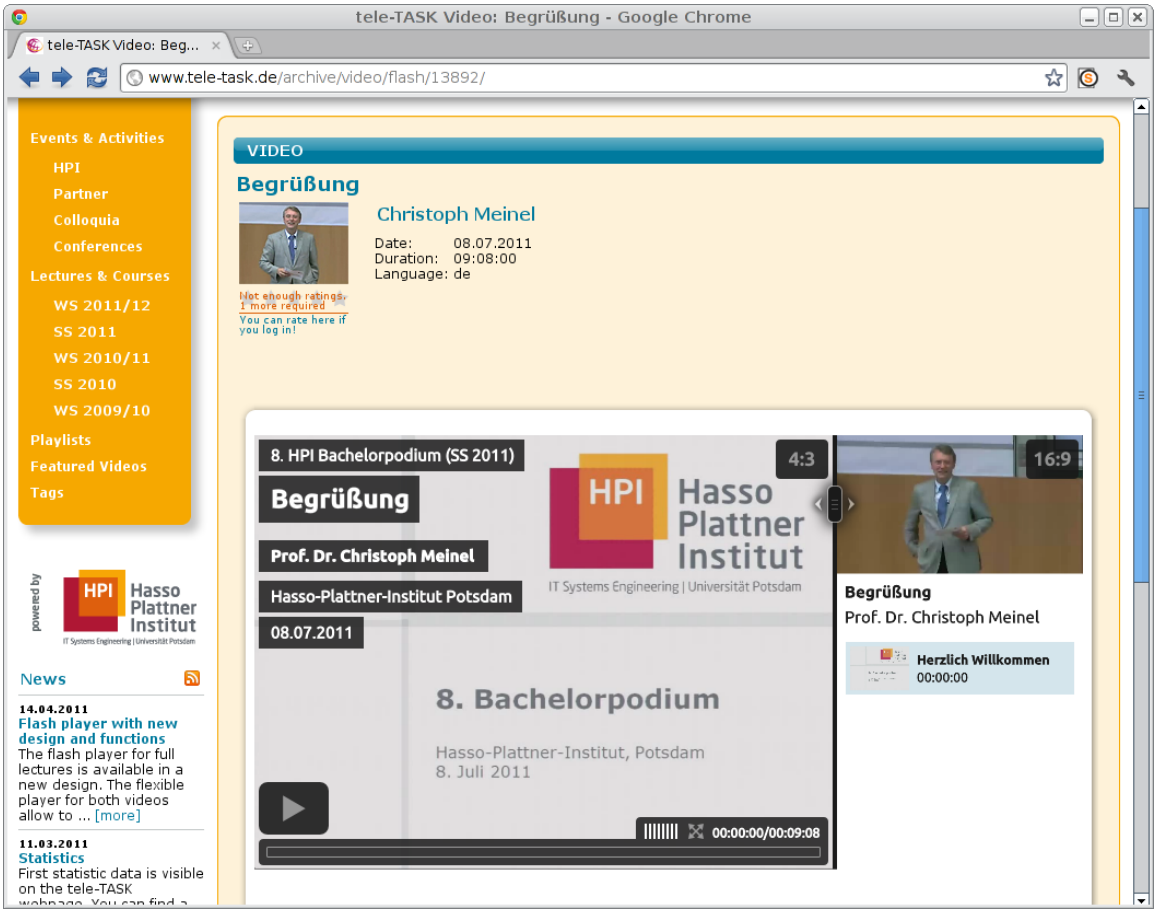



Figure 1.3: Screenshot of the tele-TASK video lecture interface (retrieved Feb. 15, 2012).

**HPI tele-TASK** – `http://www.tele-task.de` The *tele-Teaching Anywhere Solution Kit (tele-TASK)* platform is developed and maintained by the Hasso-Plattner-Institut (HPI). Similar to Matterhorn, it focuses on the acquisition, storage, organization and presentation of the videos. Fig. 1.3 shows a screen shot of the browsing interface. The main distinction to other platforms is the video pane which is divided in three regions: the currently displayed slide (left), a view of the presenter (top right), and a navigation aid (bottom right). The black slider is used to split the available area between the slide display and the speaker. The slider can be used to give more room to either the slides or the presenter, depending on the taste of the user.

Another key feature of tele-TASK is the possibility to export the videos and the browsing software to a medium such as a DVD or a portable hard disk. This is an interesting option for institutions that want to offer an alternative to a traditional textbook based study without depending on a broad-band internet connection.

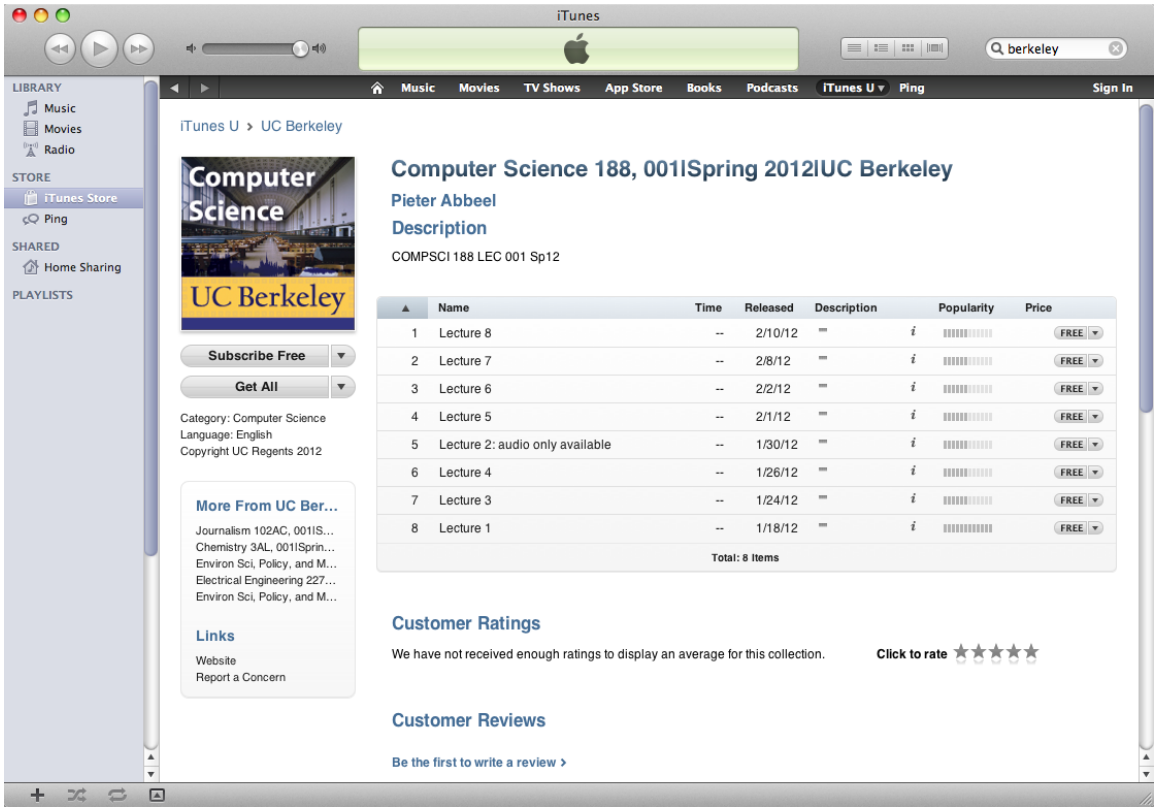



Figure 1.4: Screenshot of the Apple iTunesU application featuring the contents provided by the University of California at Berkeley (retrieved Feb. 14, 2012).

**Apple iTunes U** – `http://www.apple.com/education/itunes-u` Similar to Matterhorn, *Apple iTunes U* is more of a distribution channel than an actual lecture interface. The main difference is that iTunes U is a commercial service that only distributes consumer-ready videos, as shown in Fig. 1.4. The authoring institution, typically a university, can add audio and video lectures, and basically acts as an artist or music label. In iTunes, recordings can be grouped, annotated with descriptions, rated and reviewed by listeners/viewers. Furthermore, the recordings may be free or for purchase using the iTunes payment system.

Apple iTunes U is rather simplistic. The educational content is distributed and displayed as a plain audio or video recording, ignoring supplemental material such as presentation slides or related literature. Furthermore, it is the only non web-based platform presented in this chapter which results in technical limitations: The recordings can only be downloaded and viewed using the Apple iTunes program which is only available for Apple MacOS and Microsoft Windows whereas incompatible operating systems or devices are locked out.

Listing the lectures as "regular" iTunes content also leaves little hope for a future enhanced interface that allows the integration of additional educational material. Although iTunes U was among the first services to host video lectures, it has not evolved since. The rather small number of participating universities suggests that the setup is too rigid to be used for e-learning.

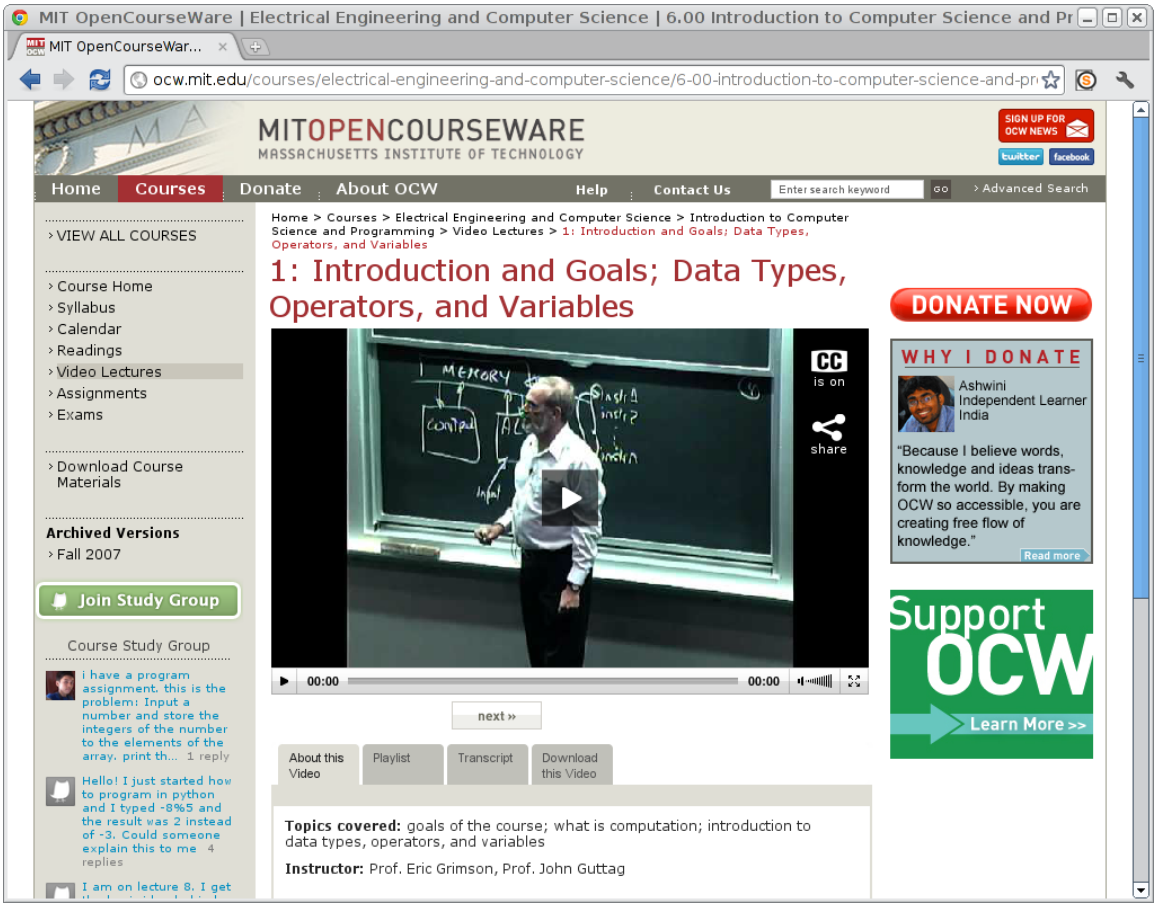



Figure 1.5: Screenshot of the MIT OpenCourseWare video lecture interface (retrieved Feb. 14, 2012).

**MIT OpenCourseWare** – `http://ocw.mit.edu` The *Massachusetts Institute of Technology (MIT) OpenCourseWare (OCW)* is an e-learning platform with extensive possibilities where video lectures are a key feature. Departments and individual professors design courses and add syllabus, video lectures, lecture notes, assignments and further material as they please. The videos are manually annotated with supplemental information such as name of the instructor, topics covered, summaries or key words. Transcripts of the lecture are partially available. Some courses offered through the OCW platform also link to respective groups at OpenStudy[5], a community to connect to fellow students, ask questions and help fellows. Depending on the class, all or parts of the material is only available to students enrolled at MIT.

The video display shown in Fig. 1.5 has no additional features. On top of the OCW video archive, the MIT Computer Science and Artificial Intelligence Laboratory (CSAIL) developed an interactive lecture browser that uses automatic speech processing to obtain automatic transcripts. In addition to viewing the video, the user can scroll through these transcripts and search for phrase occurrences in the lecture. Unfortunately, the CSAIL lecture browser is not yet part of OCW.

[5] `http://www.openstudy.com`

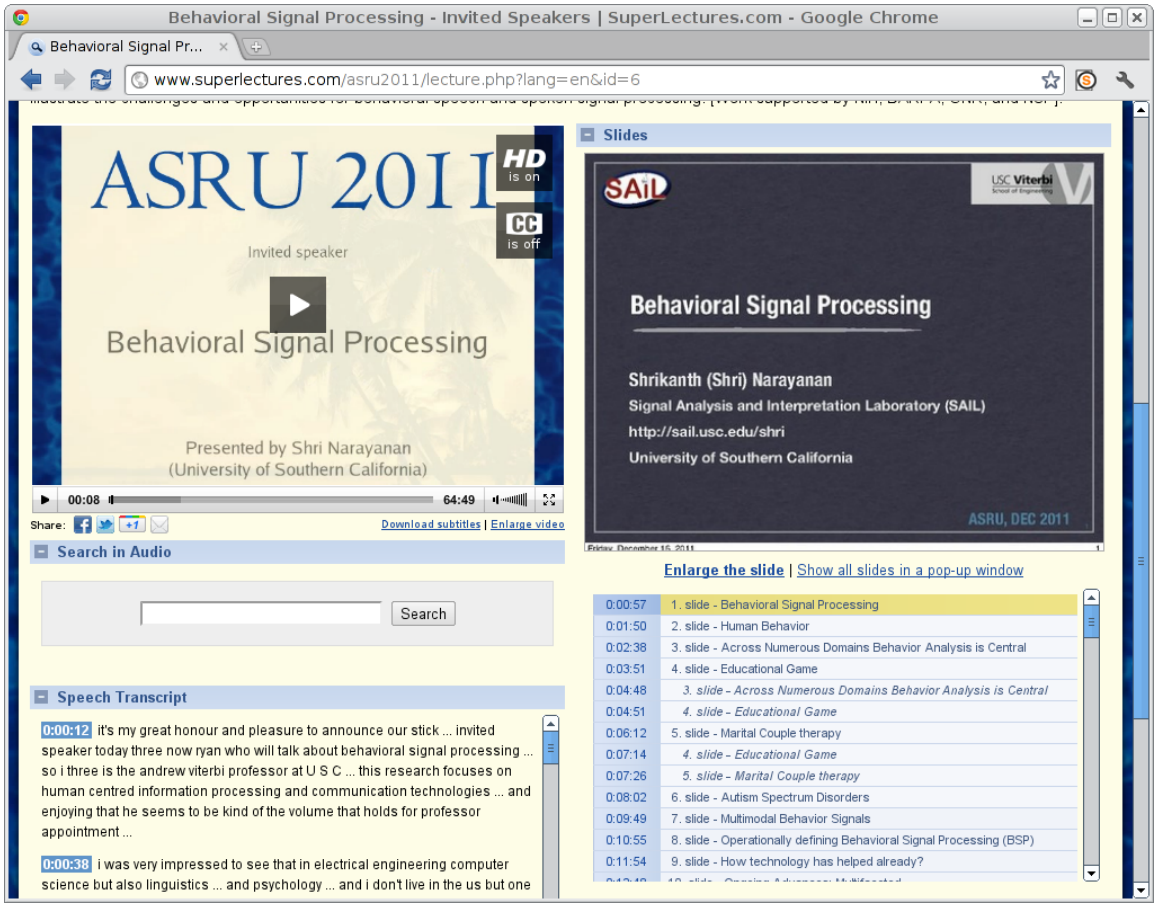



Figure 1.6: Screenshot of the `superlectures` video lecture interface (retrieved Feb. 14, 2012).

**superlectures** – `http://www.superlectures.com` The Czech start-up *superlectures* offers a full-featured video lecture solution from acquisition to storage and distribution. Their key component is a lecture browser which is displayed in Fig. 1.6. The video (top left) is accompanied by the currently displayed slide (top right). The automatic speech transcripts (with time marks. Bottom left) can be scrolled and searched, the list of slides (bottom right) contains time marks and headings. Both transcripts and slide list can be used to navigate within the video.

Although superlectures is affiliated with the Brno University of Technology, Czech Republic, their focus is not necessarily "traditional" teaching or e-learning but any type of presentation, ideally with the slides available. The web-based interface has an appealing design and seamlessly integrates video, transcript and presentation slides. For larger events, the user can view use statistics, join discussion groups and watch related videos.

A very similar interface, but without automatic transcription, is marketed by the Slovenian company Viidea[6], with a showcase at `http://videolectures.net`.

[6] `http://www.viidea.com/`

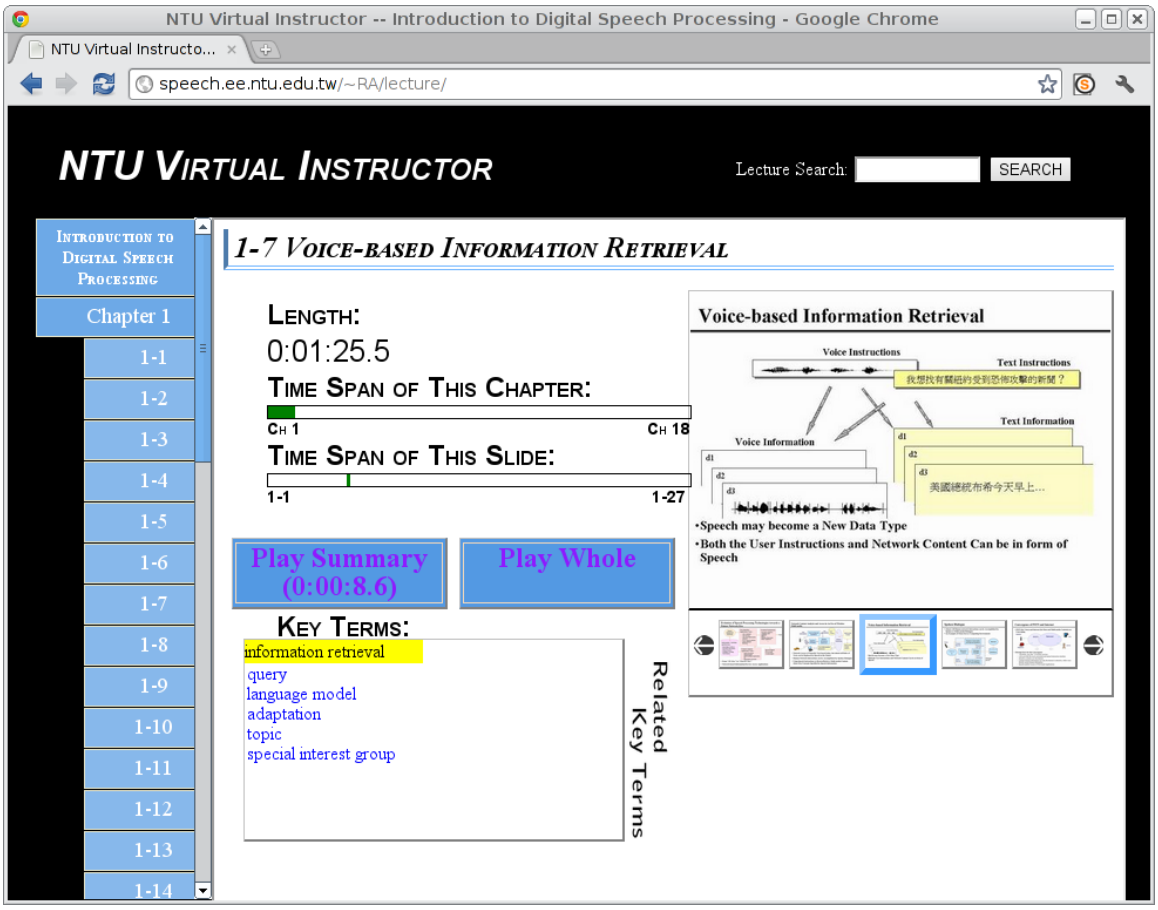



Figure 1.7: Screenshot of the NTU Virtual Instructor video lecture interface (retrieved Feb. 14, 2012).

**NTU Virtual Instructor** – `http://speech.ee.ntu.edu.tw/~RA/lecture` The National Taiwan University (NTU) *Virtual Instructor* is a case study based on the lecture series "Introduction to Digital Speech Processing" recorded at the NTU. The user interface depicted in Fig. 1.7 is designed to be a "virtual instructor:" Each chapter and its sections are associated with a video and presentation slides. Two horizontal bars indicate how much time a chapter and the currently selected slide cover in the recording. Similar to other platforms, the slides can be used to navigate within the video and a search function can be used to find occurrences of key words based on an automatic transcript. Recently, the NTU Virtual Instructor has been extended by two interesting modules. For each chapter, the user can view an audio-visual summary which consists of short extracts of the video. Furthermore, automatically extracted key terms give an idea about the contents.

In terms of features, the NTU Virtual Instructor is the most advanced video lecture platform compared to the previously presented ones. The design of the interface and user interactions are however far behind compared to for example the interface of superlectures. Another drawback is that the Virtual Instructor is built on top of a single lecture series and is thus a very specific solution exploiting structure elements such as chapters or a specific slide format.

In summary, the existing systems show a broad variety of features reaching from simple video catalogs such as iTunes U to integrated e-learning platforms as for example the OCW. However, only a few *automatically* distill further information from the data to provide features such as text search or key terms to add value to the videos. Furthermore, the great variance in optical appearance and possible interactions suggests that the developers are rarely aware that these are essential factors for usability and user acceptance. The ideal system is easy and intuitive to use, has an ergonomic and appealing interface design, and makes the most out of the available data, as for example a search function or automatic key terms and summaries.

## 1.3 Aim of this Thesis and Scientific Contributions

The aim of this thesis is to develop such an intuitive interface that allows the user to quickly extract all the important information from a video lecture — without the need of watching the whole recording. While watching the full video might be appropriate to a distant learner or someone that follows the lecture for the first time, it might be a painful loss of time to some other student. For example, consider a student that watches a video lecture to prepare for an exam. Typical questions might be: What was that lecture about in general, *i.e.*, was there anything of interest? When did the instructor talk about a specific topic? What issues are related to a certain topic? The student has two options: either watch the whole video or "skim" the video by sampling the topics at more or less random positions. The first option may be tedious – lectures are typically 50 to 90 minutes – and end in frustration if the video contains nothing of interest to the viewer. The second option saves time, of course, but the user is at risk missing important details due to the sampling. For textbooks, summaries (cover blurbs), table of contents and indices are taken for granted, so why not also have these for video lectures?

At present, the NTU Virtual Instructor is the only platform that distills additional information from the video beside a video-slide-transition synchronization and a searchable automatic transcript, but comes at the expense of an interface that is far behind in terms of the optical and technical design, for example compared to the superlectures platform. Modern web technologies such as HTML5, CSS and JavaScript allow to build web sites that appear as conventional applications that can be controlled using common user interactions such as keyboard shortcuts, mouse clicks and drag-and-drop — the so-called *web 2.0* closes the gap between web services and standalone applications.

This thesis describes the progress from the rather basic FAU Videoportal to a novel interactive video lecture browser using automatic means for speech recognition, key phrase extraction, summarization and visualization. Beside the technical implementation, the following scientific contributions are made:

- Based on two courses recorded in 2009, a multi-modal corpus of academic spoken English comprising audio, video and presentation slides is compiled. Using Blitzscribe2, a new tool for the rapid transcription of speech implemented

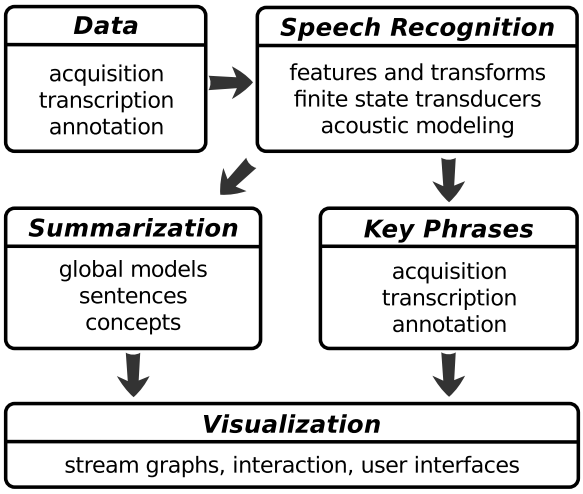



Figure 1.8: Overview and relation of the topics covered in this thesis.

on top of the JSTK [Stei 11], the manual transcriptions are generated in less-than-average time.

- The KALDI speech recognition toolkit is extended by two types of semi-continuous acoustic models along with maximum likelihood training algorithms. The implemented models feature a lower number of parameters than the conventional models which is advantageous if only limited training data is available [Ried 12].

- An unsupervised key phrase extraction and ranking is developed that can be used for automatic summarization and visualization of spoken document contents such as video lectures or meetings. A novel summarization model is described that significantly outperforms previously published systems [Ried 08b, Tur 08, Ried 08a, Gill 09, Tur 10, Ried 10].

- A novel interactive summarization tool is developed that allows the user to assess the spoken document by iteratively generating summaries based on key phrases and respective weights set by the user [Ried 08a].

- A novel lecture browser is developed that integrates the video with a key phrase annotation and visualization panel. The latter shows which key phrases occur at what time in the video, and how dominant they are compared to the others. The interface can be used for e-learning and data collection for future tasks [Grop 11, Ried 11].

## 1.4 Thesis Outline

The outline of this thesis follows the five major steps of the processing pipeline as depicted in Fig. 1.8. The audio track of the recorded video is fed to an automatic speech recognition (ASR). The resulting transcripts are the input to the key phrase

extraction and summarization modules. In a last step, the key phrases and summaries are visualized in interactive tools.

Chapter 2 describes the speech corpora used for the presented experiments. The focus is on the acquisition and annotation of a new corpus of spoken academic English which will be used throughout all experiments. Further corpora include the Wall Street Journal and ICSI Meeting Corpus which are used to compare the performance of the speech recognition and summarization systems to related work.

Chapter 3 introduces to the open source automatic speech recognition (ASR) toolkit KALDI based on weighted finite state transducers. KALDI consists of three major components. The acoustic front-end extracts features from the speech signal by splitting the continuous analog signal in discrete frames and applying spectral analysis. The actual recognition process is based on a combination of an acoustic model (AM) that represents the (English) phonemes, and a language model (LM) that describes the pronunciation of words and the possible word sequences. Special emphasis is put on the comparison of different AM topologies regarding their recognition performance and number of parameters.

Starting from the automatic or manual speech transcripts, Chapter 4 describes the automatic extraction of salient phrases, the *key phrases*. The typically numerous candidate phrases are ranked in terms of their salience using different unsupervised methods, *i.e.*, algorithms which do not depend on additional prior knowledge. The automatically generated key phrase rankings are compared to human ones to show the validity of the methods.

Chapter 5 introduces to speech summarization and describes how to automatically generate summaries of spoken documents such as lectures or meetings by extracting salient utterances. This salience is determined based on present key phrases and a notion of their salience. Special attention is paid to the comparison of models that work on the basis of utterances in contrast to a more fine-grained unit such as words or phrases.

Finally, Chapter 6 describes interactive interfaces for user-guided summarization and for the lecture browser based on the visualization of key phrases. An intuitive design of the graphical user interfaces allows the user to quickly assess the video lecture and find the sections of interest which is confirmed in two pilot user studies.

Chapter 7 and 8 conclude this thesis with an outlook on possible future work and a summary. Supplemental material is attached in the Appendix.

# Chapter 2

# Data

## 2.1 The LME Corpus of Academic Spoken English

The LME corpus of academic spoken English (LMELectures) was recorded, transcribed and annotated at the *Lehrstuhl für Informatik 5 (Mustererkennung), Univ. Erlangen-Nürnberg*. It covers two separate graduate level computer science classes read during the summer term 2009 by Prof. Dr.-Ing. Joachim Hornegger, the head of the Pattern Recognition Lab. The first lecture series, Interventional Medical Image Processing *(IMIP)*, consists of 18 lectures covering topics such as magnetic resonance tomography, x-ray computed tomography and ultrasound imaging. The second series, Pattern Analysis *(PA)*, consists of 18 lectures covering topics in machine learning such as classification, regression and optimization. All lectures were recorded in the *E-Studio* at the *Regionales Rechenzentrum Erlangen (RRZE)*[1]. The captured high-definition (HD) audio and video was professionally edited to achieve a constant high recording quality.

### 2.1.1 Audio and Video Data

The audio data was acquired at a sampling rate of 48 kHz and 16 bit quantization and stored in the *Audio Interchange File Format (AIFF)*. A 16 kHz version for the use with the speech recognizer was produced using down-sampling. A cordless close-talking microphone was used to eliminate most of the room acoustics and background noise.

The video was acquired using an HD camera with manually controlled viewpoint and zoom. Furthermore, the currently displayed slide (and on-screen writings, if applicable) was captured. The final video canvas is divided into three parts, showing the lecturer on the top left, a static display of the lecture title and the date on the bottom left, and the currently projected slide on the right (*cf.* Fig. 2.1). Although the presentation slides are provided, only the video provides accurate information whether slides were actually shown, and if the lecturer added comments using on-screen writing. The video data is provided in Apple Quicktime format.

In total, 39 hours of audio and video data was acquired; a detailed list of the recordings can be found in the appendix A.1. The experiments in this thesis focus on

[1] `http://www.rrze.uni-erlangen.de`

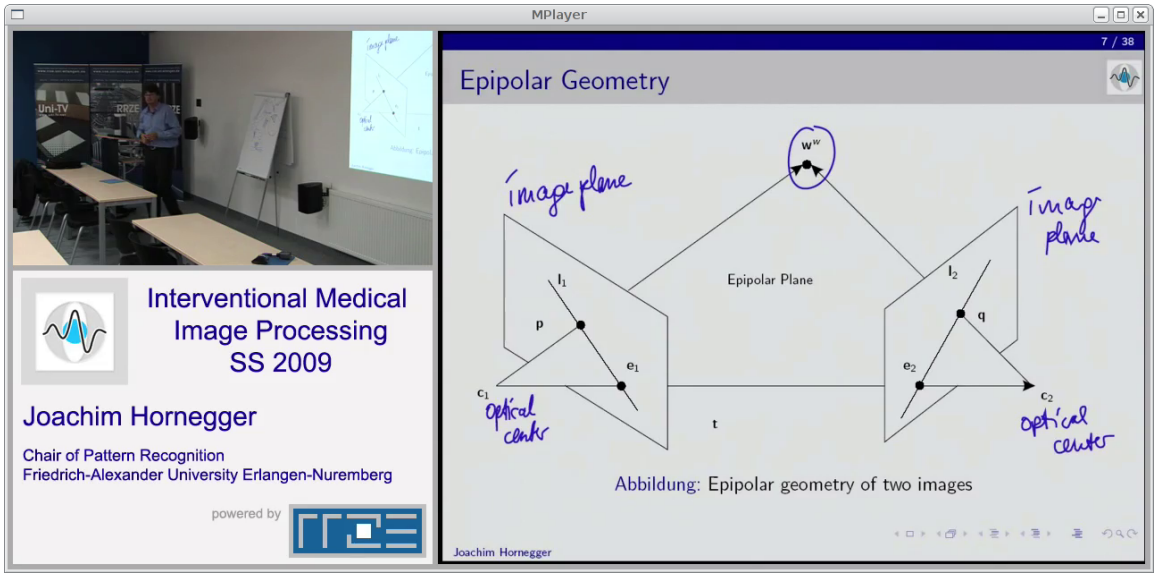



Figure 2.1: Example image from the video of lecture *IMIP01*. The left side shows the lecturer (top) and the lecture title (bottom), the right side shows the current slide and on-screen writings.

speech recognition and understanding and are thus solely based on the audio data. For the interactive part, only that part of the video canvas is shown that displays the presenter for two reasons. First, the presented methods do not make use of the information on the slides, and second, to show the suitability of the interface for the rather common situation where only the video but not necessarily the presentation slides are displayed or available.

### 2.1.2 Semi-Automatic Segmentation

For the manual transcription, as well as for the training and the evaluation of a speech recognition system, long recordings are split into short segments of speech (typically a few seconds in duration). As a by-product, silence between segments is removed. The segmentation is based on the time alignments of a Hungarian phoneme recognizer [Mate 05b] which has been successfully used for speech/non-speech detection in various speaker and language identification tasks, *e.g.*, [Mate 05a]. The rich phonetic alphabet of the Hungarian language was found to be advantageous in the presence of various languages (here German and English) or wrong pronunciations. The phoneme strings were simplified by mapping the 61 symbols to two groups: the pause (*pau*), noise (*int*, *e.g.*, a door slam) and speaker noise (*spk*, only if following *pau*, *e.g.*, cough) symbols were mapped to *silence* and the remaining symbols to *speech*. Merging adjacent segments of *silence* and *speech* results in an initial speech/non-speech segmentation (*cf.* Fig. 2.2).

Due to the nature of the phoneme recognizer, the initial segmentation is very strict and does not necessarily reflect the actual utterance or sentence structure, as even a very short pause terminates a speech segment. With the aim of producing speech

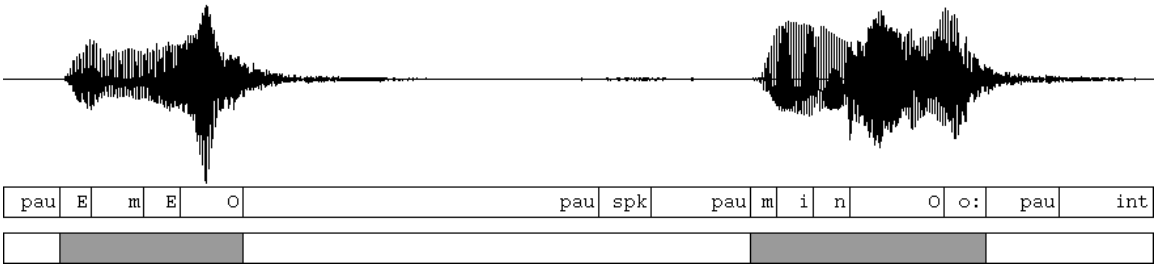



Figure 2.2: *»And then (breath) we know«.* Adjacent segments of *silence* or *speech* phonemes are merged to an initial speech (gray) and non-speech (white) segmentation.

segments of an average length of four to five seconds[2], consecutive speech segments are merged based on certain criteria regarding segment lengths and intermediate silence (*cf.* Tab. 2.1). A schematic description of the merging procedure is given in Algorithm 1.

**Algorithm 1:** Merge of consecutive segments based on their duration and interleaving silence.

```
for all segments i do
    if Pause(i, i+1) < min.pau or Duration(i) < min.dur then
        required ← true
        while required or Duration(i) < max.dur do
            if !required then
                if Duration(i) > med.dur then break
                if Duration(Merge(i, i+1)) > max.dur then break
                if Pause(i, i+1) > max.pau then break
            end
            i ← Merge(i, i+1)
            required ← (Pause(i, i+1) < min.pau)
        end
    end
end
```

To account for sharp cutoffs, 150 milliseconds were added to the end of each segment. Given the target length, the main control variables are the pauses. Allowing too long pauses within a segment (`max.pau`) might lead to segments that contain the end and beginning of two separate utterances. Requiring long silences between segments (`min.pau`) leads to unnaturally long segments. The effect of these numbers on the overall number of speech segments and their duration is given in Figs. 2.3 and 2.4.

The segmentation closest to the desired characteristics comprises 23 857 speech turns with an average duration of 4.4 seconds, and a total of about 29 hours of speech.

[2]as suggested by previous experiences of the group with manual transcription and speech recognition system training and evaluation

| *quantity* | *description* | *value* |
|---|---|---|
| `min. dur` | if segment is shorter than *min. dur*, merge with following | 2 s |
| `med. dur` | stop if merged segment is longer than *med. dur* | 4 s |
| `max. dur` | only merge if resulting segment is shorter than *max. dur* | 6 s |
| `max. pau` | maximum duration of pause within a segment | 1 s |
| `min. pau` | minimum duration of pause between two segments | 0.5 s |

Table 2.1: Final merging criteria for consecutive speech segments. No merge if the resulting segment would be longer than 30 seconds.

The right column of Tab. 2.1 shows the chosen merging criteria. The typically 0.5 s to 3 s of silence between speech segments accumulates to about 10 hours.

### 2.1.3 Manual Transcription using BLITZSCRIBE2

The manual transcription of speech data typically requires about ten to 50 times the duration of speech using professional tools like TRANSCRIBER [Barr 01, Roy 09]. TRANSCRIBER, like other tools, allows to work on long recordings by identifying segments of speech, noise and other acoustic events. Furthermore, higher level information like speaker, speech or language attributes can be annotated. However, this higher level information of the data at hand is mainly known in advance, and lectures are, as the name suggests, typically very dense in terms of speech, reducing the main task to the desirably fast transcription of the speech segments.

#### BLITZSCRIBE2: a Tool for Rapid Speech Transcription

The speech segments were manually transcribed using BLITZSCRIBE2[3], a platform independent graphical user interface specifically designed for the rapid transcription of large amounts of speech data. BLITZSCRIBE2 was implemented in the scope of this thesis and is inspired by research of Roy *et al.* [Roy 09]. It is publicly available as part of the JSTK. The interface (*cf.* Fig. 2.5) displays a waveform of the currently selected speech segment, a progress bar indicating the current playback position, an input text field for the corresponding transcription, and a list of turns with or without prior transcription.

The key idea to speed up the transcription is to simplify the way the user interacts with the program: although the mouse may be used to select certain turns for transcription or replay the audio at a desired time, the most frequent commands are accessed via keyboard shortcuts listed in Tab. 2.2. For a typical segment, the transcriber types the transcription as he listens to the audio, pauses the playback if necessary (CTRL+SPACE), and hits ENTER to save the transcription, load the next segment and start the playback. This process is very ergonomic as the hands (may) remain on the keyboard during all times.

BLITZSCRIBE2 writes a journal of user interactions to a temporary file which can be used to trace the transcription process and reconstruct transcription files at certain

[3] `http://www5.informatik.uni-erlangen.de/en/research/software/blitzscribe2/`

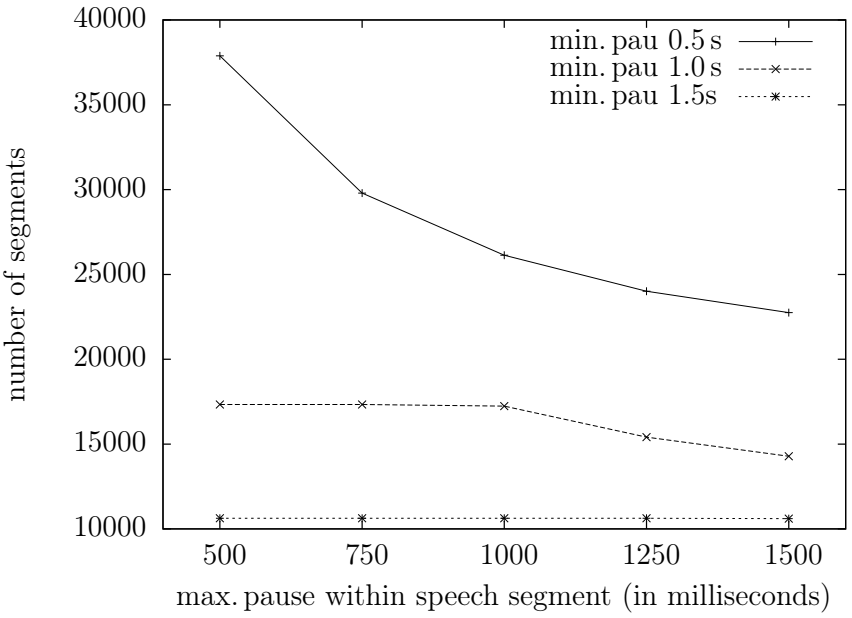



Figure 2.3: Effect of `max. pau` and `min. pau` on the overall number of speech segments. A shorter `min. pau` generally results in a larger number of shorter segments; allowing longer pauses within segments results in a smaller number of longer segments. Note that due to the enforced minimum duration, segments may contain pauses of a length that would otherwise lead to a split.

times. Although the functionality of BLITZSCRIBE2 is very limited compared to for example TRANSCRIBER, the very ergonomic interface is expected to significantly speed up simple transcription tasks.

#### Results of the Transcription Process

The lectures were transcribed by two transcribers. The work was shared among the transcribers and no lecture was transcribed twice. As the language is very technical, a list of common abbreviations and technical terms was provided along with the annotation guidelines (*cf.* App. A.2). The overall median time factor was about five times slower real time, but decreased over time with the transcriber adapting to the transcription tool, speaker and topic (*cf.* Fig. 2.6).

In total, about 300 500 words were transcribed with an average of 14 words per speech segment. The resulting vocabulary size is 5 383 including multiple forms of words (*e.g.*, plural, composita), but excluding words in foreign languages and mispronounced or word fragments.

### 2.1.4 Further Manual Annotations

For both lecture series, the presentation slides are available in machine readable (PDF) form. Note that these exclude the on-screen writings during the class. As some slide sets were used on multiple days, only the video provides accurate tim-

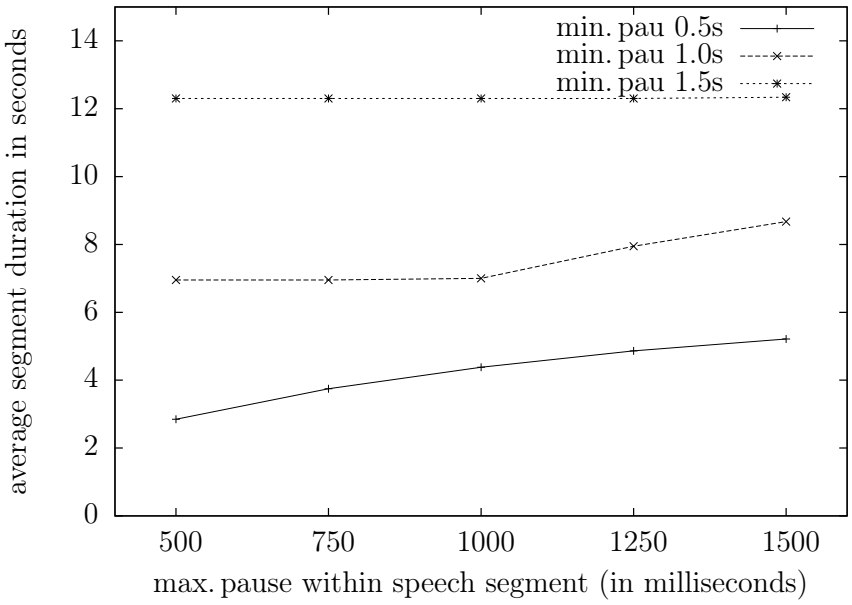



Figure 2.4: Effect of `max. pau` and `min. pau` on the average duration of the speech segments in seconds.

| *key combination* | *command* |
|---|---|
| ENTER | save transcription, load **and play** next segment |
| SHIFT + BACKSPACE | save transcription, load previous segment |
| SHIFT + ENTER | save transcription, load next segment |
| CTRL + SPACE | start/pause/resume/restart playback |
| CTRL + BACKSPACE | rewind audio and restart playback |
| ALT + S | save transcription file |

Table 2.2: Keyboard shortcuts for fast user interactions in BLITZSCRIBE2.

ing information of the individual slides. For the *PA* series, the lecturer assigned a small number of key phrases to each lecture to stress the covered topics. These are considered as a ground truth and will be referenced as the "lecturer's phrases" later on.

In the context of a student thesis, the individual lecture *PA06* was evaluated by five human subjects [Grop 10]. The German graduate students of computer science either observed the lecture, watched the video recording, or attended the same class in a different term. A list of 50 possible key phrases was selected from the transcripts according to the syllabus. From this preselection, the annotators identified and ranked up to 20 key phrases by placing them in a certain order and grading them in terms of quality from 1 – "sehr relevant" (*very relevant*) to 6 – "nutzlos" (*useless*) using a questionnaire (*cf.* App. A.3).

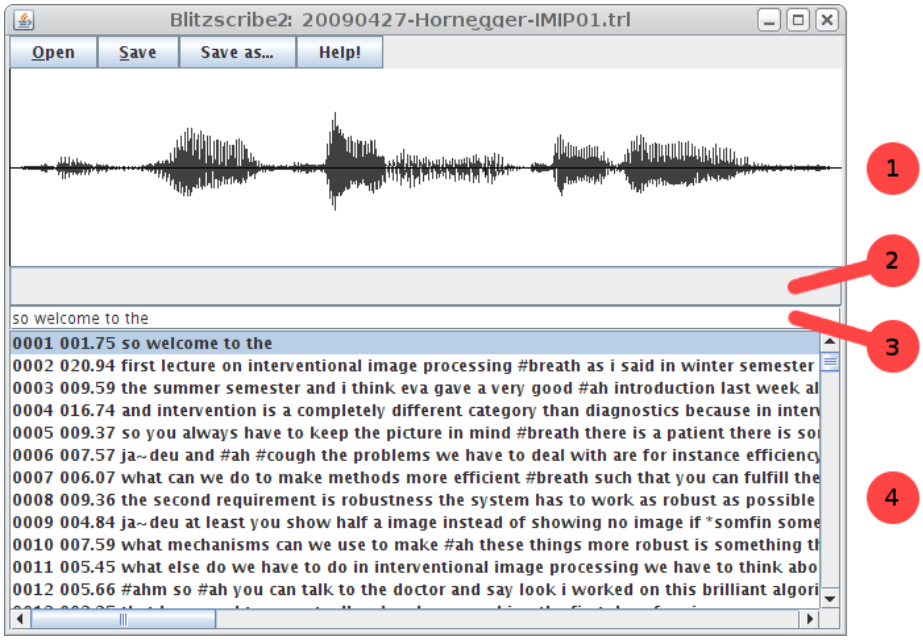



Figure 2.5: Screenshot of the BLITZSCRIBE2 transcription tool; (1) waveform of the currently selected speech segment, (2) progress bar indicating the current playback position, (3) text field for the transcription, (4) list of segments with transcription (if available).

### 2.1.5 Intended Use and Distinction from Other Corpora of Academic Spoken English

The corpus, with its annotations, is an excellent resource for various mono- and multi-modal research. The roughly 30 hours of speech of a single speaker provide a great base to work on acoustic and language modeling, speaker adaptation, prosodic analysis and key phrase extraction. The style of the language is somewhere in between read and spontaneous speech: while the speech is mainly spontaneous (along with disfluencies and hesitations), the conveyed information can usually be found on the currently displayed slide. This opens the possibility of integrating information extracted from the video using optical character or hand writing recognition to the speech recognition and key phrase extraction process. At a higher level, the timing information and content of the presented slides can provide natural labels for topic segmentation and classification.

The two main corpora of academic spoken English are the BASE corpus[4], and the Michigan Corpus of Academic Spoken English (MICASE) [Simp 02]. Although both corpora cover more than 150h of speech, their setting is different from the LMELectures. The BASE corpus covers 160 lectures and 40 seminars from four broad disciplinary groups (Arts and Humanities, Life and Medical Sciences, Physical Sciences, Social Sciences). Audio, video and transcription material are subject to different licensing options. The MICASE corpus features a wide variety of recordings

[4]The British Academic Spoken English (BASE) corpus project. Developed at the Universities of Warwick and Reading under the directorship of Hilary Nesi and Paul Thompson.

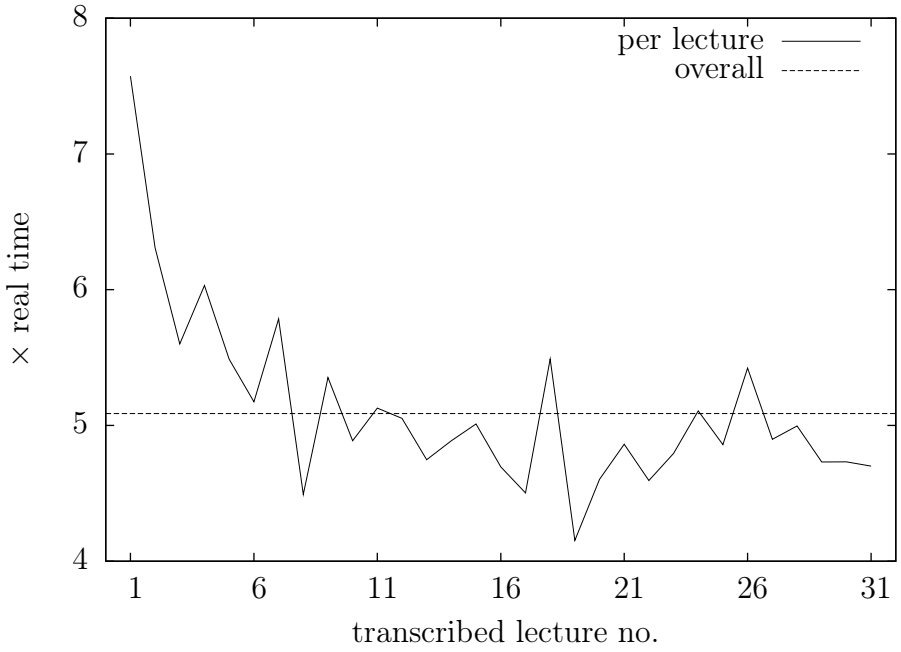



Figure 2.6: Change of the median transcription real time factor required by transcriber 1 throughout the transcription process.

of academic events including lectures, colloquia, meetings, dissertation defenses, *etc.*. Audio and transcripts are subject to licensing, but video data is unavailable.

The main distinction of the LMELectures corpus is its consistency in terms of a single speaker, fixed recording environment and two defined lecture series. Furthermore, the available presentation slides and the on-screen writings in the video make it a unique resource for multi-modal approaches.

### 2.1.6 Data Partitioning

For the experiments in this thesis, the data is partitioned in three parts. The development set, *devel*, consists of the four lecture sessions IMIP13, IMIP17, PA15 and PA17, and has a total duration of about two hours. The test set, *test*, consists of the four lecture sessions IMIP05, IMIP09, PA06 and PA08, and has also a total duration of about two hours. The remaining 28 lecture sessions form the training set, *train*, with a total of about 24 hours. Tab. 2.3 summarizes the partitioning and lists details on the duration, number of segments and words, and out-of-vocabulary (OOV) rate with respect to a lexicon based on the training set.

## 2.2 Wall Street Journal

Strictly speaking, Wall Street Journal (WSJ) [Garo 07] is not a corpus, but often used as a synonym for the two corpora CSR-I (WSJ0) and CSR-II (WSJ1), both collected and distributed by the Linguistic Data Consortium (LDC). The corpora

| *name* | *duration* | *# turns* | *# words* | *% OOV* |
|---|---|---|---|---|
| train | 24h 31m 55s | 20 214 | 250 536 | — |
| dev | 2h 07m 28s | 1 802 | 21 909 | 0.87 % |
| test | 2h 12m 30s | 1 750 | 23 497 | 0.99 % |

Table 2.3: Data partitioning for the LMELectures corpus; the number of words excludes word fragments and foreign words. The percentage of OOV words is given with respect to the words present in the *train* partition.

contain recordings of read speech recorded with a close-talking Sennheiser HMD 414 and a varying secondary microphone. Professional male and female speakers read 5 000 to 20 000 word texts from the Wall Street Journal news paper. WSJ0 consists of a longitudinal speaker dependent set (LSD), long (SI12) and short term speaker independent (SI84) training sets, as well as two "hub and spokes" test sets. Similarly, WSJ1 provides long (SI25) and short term speaker independent (SI200) training sets as well as various test data. For the experiments in this thesis, the following sets are used:

- *SI84* training set – 7 240 utterances of 84 (42 female) speakers, about 15 hours of speech.
- *SI200* training set – 29 320 utterances of 200 (100 female) speakers, about 45 hours of speech.
- *SI284* training set – Combination of SI84 and SI200.
- *Dev'93* development set – 503 utterances of WSJ1.
- *Nov'92* test set – 333 utterances of WSJ0.
- *Nov'93* test set – 213 utterances of WSJ1.

The above sets have been widely used in related work and provide a solid base to train and evaluate the acoustic models.

## 2.3 The ICSI Meeting Corpus

The ICSI Meeting Corpus (ICSIMC) [Jani 03] is a well-studied corpus of 75 regularly scheduled group meetings at the *International Computer Science Institute*, *ICSI*, at Berkeley, each lasting about 45 minutes. Following related work, a subset of 57 meetings is used. These have been transcribed and annotated with dialog acts as well as abstractive and extractive summaries (about 500 words on average). Six meetings, namely *Bed{004,009,016}*, *Bmr{005,019}* and *Bro018* will be used as a test set, for which three human abstracts are available. Automatic speech recognition transcripts are provided by SRI International using their conversational telephone speech recognition system. The achieved word error rate is about 37% [Zhu 05].[5]

[5]The experiments on the ICSIMC data set were performed while the author was with *ICSI*.

The extensive manual annotations, as well as the numerous related work on summarization, make the ICSIMC the first choice to show the performance of the key phrase extraction and summarization algorithms described in this thesis. Furthermore, the naturalness and similar academic content of the speech make it similar to the LMELectures corpus.

# Chapter 3

# Automatic Speech Recognition

"How to wreck a nice beach."

*by Dave Tompkins*
*ISBN 978-1-93-363388-6*

## 3.1 Introduction

The roots of speech recognition date back to the 1920s, long before there were computers. An electric toy, "Radio Rex," was probably the first commercially available speech recognition application. Rex was a little dog that would come out of his kennel when called, designed and patented by the Austrian inventor Christian Berger [Berg 13, Berg 16]. The mechanism behind it is a spring that is triggered by a 500 Hz acoustic signal, which is for example found in the `/eh/` sound in "Rex" [Davi 62]. Fig. 3.1 shows an excerpt of the original patent application including the circuit and toy design.

Berger's principal idea of a frequency based actuator was quickly adopted and extended by others. In 1939, Bell Labs introduced the "Vocoder" [Dudl 39a, Dudl 39b], a machine to encode, transmit, and decode a speech signal based on its frequency components. Later in 1952, with the advent of computers, the first actual automatic speech recognition system "Audrey" [Davi 52], that was already able to recognize the first ten digits with an astonishing accuracy of 97 to 99% — given the speaker was male, known to the system and spoke with at least 350 ms delay between the digits. The basic idea behind Audrey, the recognition of phonemes based on their frequency representation, is still fundamental in current state-of-the-art systems. The probably most general patent on speech technology was filed on Feb 1, 1965 by E. Newman (U.S. pat. 3,400,216) [Newm 68], claiming a "Speech Recognition Apparatus" that translated input waveform signals into visible output symbols (diodes) based on autocorrelation.

Until the late 1970s, most speech recognition systems would rely on example (or *template*) based approaches, such as the direct comparison of a recorded word with a set of possible reference recordings using dynamic time warping [Sako 78]. It was Jim Baker and Frederick Jelinek at IBM Research to revolutionize the field by

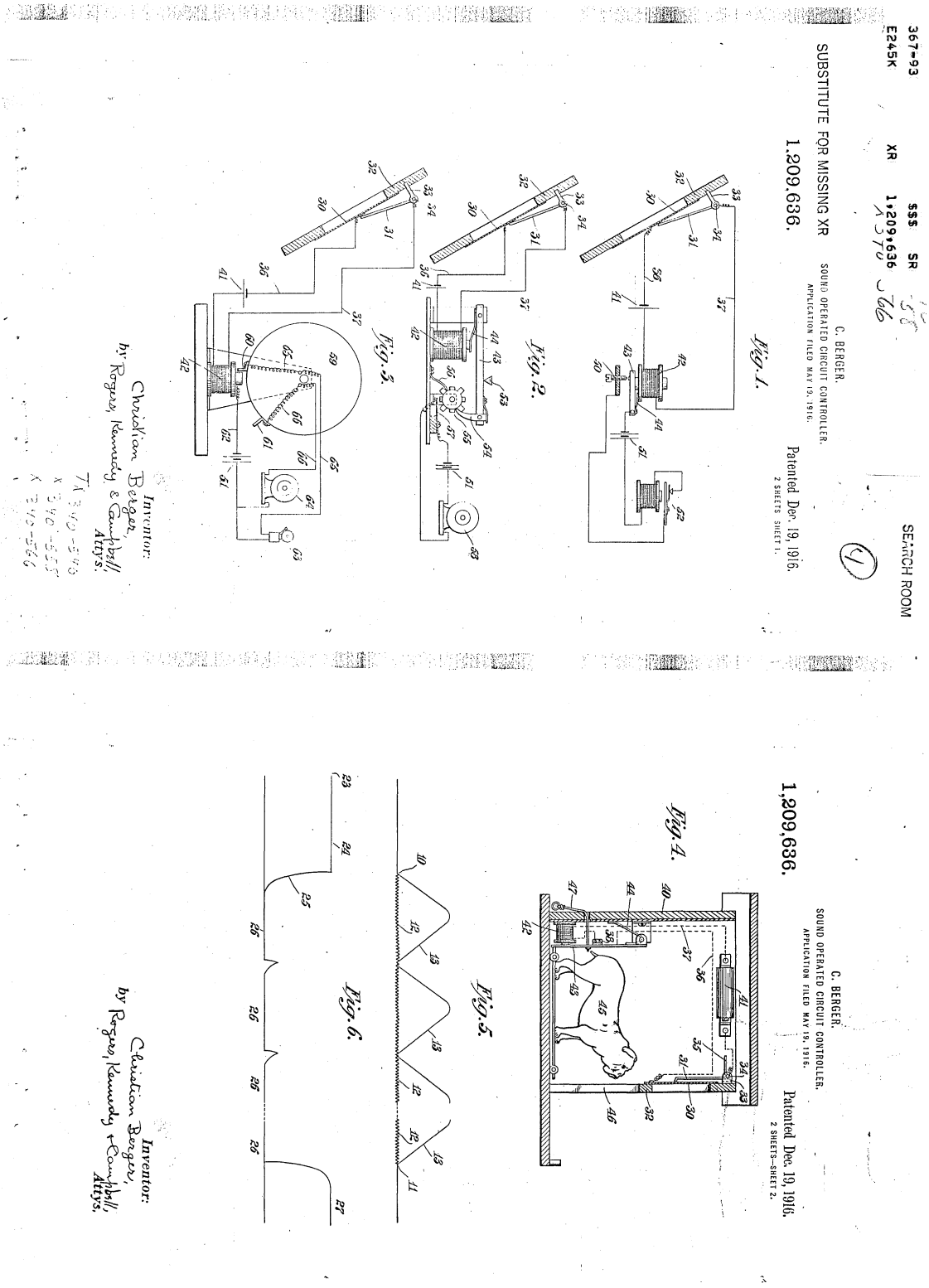


Figure 3.1: Extract of Christian Berger's U.S. pat. 1,209,636 showing the circuit and design of the "Radio Rex" voice activated toy; `http://www.google.com/patents/US1209636`

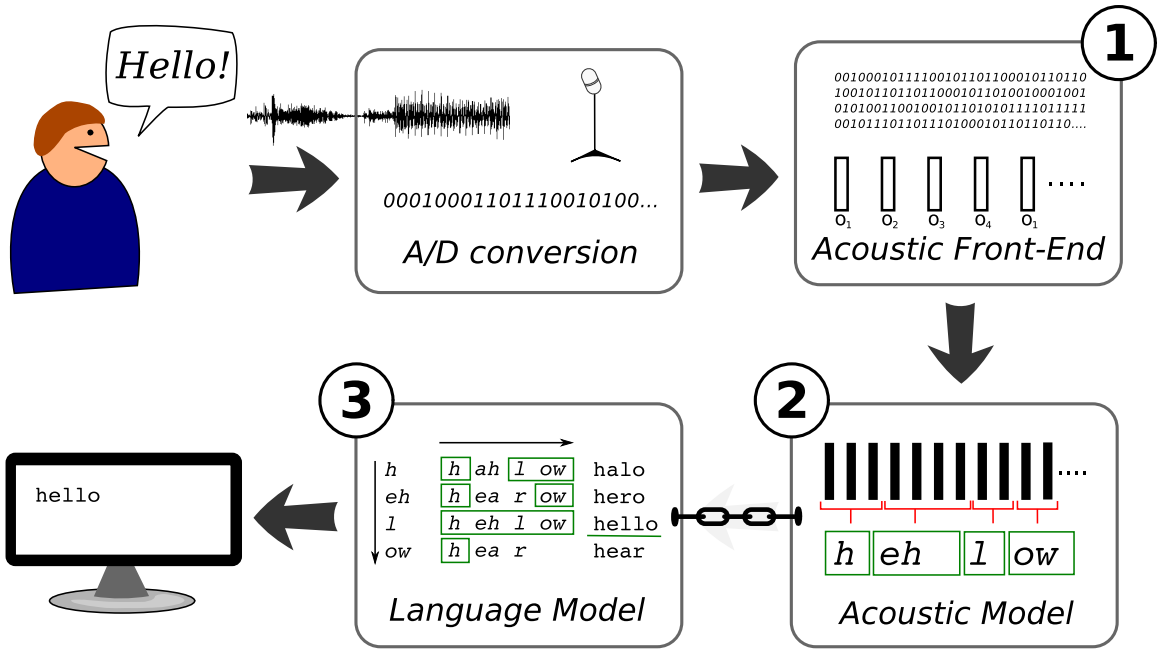



Figure 3.2: Overview of a typical speech recognition system. The analog audio signal is acquired using a microphone and converted to a digital signal. The short-time analysis (1) extracts a series of observations $(o_1, o_2, \dots)$ from the continuous signal. The acoustical model (AM, 2) and language model (LM, 3) are closely linked: The AM contains the statistic parameters that are necessary to map the observations to phonemes. On top of it, the LM defines the possible phoneme and word sequences.

establishing statistical methods for speech recognition, most successfully the hidden Markov model (HMM) [Bake 75, Jeli 76].

The general architecture of a typical automatic speech recognition (ASR) system is divided in three major components, as shown in Fig. 3.2. At the *acoustic front-end (AFE)*, the system extracts a series of observations (features) from the analog/digital converted signal based on short-time frequency analysis. The decoding process depends on an *acoustic model (AM)* and *language model (LM)* that are closely linked. Simply speaking, the AM maps observations to phonemes, while the LM determines which phoneme sequences (words) are possible, and which word sequence (sentence) is most likely.

The following sections detail the theory and algorithms of these three main components, and introduce to hidden Markov models (HMM) and their implementation using weighted finite state transducers (WFST). Special attention is paid to the AM. Different styles of models are compared with respect to their number of parameters and recognition performance. Parts of this chapter have been published in [Pove 12, Ried 12].

## 3.2 Acoustic Front-End

The first step is to extract features from the input speech signal, that is to compute a time ordered series of real valued observations $\mathcal{O} = \{o_1, \ldots, o_T\}$, $o_t \in \mathbb{R}^d$, where $d$ is typically in between 24 and 40 depending on the feature extraction procedure.

### 3.2.1 Preprocessing

It is common practice to do some preprocessing prior to the actual feature extraction. Most popular are the DC-shift removal, pre-emphasis, and dithering. The DC-shift is a constant offset in the audio recording due to an extra current in the microphone membrane. It is as easy to recognize as to remove: the mean of all values is computed and subtracted from each sample:

$$\tilde{f}(s) = f(s) - \frac{1}{S}\sum_{i=0}^{S} f(i) \tag{3.1}$$

where $S$ is the total number of samples. Ideally, the signal oscillates around zero, leading to a mean of zero. A non-zero mean indicates a DC-shift, which can be removed by subtracting it from every sample. The pre-emphasis is defined as

$$\hat{f}(s) = f(s) - a \cdot f(s-1) \quad , \tag{3.2}$$

where the pre-emphasis factor $a$ is typically set to a value around 0.97. The resulting high-pass filter helps to compensate for the high-frequency component that was suppressed during the human voice production. Dithering slightly smoothes the signal with small random numbers to ease errors from the quantization process (*e.g.*, [Pohl 05]):

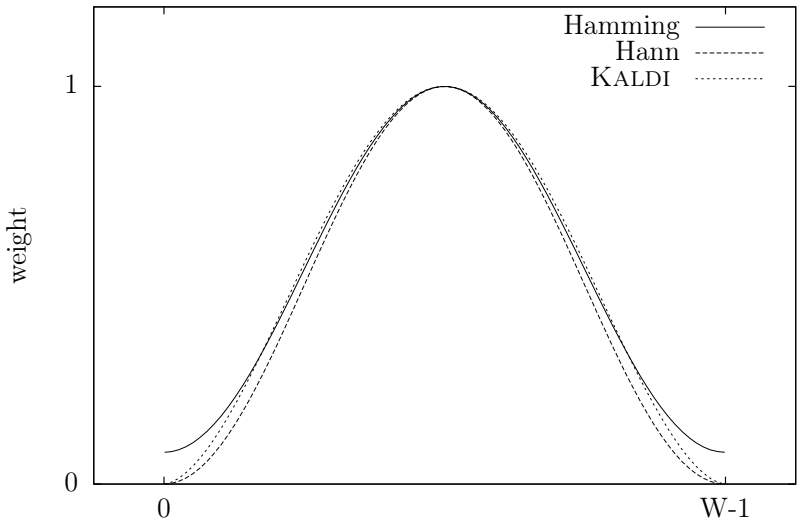
$$\bar{f}(s) = f(s) + \text{nextRand}() \cdot b \tag{3.3}$$

where the dithering factor $b$ is typically set to 1 and *nextRand* draws the next random number from a Gaussian with zero mean and variance one.

More complex preprocessing methods include source separation in presence of multiple speakers [Mand 09] or dereverberation in presence of room reverberation [Leba 01, Zah 10]. These are however not further considered as the available data is from a single speaker using a close-talking microphone, thus almost eliminating speech overlap and reverberation.

### 3.2.2 Mel Frequency Cepstral Coefficients

The *de facto* standard for speech recognition is to extract Mel frequency cepstral coefficients (MFCC) [Davi 80]. Their computation is detailed in the relevant literature [Schu 95, Huan 01, Bene 08]. The input signal is cut into a sequence of overlapping frames using a sliding window. Typically, segments of 16 to 25 ms are extracted every 10 ms. To avoid artifacts in the later frequency analysis, the samples within a frame are weighted using a window function. Fig. 3.3 shows the Hamming, Hann and KALDI windows in comparison. The latter is a compromise between the shape of the Hamming window and a zero fade-in and fade-out.




| *name* | *weighting function* |
|---|---|
| Hamming | $\text{hamming}(n) = 0.54 - 0.46 \cos\left(\frac{2n\pi}{W-1}\right)$ |
| Hann | $\text{hann}(n) = 0.50 - 0.50 \cos\left(\frac{2n\pi}{W-1}\right)$ |
| KALDI | $\text{kaldi}(n) = \left(0.50 - 0.50 \cos\left(\frac{2n\pi}{W-1}\right)\right)^{0.85}$ |

Figure 3.3: Comparison of different windowing functions; $W$ is the window sample size.

The power spectrum is computed from the fast Fourier transform (FFT) applied to each window, and is compressed using a Mel filter bank (*cf.* Fig. 3.4). Beside the reduction of coefficients, the Mel filter bank is motivated by the way humans perceive frequencies: different regions of the inner ear are stimulated by a certain frequency band. To simulate the human logarithmic loudness perception, the logarithm is applied to each filter bank coefficient.

As a final step, the discrete cosine transform (DCT) is applied to decorrelate the dimensions and obtain the *cepstrum*. Typically, the first twelve coefficients $c_0, \ldots, c_{11}$ are used; $c_0$ is often substituted by the (normalized) short-time energy, that is the sum of all absolute values within a window. Fig. 3.5 summarizes the computation of the MFCCs for an example 16 ms window of a 16 kHz signal. Note that the power spectrum has half the number of coefficients than the input frame due to the symmetry property of the FFT for real valued signals.

### 3.2.3 Delta Coefficients

The above coefficients are called *static* features as their computation does not depend or reflect their temporal context. In order to model the *change* of the coefficients over time, for example to catch a characteristic phoneme transition, the first and second order derivatives are computed. The moments are approximated by computing the

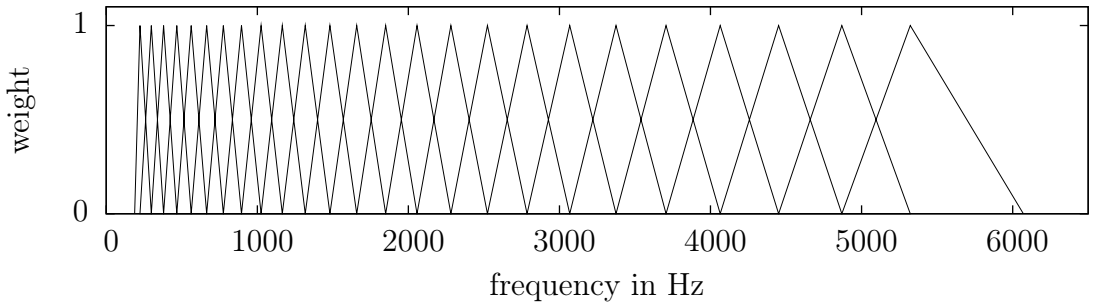



Figure 3.4: Mel filter bank consisting of 25 triangular filters, which are equidistant on the Mel-scale.

slope of the regression line over a context of consecutive frames. These dynamic *delta* features can be computed as suggested by Furui [Furu 86]:

$$\triangle c_q(t) = \frac{\sum_{j=-\tau}^{\tau} j \cdot c_q(m+j)}{\sum_{j=-\tau}^{\tau} j^2} \tag{3.4}$$

where $c_q(t)$ is the $q$-th coefficient at time $t$ and $\tau$ is the context size, typically $\tau = 2$. The delta coefficients cover a temporal context of $2 \times \tau$ times the window shift plus the window length. The static and dynamic features are concatenated to form the observation vectors $o_t$.

## 3.3 Hidden Markov Models

Following the extraction at the acoustic front-end, the observation sequence $\mathcal{O} = \{o_1, \ldots, o_T\}$ needs to be mapped to the most likely sequence of phones or words. Hidden Markov Models (HMMs) are the *de facto* standard statistical model for speech recognition, and are in detail explained in for example [Schu 95, Huan 01, Bene 08]. An HMM is a statistical automaton consisting of several states and possible transitions. In contrast to other automata, an HMM changes its state depending on probabilities instead of input tokens, and emits a symbol on entering a state, again with a certain probability. An HMM is generative in the way that a path through the model, *i.e.*, a state sequence, produces a certain sequence of observations with a certain production probability.

More formally, an HMM has $S$ states. It initially enters a certain state $i$ with an *entry probability* $\pi_i$ and allows transitions from state $i$ to state $j$ with a certain *transition probability* $a_{ij}$. Following a transition, the target state $i$ emits a symbol $o$ with a certain *emission probability* $b_i(o)$. Depending on the type of $o$, this distribution is discrete or continuous, and is typically modeled as discrete probabilities, Gaussian mixture models, or artificial neural networks. These parameters of the complete model are denoted as $\Theta = \{\boldsymbol{\pi}, \boldsymbol{A}, \boldsymbol{B}\}$, where $\boldsymbol{\pi} = \{\pi_1, \ldots, \pi_S\}$, $\boldsymbol{A} = \{a_{11}, \ldots, a_{SS}\}$ and $\boldsymbol{B} = \{b_1(\cdot), \ldots, b_S(\cdot)\}$. Fig. 3.6 shows an example HMM with three states that allows transitions to the current state ("loops") and the subsequent state. This tran-

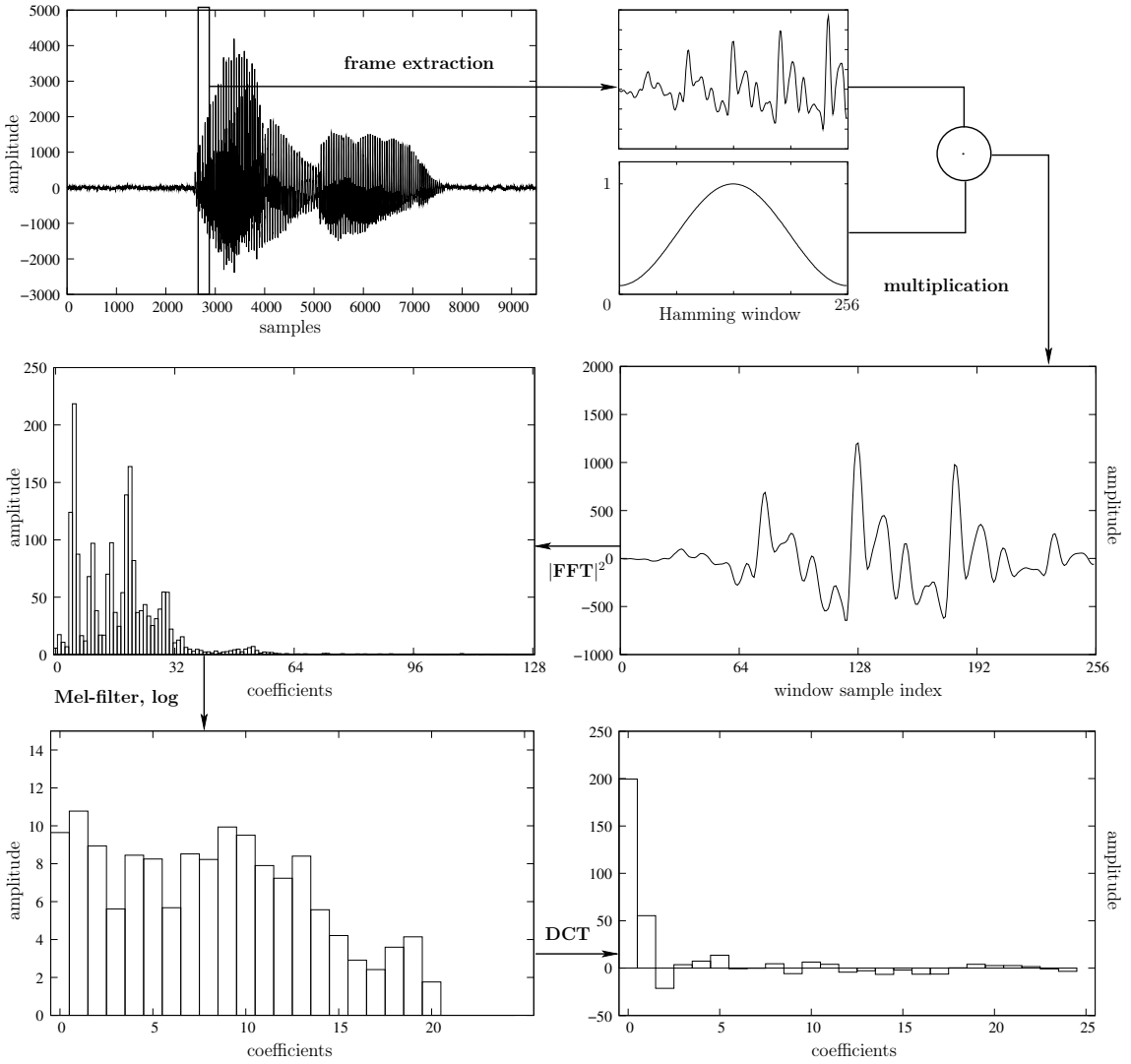



Figure 3.5: Schematic description of the MFCC feature extraction [Ried 07]. Starting top left, a frame is extracted from the continuous speech signal. Following the arrows, the processing steps are: Hamming window, power spectrum, Mel filter, logarithm and DCT. In the last step, the 0-th coefficient is replaced by the short-time energy of the initial signal frame. The spoken word is the German "hallo," the example window is from the onset of the sound `/a/`.

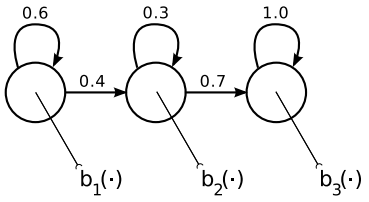



Figure 3.6: A simple three state linear HMM without entry probabilities.

sition model is called "linear" and is typically used to model time-ordered sequences of observations such as speech.

Consider the following toy problem. The speech recognition system knows several words units, each represented by an HMM. Recognizing a certain word is to determine which of the HMMs is most likely to produce the given input sequence of observations, *i.e.*, to decide for the model with the highest production probability $P(\mathcal{O}|\Theta)$. The underlying state sequence, however, remains hidden, which gives the HMM its name.

### 3.3.1 Forward and Backward Probabilities

The production probability of an observation sequence can be computed using the forward or backward probabilities. The *forward* probabilities $\alpha$ explore the HMM from the beginning using a dynamic program

$$\alpha_1(j) = \pi_j b_j(o_1) \quad \forall j \tag{3.5}$$

$$\alpha_t(j) = \left[\sum_i \alpha_{t-1}(i) a_{ij}\right] b_j(o_t) \quad \forall j, t = 2, \ldots, T \tag{3.6}$$

$$P(\mathcal{O}|\Theta) = \sum_j \alpha_T(j) \tag{3.7}$$

where $\pi_j$ is the entry probability of state $j$, $b_j(o_t)$ is the probability that state $j$ emits $o_t$, and $P(\mathcal{O}|\Theta)$ is the overall production probability.

Equivalently, the *backward* probabilities $\beta$ deduce the production probability beginning at the end of the observation sequence, again using a dynamic program

$$\beta_T(i) = 1 \quad \forall i \tag{3.8}$$

$$\beta_t(i) = \sum_j a_{ij} b_j(o_{t+1}) \beta_{t+1}(j) \quad \forall i, t = (T-1), \ldots, 1 \tag{3.9}$$

$$P(\mathcal{O}|\Theta) = \sum_j \pi_j b_j(o_1) \beta_1(j) \tag{3.10}$$

The $\alpha$ and $\beta$ probabilities can be used to express certain other probabilities within the HMM framework, most importantly the probability to "be in state $i$ at time $t$ and observing $o_t$", defined as

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{\sum_l \alpha_t(l)\beta_t(l)} \tag{3.11}$$

which will be required later in this chapter. Regarding certain transitions, the probability to "be in state $i$ at time $t$ and state $j$ at time $t+1$ and observing $o_t$" is

$$\xi_t(i,j) = \frac{\alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j)}{\sum_{k,l}\alpha_t(k)a_{kl}b_l(o_{t+1})\beta_{t+1}(l)} = \frac{\alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j)}{\sum_l \alpha_t(l)\beta_t(l)} \quad . \tag{3.12}$$

### 3.3.2 Viterbi Path

The recognition system might not only allow single words, but word sequences. Instead of allocating an additional, longer model for every possible sequence, the existing models can be concatenated to form the word sequences. The most likely word sequence can again be determined by computing the production probabilities of the joint models. However, it might be of interest which segments of the observation sequence belong to which word, *i.e.*, which observations are associated with which state. This *state alignment* can be computed using the Viterbi algorithm, a dynamic programming technique:

$$\begin{aligned}
\phi_1(j) &= \pi_j b_j(o_1) \quad \text{and} \quad \psi_1(j) = 0 && (3.13)\\
\phi_t(j) &= \max_i[\phi_{t-1}(i)a_{ij}]b_j(o_t) && (3.14)\\
\psi_t(j) &= \text{argmax}_i\phi_{t-1}(i)a_{ij} \quad \forall j, t = 2,\ldots,T && (3.15)\\
P(\mathcal{O}|\Theta) &= \max_j\phi_T(j) \quad \text{and} \quad q_T = \text{argmax}_j\phi_T(j) && (3.16)\\
q_t &= \psi_{t+1}(q_{t+1}) \quad t = (T-1),\ldots,1 && (3.17)
\end{aligned}$$

where $\boldsymbol{q} = \{q_1,\ldots,q_T\}$ is the most likely state sequence regarding the observation $\mathcal{O} = \{o_1,\ldots,o_T\}$. If $q_T$ is manually set to the last state of the HMM, the resulting trace is called a *forced alignment* that can be used for model training which is sketched in the next section.

### 3.3.3 Parameter Estimation

The prior sections assumed already existing HMM parameters. This section describes how to iteratively learn these HMM parameters $\Theta$ from an initial, often uniform, estimate. Given training observation sequences and an initial model, the $\alpha$, $\beta$, $\gamma_t(i)$ and $\xi_t(i,j)$ probabilities can be used to formulate an iterative update of the entry and transition probabilities, using the *Baum-Welch* formulas:

$$\begin{aligned}
\hat{\pi}_i &= \gamma_1(i) \quad \forall i && (3.18)\\
\hat{a}_{ij} &= \frac{\sum_{t=1}^{T-1}\xi_t(i,j)}{\sum_{t=1}^{T-1}\gamma_t(i)} = \frac{\sum_{t=1}^{T-1}\alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j)}{\sum_{t=1}^{T-1}\alpha_t(i)\beta_t(i)} && (3.19)\\
\text{subject to} \quad \hat{a}_{ij} &= \frac{\hat{a}_{ij}}{\sum_k \hat{a}_{ik}} \quad \text{and} \quad \sum_i \pi_i = 1 \quad . && (3.20)
\end{aligned}$$

In case of multiple training sequences, the sums in both the numerator and denominator of the update formulas are extended by a sum over all sequences. The numerator and denominator sums are also called *accumulators* or *sufficient statistics*, as together they are sufficient to re-estimate the model parameters.

The Baum-Welch training comes at a fairly high computational complexity, as every possible path through the HMM is considered. A faster, yet often sufficient alternative is the Viterbi training. Instead of computing the $\alpha$ and $\beta$ values, the Viterbi path $\boldsymbol{q}$ is computed for each training sequence. The entry and transition probabilities are estimated from the relative counts of the individual transitions as

$$\hat{\pi}_i = \frac{\chi_{[q_1=i]}}{\sum_j \chi_{[q_1=j]}} \tag{3.21}$$

$$\hat{a}_{ij} = \frac{\sum_t^{T-1} \chi_{[q_t=i,q_{t+1}=j]}}{\sum_j \sum_t^{T-1} \chi_{[q_t=i,q_{t+1}=j]}} \quad . \tag{3.22}$$

Again, the sums are extended in presence of multiple training instances.

The update of the emission probabilities $\boldsymbol{B}$ is independent of the entry and transition probabilities and will be discussed in the later sections. For the Viterbi training, however, the computation of the sufficient statistics is simplified by setting

$$\gamma_t(i) = \begin{cases} 1 & \text{if } q_t = i \\ 0 & \text{otherwise.} \end{cases} \tag{3.23}$$

## 3.4 Phonetic Modeling

Individual HMMs for each word or word sequence are only appropriate for very small vocabulary such as isolated digits or letters. Medium to large vocabularies (5 000 to 50 000 words) require a more fine grained model: The HMMs are used to represent phonemes in context while a *lexicon* specifies how to synthesize word models by concatenating the appropriate phoneme models. This helps to keep the number of parameters constant while allowing an arbitrary number of words. For example, the *Standard American English (SAE)* is formed by 25 consonants, 19 vowels, plus one consonant and three vowels for foreign words [Well 00]. Often, these phonemes are not mapped to models one to one but similar or rare phonemes are represented by a single model. Additional models are used to represent silence, noise and non-verbal events such as coughing or laughter.

In the simplest case, each phoneme is, regardless of its context, associated with one HMM. This *mono-phone* setup has a small number of parameters, but neglects the fact that phonemes sound different depending on the context. The human articulatory system is continuously changed to while speaking, thus the articulators can have a different position for the same phoneme depending on the past and future phoneme. In consequence, different models are instantiated for different phonemes in context, for example *bi-phones* modeling either left *or* right context, or the most common *tri-phones*, that distinguish phoneme models by the left *and* right context. To account for continuous speech, the context can also be considered across words. The mapping between model parameters and phonemes-in-context to compose a certain word is described in detail in Sec. 3.6.1.

## 3.5 Weighted Finite State Transducers

For any given training utterance, a large HMM is composed by concatenating the phoneme models according to the transcription and the lexicon[1]. For this meta HMM, the Baum-Welch and Viterbi training formulas can be used to update the individual parameters.

The decoding, *i.e.*, the recognition of previously unseen observation sequences, is however more complex. A standard strategy is to apply a beam Viterbi search (*e.g.*, [Schu 95, Huan 01]): Initially, all possible words are added to the search beam. At each possible word boundary, *i.e.*, if the search is in the final state of the last phoneme model of a word, a *language model* predicts the most likely subsequent word(s), which are then expanded to the respective model sequences, and added to the search space. The search beam is pruned to keep only a little number of promising word sequences.

A more elegant way to integrate phonetic modeling, lexicon and language model is based on weighted finite state acceptors and transducers. Mohri *et al.* give a comprehensive introduction which is summarized in the following paragraphs [Mohr 02].

### 3.5.1 Definitions

**Semiring** The following definitions rely on a semiring $(\mathbb{K}, \oplus, \otimes, \bar{0}, \bar{1})$ that has an associative and commutative operation $\oplus$ and an associative operation $\otimes$, with identities $\bar{0}$ and $\bar{1}$, respectively, such that $\otimes$ distributes over $\oplus$ and $\bar{0} \otimes a = a \otimes \bar{0} = \bar{0}$. An example semiring is $(\mathbb{N}, +, \cdot, 0, 1)$. The weights used in speech recognition are usually probabilities, thus a suitable semiring is $(\mathbb{R}, +, \cdot, 0, 1)$.

**Weighted Acceptors** Hidden Markov models are special cases of weighted finite state acceptors (WFSA). A WFSA $A = (\Sigma, Q, E, i, F, \lambda, \varsigma)$ over the semiring $\mathbb{K}$ is defined by an input label set $\Sigma$, a set of states $Q$, a set of transitions $E \subseteq Q \times (\Sigma \cup \{\epsilon\}) \times \mathbb{K} \times Q$, an initial state $i \in Q$, a set of final, or *accepting*, states $F \subseteq Q$, an initial weight $\lambda$, and a final weight function $\varsigma$. Graphically, a transition $t = (s, l, w, n) \in E$ is represented as an arc from the source state $s$ to the target state $n$. The transition requires the input label $l$ and is associated with the weight $w$. Transitions labeled with the empty symbol $\epsilon$ consume no input. A successful path $\pi = \{t_1, \ldots, t_n\}$ is a path from the initial state $i$ to a final state $f \in F$, and corresponds to the sequence of respective input symbols. The weight associated with $\pi$ is the $\otimes$-product of the initial weight, the weights of the transitions, and the final weight $\varsigma$ of the last state. Thus, a WFSA can be used to map input symbol sequences to weights [Salo 78].

**Weighted Transducers** Weighted finite state transducers (WFST) extend WFSAs by replacing the individual input labels of the transitions with pair $(i, o)$ of an input label $i$ and output label $o$. As a result, a WFST translates ("transduces") a sequence of input labels into a sequence of output labels, also associated with a weight.

[1] A detailed transcription or recognition run is required in case of possible pronunciation alternatives.

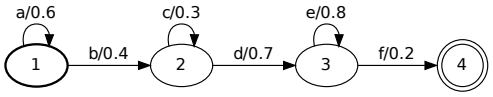



Figure 3.7: A three-state linear HMM defined as a weighted finite state acceptor using an auxiliary fourth state to model the exit probability.

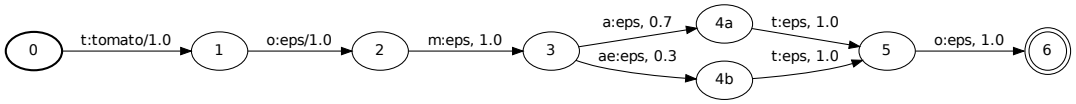



Figure 3.8: Lexicon WFST $L$ representing two pronunciation alternatives of the word "tomato."

The figures 3.7-3.9 show simple examples of how to use WFSTs in speech recognition. The bold circles represent the initial states, the double-circles the final states. In all figures, the initial weight is omitted because it is set to $\lambda = \bar{1}$.

### 3.5.2 Operations on WFST

**Composition** Two WFST $R$ and $S$ can be composed, such that the resulting transducer $T = R \circ S$ has exactly one path that maps sequence $u$ to sequence $w$, where $R$ maps $u$ to some sequence $v$, and $S$ maps $v$ to $w$. The weights of a path in $T$ are the $\otimes$-product of the weights of the corresponding paths in $R$ and $S$ [Salo 78]. The composition can be explicit, resulting in a possibly over-sized WFST, or *on-demand*, where the composition is only generated as needed.

**Determinization** A WFST can be *non-deterministic*, *i.e.*, there exists at least one state that has multiple transitions with the same label, which is undesirable for the

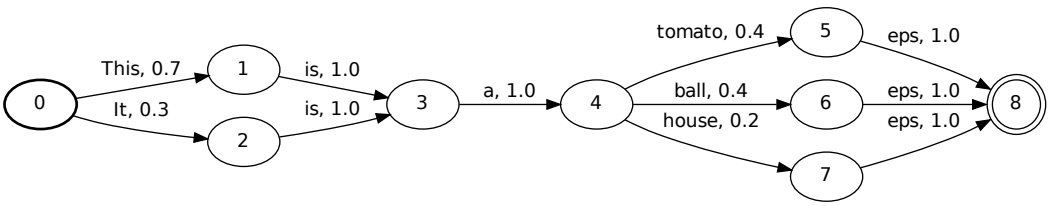



Figure 3.9: Grammar WFSA $G$ representing a simple language model.

| | *input* | *output* |
|---|---|---|
| $H$ | emission prob. ID | context-dependent phone |
| $C$ | context-dependent phone | phone |
| $L$ | phone | word |
| $G$ | word | $(\text{word})^*$ |

Table 3.1: Input and output labels of the WFST that are composed to the final WFST. The transitions of $G$ have the same output and input symbol.

later decoding. Weighted determinization, a generalization of the subset method for determinizing finite automata [Aho 86], can be applied to a weighted automaton to compute an equivalent deterministic automaton. In contrast to the unweighted case, not all weighted automata can be determinized, however, most automata used in speech processing can be either determinized directly, or easily made determinizable by applying certain transformations [Mohr 97, Mohr 02].

**Minimization** In the same way as regular deterministic automata, every deterministic weighted automata can be minimized [Mohr 97]. The resulting weighted automaton is equivalent to the original, but has the least number of states and transitions among all other equivalent automata.

### 3.5.3 Speech Recognition using WFST

#### Transducer Composition

The WFST and the related operations allow a very intuitive design of a speech recognition system. The general idea is to model several layers of the system independently using WFSTs, and then combine them using composition. To reduce the complexity of the resulting WFST and to allow a simple decoding, the automata are determinized and minimized.

Beginning top-down, the first layer is the *grammar* WFSA $G$ that models the possible word sequences to be recognized based on a language model (*e.g.*, Fig. 3.9; see Sec. 3.8.1 for further details). The $G$ automaton is composed with the *lexicon* WFST $L$ that takes phoneme symbols as input and produces words as output (*e.g.*, Fig. 3.8). The input to $L$ is produced by an automaton $C$ that models the context dependency of the phones [Mohr 02]. The idea is to model a transducer that outputs a sequence of phoneme symbols but where the paths depend on the left and right phonetic context to have different models for the same phone in different context. The last automaton in the processing chain is the WFST $H$ that models the actual HMM. It uses an emission distribution (or state) ID as input, which is used by the decoder to compute the actual emission probability, and outputs context-dependent phones. The automaton is determinized and minimized after each composition.

The final WFST is $R = H \circ C \circ L \circ G$. Tab. 3.1 shows the input/output configurations prior to composition.

### Training and Decoding

Both training and decoding are based on the path within the compound WFST $R$ that represents the state sequence of the underlying HMMs. The Viterbi algorithm presented in Sec. 3.3.2 can be implemented straight forward only for small WFST as the memory complexity is the number of possible states times the number of observations. Longer utterances or utterances including pronunciation alternatives and optional silences result in larger WFST which result in an unacceptably high complexity: typically, the recognition WFST has millions of states for large vocabularies. However, there are typically only a small number of likely paths, and for each path, likely next steps based on the observations. This is exploited by a modification of the Viterbi algorithm. Instead of a full matrix that represents the most probable path for each state at a certain time, the time-synchronous Viterbi *beam search* (*e.g.*, [Huan 01]) considers only a certain number of best paths, the *beam*. For each time step, each path in the beam is extended by all possible transitions before the beam is again reduced to a small number of best candidates. This pruning step is either implemented by considering a fixed number $b$ of paths, or by allowing paths that are within a certain likelihood range of the current best path. The algorithm is outlined below.

1. *Initialization*— For all initial states of $R$, add a path to the beam.
2. *Iterate*— For time $t = 1, \ldots, T$
   (a) Expand all paths and add them to the beam.
   (b) Compute the current cost of all paths and keep the $b$ best paths.
3. *Termination*— Pick the best path according to the current cost and generate word sequence from the state trace.

To account for the different ranges of acoustic and language model scores, the acoustic scores obtained from the AM are scaled using an acoustic weight $\mathrm{w}_a$. Shorter words will induce less weight on the WFST, to avoid a preference of short (and thus many) words, a word insertion penalty $\mathrm{w}_{ip}$ is added for each word transition.

The Kaldi training procedure uses the Viterbi beam search to decode the forced alignments WFST to obtain the state alignments required for the acoustic model training. In contrast to the recognition WFST, the forced alignment WFST uses a reduced $G$ automaton that only allows the word sequence based on the transcription. This significantly speeds up the training process: the typically small alignment WFST can be decoded using small beam sizes. If the decoding fails, *i.e.*, no path reached a final state at $t = T$, the utterance is discarded from the training.

## 3.6 Acoustic Modeling

The emission probability IDs of the $H$ transducer are the juncture between the WFST-based part that covers the phonetic, lexical and language model, and the statistical model of the acoustic observations. Each state of $H$ is associated with such an ID that links to an emission probability in the acoustic model.

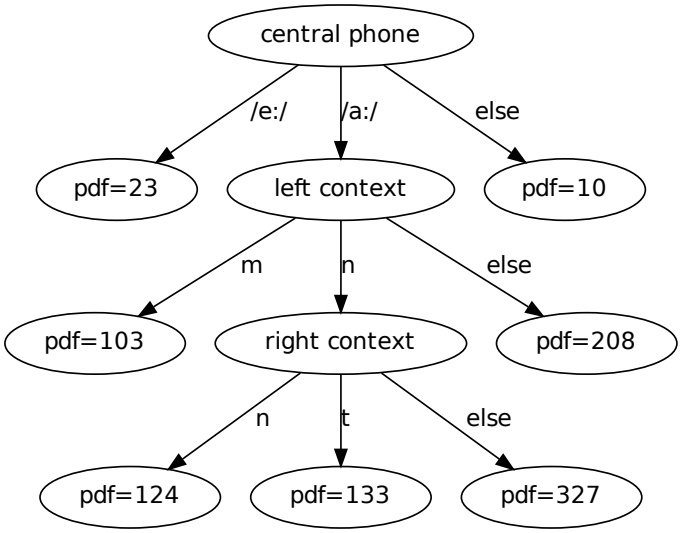



Figure 3.10: A decision tree based on questions regarding the central and left/right context phones as described in [Schu 95]. The order of the splits is heuristically defined and leads to increasing context size with increasing depth of the tree.

### 3.6.1 Acoustic-Phonetic Decision Tree Building

The acoustic-phonetic decision tree maps a certain phonetic context, *e.g.*, `b/a/t` (a *tri*-phone with central phone `/a/`, left context `/b/`, right context `/t/`), together with an HMM state index, to an emission probability density function (PDF). The tree is used to compile the $H$ WFST. The primary purpose of the tree is to tie the statistical parameters of certain phones with similar acoustic context to reduce the overall number of parameters, which is necessary even for small context sizes. For example, consider a phonetic alphabet of 39 phonemes[2]. A context size of three means that there are theoretically $39^3 = 59,319$ different tri-phones. Even after removing these tri-phones that do not appear in the training set, there will still be a large number of tri-phones that occur only a couple of times, thus the statistical parameters have to be learned from sparse data which is obviously unfavorable. Parameter (or state) tying is to use the same statistical parameters for different phones or states in context. For example, instead of modeling `b/a/t`, `p/a/t` and `t/a/t` explicitly, the three models could be tied together as `P/a/t`, where `P` stands for plosive. The corresponding question within a decision tree would be "left context is a plosive."

There are various ways of constructing the acoustic-phonetic decision tree. Schukat-Talamazzini suggests a heuristic set of questions that first ask about the central phone and subsequently split on the left and right context until a certain context size is reached [Schu 95]. In a post-clustering, phones in context that appear less than 50 times in the training set are removed. Fig. 3.10 shows a small example decision tree.

[2]Often, rare but similar phonemes are mapped to the same phonetic symbol.

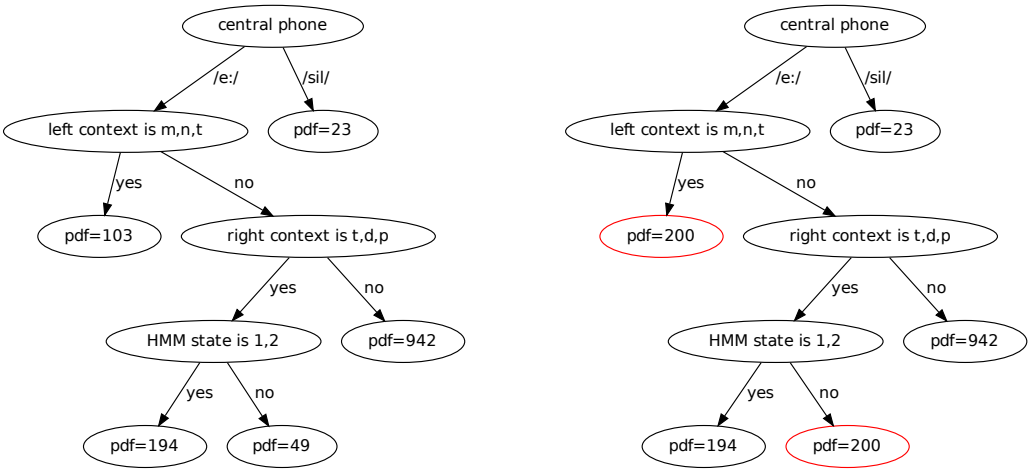



Figure 3.11: *Left:* A decision tree based on questions regarding the central and left/right context phones; the order of the splits is determined using a greedy maximum likelihood search. *Right:* The post-clustering ties certain leafs, here to PDF 200.

The tree building algorithm implemented in the KALDI toolkit is more flexible. The splitting procedure asks questions not only about the central and context phones, but also about the HMM state. The actual questions are not hand-crafted but automatically generated based on acoustic similarities. The splits are not executed according to a fixed heuristic, but in a greedy way to optimize the likelihood of the training data based on a single Gaussian associated with each tree leaf. These Gaussians are collected for each state in context from the state alignments of the training data. The post-clustering process typically reduces the number of leaves by around 10-20%, and is also implemented in a greedy way to minimize the loss of likelihood [Pove 11b]. Fig. 3.11 shows an example tree before and after post-clustering.

### 3.6.2 Continuous Hidden Markov Models

Hidden Markov models with continuous emission probabilities have been used for a long time (*e.g.*, [Merg 85, Zhao 91]), and are still the most frequent choice of model. For continuous models, every tree leaf is associated with a separate Gaussian mixture model. Formally, the emission probability of tree leaf $j$ is defined as

$$p(\mathbf{o}|j) = \sum_{i=1}^{N_j} c_{ji} \mathcal{N}(\mathbf{o}; \boldsymbol{\mu}_{ji}, \boldsymbol{\Sigma}_{ji}) = \frac{1}{(2\pi)^{\frac{d}{2}} \sqrt{|\boldsymbol{\Sigma}_{ji}|}} \exp\left(-0.5(\mathbf{o} - \boldsymbol{\mu}_{ji})^T \boldsymbol{\Sigma}_{ji}^{-1} (\mathbf{o} - \boldsymbol{\mu}_{ji})\right) \tag{3.24}$$

where $N_j$ is the number of Gaussians assigned to leaf $j$, and the $\boldsymbol{\mu}_{ji}$ and $\boldsymbol{\Sigma}_{ji}$ are the means and covariance matrices, respectively.

To initialize the models, each mixture is populated with a single Gaussian which is subsequently split throughout the training iterations, until a certain number of

total Gaussians is reached. For each split, the number of Gaussians is allocated proportional to a small power (*e.g.*, 0.2) of the leaf's occupation count. That way, states with a large amount of training data will be modeled by a larger number of Gaussians than states that are rather infrequent.

The update of the components of each Gaussian mixture requires the definition of the probability to "be in state $j$ at time $t$ with the $k$-th component active" as

$$\gamma_t(j,k) = \gamma_t(j) \cdot \frac{c_{jk}\mathcal{N}(\mathbf{o}_t; \boldsymbol{\mu}_{jk}, \boldsymbol{\Sigma}_{jk})}{\sum_i c_{ji}\mathcal{N}(\mathbf{o}_t; \boldsymbol{\mu}_{ji}, \boldsymbol{\Sigma}_{ji})} \quad . \tag{3.25}$$

With that definition, the weights, means and covariances are re-estimated using the following formulas (*e.g.*, [Schu 95, Huan 01].

$$\hat{c}_{jk} = \frac{\sum_t \gamma_t(j,k)}{\sum_t \sum_m \gamma_t(j,m)} \tag{3.26}$$

$$\hat{\boldsymbol{\mu}}_{jk} = \frac{1}{\sum_t \gamma_t(j,k)} \sum_t \gamma_t(j,k)\mathbf{o}_t \tag{3.27}$$

$$\hat{\boldsymbol{\Sigma}}_{jk} = \left[\frac{1}{\sum_t \gamma_t(j,k)} \sum_t \gamma_t(j,k)\mathbf{o}_t\mathbf{o}_t^T\right] - \hat{\boldsymbol{\mu}}_{jk}\hat{\boldsymbol{\mu}}_{jk}^T \tag{3.28}$$

### 3.6.3 Semi-Continuous Hidden Markov Models

In the scope of this thesis, the KALDI toolkit was extended by two types of semi-continuous HMM and related smoothing techniques which are described in the following sections.

#### Traditional Implementation

The idea of semi-continuous models is to have a large number of Gaussian components that are shared by every tree leaf using individual weights [Huan 89]. The emission probability of tree leaf $j$ can be computed as

$$p(\mathbf{x}|j) = \sum_{i=1}^{N} c_{ji}\, \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \tag{3.29}$$

where $c_{ji}$ is the weight of Gaussian $i$ for leaf $j$. Tying all means and covariances has several benefits. For the model training, increasing the number of tree leaves introduces rather few new parameters ($N$ weights per state) while the Gaussian parameters are still estimated from the same (large) amount of data, which allows to reliably estimate full covariances. For the model evaluation, the Gaussians need to be evaluated only once for each feature vector. The individual emission probabilities are then computed by a simple multiplication and addition.

Another advantage is the initialization and use of the codebook. It can be initialized and adapted in a fully unsupervised manner using expectation maximization (EM) [Demp 77], maximum a posteriori adaptation (MAP) [Reyn 00], maximum likelihood linear regression (MLLR) [Gale 96] or similar algorithms — on solely acoustic data, *i.e.*, without the need of transcriptions or a Viterbi alignment.

The parameter update follows in principle the update of the continuous model parameters. The accumulators of the means and covariances are, however, shared among all states.

### Two-Level Tree based Semi-Continuous Hidden Markov Models

It is possible to combine the strengths of continuous and semi-continuous models using a special two-level acoustic-phonetic decision tree and several codebooks. This is also referred to as a state-clustered tied-mixture (SCTM) system [Pras 05]. The general idea is to identify acoustically similar states (in context) which will share the same codebook prior to the actual tree building. The KALDI implementation is to first build a coarse decision tree where each leaf corresponds to a codebook, *i.e.*, the Gaussians are tied at this level. The tree leaves are then further split to give a more fine-grained resolution. Each new leaf is associated with the corresponding codebook. The leaves resulting from the second splitting correspond to the actual context-dependent HMM states, and contain the individual weights as with the semi-continuous model. This type of system is typically not post-clustered, as the number of parameters is better controlled by the number of codebooks.

Formally, each context dependent state $j$ is associated with a codebook $k$ by a function $m(j) \rightarrow k$. The likelihood function can then be written as

$$p(\mathbf{x}|j) = \sum_{i=1}^{N_{m(j)}} c_{ji}\, \mathcal{N}\left(\mathbf{x}; \boldsymbol{\mu}_{m(j),i}, \boldsymbol{\Sigma}_{m(j),i}\right) \tag{3.30}$$

where $\boldsymbol{\mu}_{m(j),i}$ is the $i$-th mean vector of the codebook associated with $j$.

The initialization process is also a hybrid between the continuous and semi-continuous initialization strategy. Starting from the tree statistics, the individual Gaussians associated with the leaves are distributed to the respective codebooks based on the tree mapping to form preliminary codebooks. The final target size $N_k$ of codebook $k$ is determined with respect to the state occupancies of the respective leaves as

$$N_k = N_0 + \frac{\sum_{l \in \{m(l)=k\}} \mathrm{occ}(l)}{\sum_t \mathrm{occ}(t)} \left(N - K \cdot N_0\right) \tag{3.31}$$

where $N_0$ is a minimum number of Gaussians per codebooks (*e.g.*, 3), $N$ is the total number of Gaussians, $K$ the number of codebooks, and $\mathrm{occ}(j)$ is the occupancy of tree leaf $j$. A different way to set the target codebook sizes is to use a power law similar to the initialization of the continuous models:

$$N_k = N_0 + \frac{\left(\sum_{l \in \{m(l)=k\}} \mathrm{occ}(l)\right)^q}{\sum_r \left(\sum_{t \in \{m(t)=r\}} \mathrm{occ}(l)\right)^q} \left(N - K \cdot N_0\right) \quad . \tag{3.32}$$

Here, $q$ controls the influence of the occupancy regarding one codebook. A typical value of $q = 0.2$ leads to rather homogeneous codebook sizes while still attributing more Gaussians to states with larger occupancies, a value of $q = 1$ reduces Eq. 3.32 to Eq. 3.31.

The target sizes $N_k$ are again enforced by splitting and merging the components.

### Interpolations for Semi-continuous Models

**Intra-Iteration Smoothing** Although increasing the state context resolution does not imply a huge increase of the number of parameters for semi-continuous systems, the estimation of the per-state Gaussian weights $c$ suffers from data fragmentation, especially when training the models with small amounts of data. The *intra-iteration* smoothing is motivated by the fact that tree leaves with the same root are acoustically similar, thus, if the sufficient statistics of the weights of a leaf $j$ are small, they should be interpolated with the statistics of closely related leaf nodes.

This is implemented in a two step algorithm. First, the sufficient statistics are propagated bottom-up the tree in a way that the statistics of any node is the sum of its children's statistics. Second, the statistics of each node and leaf are interpolated top-down with their parent's statistics using an interpolation weight $\tau$:

$$\hat{\gamma}_{ji} \leftarrow \underbrace{\left(\gamma_{ji} + \frac{\tau}{\left(\sum_k \gamma_{\mathcal{P}(j),k}\right) + \epsilon}\,\gamma_{\mathcal{P}(j),i}\right)}_{:=\bar{\gamma}_{ji}} \cdot \underbrace{\frac{\sum_k \gamma_{jk}}{\sum_k \bar{\gamma}_{jk}}}_{\text{normalization}} \quad , \tag{3.33}$$

where $\gamma_{ji}$ is the occupancy of the $i$-th component of tree leaf $j$, and $\mathcal{P}(j)$ is the parent node of $j$. In case of a two-level tree architecture, the propagation and interpolation cannot be applied across different codebooks, as the number of components may vary, and the statistical relation of the individual statistics does not necessarily hold.

This is similar to the $P$ and $I$ steps of the *APIS* algorithm [Schu 94, Schu 95] but smoothes the statistics with respect to their counts instead of replacing them. This modification is necessary as the tree in [Schu 95] is constructed in a way, that the tree splits (nodes) represent acoustic models (the *poly-phones*), *i.e.*, the first node level would be the mono-phones, and any phone with more context would be a child node, with only the phones with the largest context being a leaf. For the training, any data contributing to a node $j$ should also contribute to its parent $\mathcal{P}(j)$, and if a child has insufficient statistics, these can be interpolated with the parent's. As the models are only attached to the leaves, this motivation does not (necessarily) hold, thus requiring the above modification of the interpolation scheme.

**Inter-Iteration Smoothing** A typical problem that arises when training semi-continuous systems is that the different types of parameters converge at a different rate. Especially the weights tend to converge to a poor local optimum over the iterations. Schukat-Talamazzini [Schu 95] suggested to smooth newly estimated parameters $\Theta^{(i)}$ with the ones from the prior iteration using an interpolation weight $\rho$:

$$\hat{\Theta}^{(i)} \leftarrow (1-\rho)\,\Theta^{(i)} + \rho\,\Theta^{(i-1)} \quad . \tag{3.34}$$

This can be interpreted as a Bayes estimation with the initial parameters as prior information. A constant $\rho$ leads to an exponentially decreasing impact of the initial parameters [Schu 95]. The smoothing is only applied to the Gaussian weights to prevent them from overfitting. The notion is to artificially slow down the convergence of the weights to allow the remaining Gaussian parameters to move farther away from the initial values, as they otherwise would.

### 3.6.4 Subspace Gaussian Mixture Models

A different, more sophisticated way to reduce the number of parameters is to select the Gaussian components from a subspace spanned by a universal background model. The emission likelihoods of the subspace Gaussian mixture models (SGMM) [Pove 11a] can be computed as

$$p(\mathbf{x}|j) = \sum_{i=1}^{N} c_{ji} \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_{ji}, \boldsymbol{\Sigma}_i) \tag{3.35}$$

$$\boldsymbol{\mu}_{ji} = \mathbf{M}_i \mathbf{v}_j \tag{3.36}$$

$$c_{ji} = \frac{\exp \mathbf{w}_i^T \mathbf{v}_j}{\sum_l^N \exp \mathbf{w}_l^T \mathbf{v}_j} \tag{3.37}$$

where the covariance matrices $k_i$ are shared among all states $j$. The weights $w_{ji}$ and means $m_{ji}$ are derived from $\mathbf{v}_j$ together with $\mathbf{M}_i$ and $\mathbf{w}_i$, thus they are all drawn from a common subspace from the underlying codebook. A derivation of the model and the respective update formulas goes beyond the scope of this thesis, but is described in detail in [Pove 11a].

## 3.7 Feature and Model Transformations

### 3.7.1 Linear Discriminant Analysis

The linear discriminant analysis (LDA) framework [Fish 36, Wilk 62] and the related linear transformations have successfully been applied to speech recognition [Hunt 91, Haeb 92]. The idea is to estimate a linear transformation $\mathcal{T}$ that helps to separate samples from different classes, *i.e.*, phonemes in speech recognition, while maintaining compact classes. Mathematically, the overall covariance is decomposed as a sum of the intra-class scatter $\boldsymbol{S}_a$ and the inter-class scatter $\boldsymbol{S}_b$:

$$\boldsymbol{S}_a = \sum_{i}^{K} p_i \boldsymbol{\Sigma}_i \tag{3.38}$$

$$\boldsymbol{S}_b = \sum_{i}^{K} p_i (\boldsymbol{\mu}_i - \boldsymbol{\mu})(\boldsymbol{\mu}_i - \boldsymbol{\mu})^T \tag{3.39}$$

$$\boldsymbol{\Sigma} = \boldsymbol{S}_a + \boldsymbol{S}_b \tag{3.40}$$

where $p_i$, $\boldsymbol{\mu}_i$ and $\boldsymbol{\Sigma}_i$ are the prior probability, mean and covariance matrix of class $i$, $K$ is the overall number of classes, and $\boldsymbol{\mu}$ is the overall sample mean.

The transformation $\mathcal{T}$ is found by solving

$$\text{maximize:} \quad \text{trace}(\mathcal{T}^T(\boldsymbol{S}_a + \boldsymbol{S}_b)\mathcal{T}) \tag{3.41}$$

$$\text{subject to:} \quad \text{trace}(\mathcal{T}^T \boldsymbol{S}_a \mathcal{T}) = \text{const.} \tag{3.42}$$

The optimum $\mathcal{T}$ is found by solving the related eigenvalue problem $\boldsymbol{S}_a^{-1}\boldsymbol{S}_b\mathcal{T} = \mathcal{T}\Lambda$, where $\Lambda$ is the diagonal covariance of the samples. The final transformed sample vector $\boldsymbol{y}$ is a translation and transformation [Hunt 91]

$$\hat{\boldsymbol{o}} = \mathcal{T}^T(\mathbf{o} - \boldsymbol{\mu}) \quad . \tag{3.43}$$

The class-dependent statistics are collected using Viterbi alignments.

### 3.7.2 Maximum Likelihood Linear Transformation

Similar to the human perception, the accuracy of a speech recognition system can be greatly improved by adapting to the speaker. The recognition system used for the later experiments is adapted using a constrained maximum likelihood linear transformation (MLLT), which is in detail described in [Gale 98]. The idea is to find a transformation $\mathcal{A}$ that transforms the mean vectors and covariance matrix of the Gaussian components of the acoustic model so that the likelihood $\mathcal{Q}$ of the available adaptation data is maximized.

$$\mathcal{Q}(\Theta|\hat{\Theta}) = K - \frac{1}{2}\sum_k \sum_t \gamma_t(j,k)\left(K_k + \log(|\hat{\boldsymbol{\Sigma}}_k|) + (\mathbf{o}_t - \hat{\boldsymbol{\mu}}_k)^T \hat{\boldsymbol{\Sigma}}_k^{-1}(\mathbf{o}_t - \hat{\boldsymbol{\mu}}_k)\right) \tag{3.44}$$

where $\hat{\boldsymbol{\mu}}_k$, $\hat{\boldsymbol{\Sigma}}_k$ are the transformed mean and covariance for Gaussian component $k$, the constant $K$ depends only on the transition probabilities, and $K_k$ is the normalization constant with Gaussian component $k$. Note that the state $j$ required for the posterior $\gamma_t(j,k)$ is given by the state alignment of the adaptation data.

The transformed mean $\hat{\boldsymbol{\mu}}_k$ and covariance $\hat{\boldsymbol{\Sigma}}_k$ of Gaussian $k$ are given by

$$\hat{\boldsymbol{\mu}}_k = \mathcal{A}'\boldsymbol{\mu}_k + \boldsymbol{b}' \quad \text{and} \quad \hat{\boldsymbol{\Sigma}}_k = \mathcal{A}'\boldsymbol{\Sigma}_k\mathcal{A}'^T \quad . \tag{3.45}$$

The substitution of Eq. 3.45 in Eq. 3.44 and re-arrangement of the likelihood equation as

$$\begin{aligned}\mathcal{Q}(\Theta|\hat{\Theta}) = K - \frac{1}{2}\sum_k \sum_t \gamma_t(j,k)\big(K_k + \log(|\boldsymbol{\Sigma}_k|) - \log(|\mathcal{A}|^2) \\ + (\hat{\mathbf{o}}_t - \boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}_k^{-1}(\hat{\mathbf{o}} - \boldsymbol{\mu}_k)\big)\end{aligned} \tag{3.46}$$

shows that the transformation can be applied to the features instead of the parameters

$$\hat{\mathbf{o}}_t = \mathcal{A}'^{-1}\mathbf{o}_t + \mathcal{A}'^{-1}\boldsymbol{b}' = \mathcal{A}\mathbf{o}_t + \boldsymbol{b} = \boldsymbol{W}\zeta_t \tag{3.47}$$

where $\boldsymbol{W}$ is the extended transform $\begin{bmatrix}\boldsymbol{b}^T & \mathcal{A}^T\end{bmatrix}^T$, $\zeta_t$ is the extended observation vector $\begin{bmatrix}1 & \mathbf{o}_t^T\end{bmatrix}^T$, $\mathcal{A}' = \mathcal{A}^{-1}$ and $\boldsymbol{b} = \mathcal{A}\boldsymbol{b}'$. An iterative solution scheme for $\boldsymbol{W}$ is given in [Gale 98].

## 3.8 Language Modeling

The acoustic model is used to map the acoustic observations to phonemes. However, identifying the correct word or word sequence solely based on the acoustics is almost impossible even for small lexicons. The English language, among numerous others, feature *homophones*, *i.e.*, words of different meaning but same pronunciation. For example, *their–they're–there* or *see–sea–c* can only be distinguished if the context is considered. Similarly, phrases may sound alike, but consist of different words. An exaggerated example is given in this chapter's preamble: "It's hard to wreck a nice

beach" may very much sound like "it's hard to recognize speech," given a sloppy pronunciation.

The key to sort out these ambiguities is to introduce a language model that limits the speech recognition process to likely (or possible) word sequences. Depending on the use case of the recognition system, this is facilitated either by a set of rules, *e.g.*, a fixed set of sentences, standard or probabilistic context free grammars, or by means of statistics that give a notion of how likely a certain word sequence is (for an extensive overview, see [Huan 01, Schu 95]). These *statistical n-gram models* have been applied with great success to spontaneous speech, and are also used for the experiments in this thesis.

### 3.8.1 Statistical n-gram Models

In general, the n-gram probability of a word sequence $\mathcal{W} = \{w_1, w_2, \ldots, w_L\}$ is expressed in terms of the conditional probabilities

$$P(\mathcal{W}) = P(w_1) \cdot P(w_2|w_1) \cdot P(w_3|w_1 w_2) \cdots P(w_L|w_1 \cdots w_{L-1}) \quad , \tag{3.48}$$

where $P(w_1)$ is called the *uni-gram* probability, $P(w_2|w_1)$ the *bi-gram* probability, and so forth. The individual n-gram probabilities are typically learned from a large corpus of documents or speech transcripts based on their relative frequencies. This raises a few concerns: the larger $n$, the more different n-grams are possible, that is the number of uni-grams to the power of $n$.[3] In consequence, the relative frequencies of higher order n-grams may be unreliable as they tend to be specific for certain domains which might be an unwanted artifact: A recent study on English and Chinese web data shed some light on how many unique n-grams to approximately expect with increasing $n$. The strongest increase is from uni- to bi-gram ($\times$6.5 for English, $\times$5.2 for Chinese) and bi- to tri-grams ($\times$4 for English, $\times$5 for Chinese), featuring a highest number of about 14 000 000 000 unique n-grams for $n = 6$ for English [Yang 07]. Thus, most large vocabulary speech recognition systems limit the context size of their statistical language model to $n = 2$ or $n = 3$ to avoid unreliable relative frequency counts.

The probability of a word sequence given such a bi-gram language model is

$$P(\mathcal{W}) = P(w_1) \cdot P(w_2|w_1) \cdot P(w_3|w_2) \cdots P(w_L|w_{L-1}) \quad . \tag{3.49}$$

Tri-gram models consider two preceding words, respectively. The n-gram probabilities can be interpreted as a Markov chain with a memory length of $(n-1)$, and is implemented as a WFSA in the KALDI system.

### 3.8.2 Smoothing Techniques for Statistical n-gram Models

One of the key questions when using an n-gram language model is how to handle unseen n-grams, *i.e.*, an n-gram for which the relative frequency count is zero. [Chen 96] give an extensive overview of the most common techniques. The following paragraphs briefly introduce the methods which will be used for the experiments in this thesis.

[3] A theoretic upper boundary. Word repetitions are of course rather unrealistic for larger $n$.

**Add-One Smoothing** A straight forward solution to avoid zero probability for an unseen n-gram is to add one to each frequency count (*e.g.*, [Jeff 48]). The smoothed bi-gram probability is then

$$P(w_i|w_{i-1}) = \frac{1 + C(w_{i-1}w_i)}{\sum_i 1 + C(w_{i-1}w_i)} = \frac{1 + C(w_{i-1}w_i)}{|V| + \sum_i C(w_{i-1}w_i)} \tag{3.50}$$

where $C(w_{i-1}w_i)$ is the count of the bi-gram $w_{i-1}w_i$ and $|V|$ is the size of the vocabulary.

**Back-Off Smoothing** Instead of adding one, a missing n-gram can be approximated by a more frequent lower order n-gram. In case of a bi-gram:

$$P(w_i|w_{i-1}) = \begin{cases} \eta(w_i|w_{i-1}) & \text{if } C(w_{i-1}w_i) > 0 \\ \upsilon(w_{i-1}) \cdot P(w_i) & \text{else} \end{cases} \tag{3.51}$$

where $\eta(\cdot)$ is actual (discounted) n-gram probability and $\upsilon(\cdot)$ is the back-off weight chosen to make the conditional distribution sum up to one.

**Kneser-Ney Discounting** Kneser *et al.* [Knes 95] implement the above back-off model as

$$P_{KN}(w_i|w_{i-1}) = \begin{cases} \frac{\max\{C(w_{i-1}w_i) - D, 0\}}{\sum C(w_{i-1})} & \text{if } c(w_{i-1}) > 0 \\ \upsilon(w_{i-1}) P_{KN}(w_i) & \text{otherwise} \end{cases} \tag{3.52}$$

where $D$ is a positive discount,

$$P_{KN}(w_i) = \frac{C(\bullet w_i)}{\sum_i C(\bullet w_i)} \tag{3.53}$$

is the "continuation probability," *i.e.*, that the unseen n-gram is actually an unseen continuation, and $C(\bullet w_i)$ is the number of unique words preceding $w_i$. $\upsilon(w_{i-1})$ is set to make the distribution sum up to one.

## 3.9 The KALDI Speech Recognition Toolkit

The KALDI speech recognition toolkit[4] [Pove 11b] is an open-source project licensed under the Apache 2.0 License[5]. KALDI is implemented in C++ and compiles on Windows and Unix-based operating systems. Beside the specific algorithms and utilities for speech recognition, the toolkit maintains an extensive matrix (and vector) package that provides common linear algebra routines such as arithmetics and factorizations.

One of the key aspects of KALDI is the strict separation of the acoustic model from the language model. WFSTs are used to model language model, lexicon including pronunciation alternatives, and the phonetic states in context. The acoustic model stores solely the statistical parameters of the emission probabilities which are indexed by the state numbers of the WFST. This strict decoupling of the emission probabilities

[4] `http://sourceforge.net/projects/kaldi/`

[5] `http://www.apache.org/licenses/LICENSE-2.0`

and the HMM (or WFST) framework makes it very simple to change the underlying AM without touching the remaining system setup.

The KALDI framework implements most of the current state-of-the-art. As acoustic front-end, MFCC and perceptual linear prediction (PLP) features can be computed. The deltas (up to third order) are computed on the fly to save disk space. For the acoustic model, there is a choice of standard continuous, semi-continuous, multiple-codebook continuous, and Gaussian subspace models, which can be trained using maximum likelihood or discriminative objective functions. Several feature and model transformations for channel and speaker robustness are available, including LDA, VTLN and MLLR, to name only a few. Language modeling is not directly part of KALDI, however the toolkit provides routines to read the common LM file formats such as ARPA[6].

KALDI utilizes and integrates a number of other toolkits and libraries. The linear algebra parts are supported by the BLAS[7] and LAPACK[8] libraries. The OpenFST toolkit[9] [Alla 07] is used to construct, compose, minimize and determinize the weighted transducers. For the experiments in this thesis, the SRILM[10] [Stol 11] is used to estimate the n-gram language models that are required to compose the final decoding WFST.

KALDI focuses on open and reproducible research. The developer community maintains an archive of recipes for standard speech recognition data sets.

## 3.10 Speech Recognition Performance Measures

The performance of a speech recognition system is typically measured by comparing the recognized word sequence (hypothesis) with the reference word sequence that was obtained by manual transcription. The word error rate (WER) is derived from the Levenshtein *(edit)* distance [Leve 66]. A dynamic programming algorithm determines the optimal number of string insertions ($n_i$), deletions ($n_d$), and substitutions ($n_s$) that are necessary to transform the hypothesis into the reference. The word error rate

$$\mathrm{WER} = \frac{n_i + n_d + n_s}{N} \cdot 100\% \quad , \tag{3.54}$$

where $N$ is the number of words in the reference, considers insertions, deletions and substitutions equally. The WER has a minimum of 0% in case of no errors, and is greater than 100% if the hypothesis is longer than the reference, but does not contain any correct word ($n_s = N \wedge n_d > 0$). Sometimes, the reciprocal word accuracy

$$\mathrm{WA} = 100\% - \mathrm{WER} = \left(1 - \frac{n_i + n_d + n_s}{N}\right) \cdot 100\% \tag{3.55}$$

is used in conjunction with the word correctness

$$\mathrm{WC} = \left(1 - \frac{n_i + n_s}{N}\right) \cdot 100\% \tag{3.56}$$

[6] A description of the ARPA LM format can be for example found in the manual of the SRILM

[7] Basic Linear Algebra Subroutines, `http://netlib.org/blas/`

[8] Linear Algebra PACKage, `http://netlib.org/lapack/`

[9] `http://www.openfst.org/`

[10] `http://www.speech.sri.com/projects/srilm/`

to quantify the impact of the deletions on the performance. Analogously to the WER, the WA has a maximum of 100% in case of no errors, and can be negative depending on the input. The WC, however, has a minimum of 0% as the number of insertions are not accounted for. Considering the WC instead of the WER or WA may reasonable for certain tasks such as key word spotting or dialog systems, where additional words in the hypothesis do only little harm. The general performance should however be given in WER or WA as the insertions make a big difference in common setups.

The $n_i$, $n_s$ and $n_d$ are furthermore good indicators if the decoding parameters were properly set. An high number of deletions or insertions suggests that the word insertion penalty was set too low or too high. A high number of substitutions is likely due to an unfitting language model, or a wrong acoustic scale.

## 3.11 Experiments and Results

The experiments presented in this section are split in two main parts. First, the semi-continuous acoustic models and the related smoothing techniques are evaluated for the WSJ data set, using the continuous and SGMM system configurations as found in the KALDI recipes `wsj/s3` and extending prior experiments using different data sets [Ried 12]. These prior experiments already showed, on a much smaller data set, that the two-level tree based semi-continuous system is sensitive to the choice of parameters. The three parameters, number of codebooks, number of Gaussians, and number of tree leaves (*i.e.*, the actually modeled HMM states in context), are closely geared, both in terms of run-time and recognition performance. To get a feeling for their effect, the parameters are initially explored separately before extending the experiments to the smoothing techniques.

In the second part, the system configurations found on the WSJ data set are used to train the LMELectures recognition systems which produce the automatic transcriptions that will be used in the following chapters.

The model and decoding parameters are estimated on the development sets (*dev93* for WSJ, *devel* for LMELectures) and then applied to the test sets (*eval92, eval93* for WSJ, *test* for LMELectures).

### 3.11.1 Features and Transformations

The primary features used in the following experiments are the MFCC coefficients $c_0, \ldots, c_{12}$ (*cf.* Sec. 3.2.2) and the respective first and second order derivatives (*cf.* Sec. 3.2.3), which are used for the initial system. For the advanced systems, the features are derived from the static MFCC coefficients using LDA and MLLT. For each frame, the neighboring nine consecutive MFCC vectors are concatenated and the dimension reduced from 117 to 40 using LDA. The class labels required for LDA are obtained from existing Viterbi state alignments, associating each tree leaf with a distinct class. The LDA transformation matrix is further (left-) multiplied by the MLLT transformation matrix to obtain the final feature vector.

Both LDA and MLLT transformations are estimated on the continuous tri-phone systems, and re-used for the semi-continuous and SGMM systems to ensure a fair comparison of the models. Furthermore, the update formulas for the full-covariance

Gaussian are components are less straight-forward and require a statistical gradient optimization which is not yet implemented for the semi-continuous models.

### 3.11.2 Language Models and Pronunciation Dictionary

The WSJ corpus provides a tri-gram language model consisting of 19 982 uni-grams, 3 518 595 bi- and 3 153 527 tri-grams. For faster decoding, a pruned model with 758 936 bi- and 709 089 tri-grams is used, as found in the KALDI `wsj/s3` recipes.

For the experiments on the LMELectures data, three different tri-gram language models are derived from a large tri-gram language model for the lecture domain covering about 60 000 words and about 11 000 000 bi- and trigrams each [Stol 08], which is too large for the direct modeling within the WFST framework. In a first step, the `srilm` toolkit [Stol 11] is used to limit the model to the 5 383 words of the LMELectures lexicon, reducing the number of bi- and tri-grams to 2 809 090 and 6 410 828, respectively, which is denoted as *small*. This still rather large model is further pruned by removing n-gram probabilities if their removal causes the perplexity of the model to increase by less than a certain threshold relative [Stol 98]. For the *p-tri* model, tri-grams are removed with a threshold of $5 \cdot 10^{-8}$, resulting in 184 685 tri-grams. For the *p-all* model, both bi- and tri-grams were pruned using a threshold of $10^{-9}$, resulting in 1 752 182 bi- and 3 617 858 tri-grams. The intent is to analyze whether a more detailed modeling of bi-grams is to be favored over a more balanced pruning of both bi- and tri-grams. The back-off probabilities for missing n-grams are computed using the Kneser-Ney Smoothing.

The CMU Pronunciation Dictionary v. 0.7a[11] was used to find the pronunciations for each word in the language model. Missing pronunciations were generated using the RWTH Aachen `g2p` graphem-to-phoneme tool [Bisa 08] which was trained on the CMU dictionary.

### 3.11.3 Experiments on WSJ

#### The KALDI Baseline Systems

The semi-continuous systems are compared to a continuous and SGMM system taken from the KALDI recipes (`wsj/s3`). Tab. 3.2 shows the incremental system build process along with the data, model parameters and adaptations that are applied. Note that, for clarity, the systems are named as in the recipes. Each training run starts from either linear time alignments or a forced alignments computed using a preceding recognizer, and re-computes the forced alignments every ten iterations, using a total of 35 Viterbi training iterations.

The flat-start mono-phone system *mono0a* uses the 2 000 shortest utterances of *si84* and an initial linear time alignment of the training utterances. The features used are the first 13 MFCC, along with the first and second order derivatives. The rather high number of tree leaves is due to the distinct modeling of singleton, beginning and end phones, as well as the silence models. The *tri1* system uses tri-phones and is based on 50% of *si84*, using forced alignments computed by *mono0a* and the same

[11] `http://www.speech.cs.cmu.edu/cgi-bin/cmudict`

| *system* | *training data* | *# leaves* | *# Gaussians* | *comments* |
|---|---|---|---|---|
| mono0a | si84 (2 000 shortest utt.) | 146 | 1 000 | mono-phones |
| tri1 | si84 (50%) | 2 000 | 10 000 | tri-phones |
| tri2b | si84 | 2 500 | 15 000 | LDA, MLLT |
| tri4b | si284 | 4 200 | 40 000 | LDA, MLLT |
| sgmm4c | si284 | 5 500 | (25 000) | LDA, MLLT |

Table 3.2: Incremental system build process from the KALDI WSJ recipes (`wsj/s3`). From a flat start, an increasing amount of data, model parameters and adaptation techniques are applied. Only the flat start system *mono0a* is initialized using linear time alignments, the remaining systems are trained using forced alignments obtained from the preceding system. The SGMM system uses a codebook of 600 full covariance Gaussians to span the 25 000 substates.

| *system* | *dev03* | *eval92* | *eval93* |
|---|---|---|---|
| mono0a | 34.32 | 25.78 | 30.28 |
| tri1 | 20.18 | 13.22 | 18.10 |
| tri2b | 17.44 | 11.93 | 15.60 |
| tri4b | 14.57 | 10.03 | 13.08 |
| sgmm4c | **11.85** | **8.15** | **11.48** |

Table 3.3: WER of the KALDI baseline systems for the development and test sets. Parameters are optimized on the development set.

features. Using *tri1* to force-align the complete *si84* set, the *tri2b* system is trained using both LDA and MLLT transformations, and increasing the number of leaves and total Gaussian components to 2 500 and 15 000, respectively. The *tri2b* system is used to compute the forced alignments of the large training set *si284*, which will be used for the final recognizer training.

The *tri4b* system extends *tri2b* by using all training data, 4 200 tree leaves and a total of 40 000 Gaussian components. The *sgmm4c* system uses a codebook of 600 Gaussian components with full covariance which is obtained by merging the individual Gaussian components of the *tri4b* system. Due to the design of the acoustic model, the *sgmm4c* system can be set up with more tree leaves and substates as the parameters are still tied to the codebook. The KALDI community found 5 500 tree leaves and 25 000 substates (tied to the 600 Gaussian codebook) a useful setting.

Tab. 3.3 shows the WER for the above introduced systems. Intuitively, the increasing amount of data and model parameters result in a better performance on both development and test sets. The largest decrease in WER beside using tri- instead of mono-phones is found for using the SGMM acoustic model, resulting in a WER of 11.85% on the *dev93* development set, and 8.15% and 11.48% for the *eval92* and *eval93* test sets, respectively.

| # *codebooks* | # *Gaussians* | # *leaves* | *dev93* | *eval92* | *eval93* |
|---|---|---|---|---|---|
| 200 | 3 000 | 1 000 | 16.55 | 11.27 | 16.10 |
| 200 | 3 000 | 2 000 | 16.08 | 11.20 | 15.43 |
| 200 | 3 000 | 3 000 | 15.90 | 11.34 | 15.34 |
| 200 | 3 000 | 4 000 | **15.59** | 11.16 | 14.94 |
| 200 | 3 000 | 5 000 | 15.65 | **10.67** | **14.44** |

Table 3.4: WER in percent for two-level tree based systems using varying numbers of tree leaves and a constant number of codebooks and total Gaussian components.

### Semi-continuous Acoustic Models

In [Ried 12], experiments on a smaller data set suggested that using full instead of diagonal covariance Gaussian components is beneficial, similar to SGMM, and that traditional semi-continuous models are inferior to two-level tree based multi-codebook models both in terms of run-time and recognition performance. For the two-level tree based model, a setting of 3 072 Gaussians distributed on 208 codebooks together with 2 500 tree leaves showed a slightly better performance than a continuous system with about 9 000 (diagonal) Gaussians — for a small data set both in terms of words and training data. These results could however not be confirmed for the significantly larger WSJ corpus, which will be corrected in the following paragraphs. The focus is on the two-level tree based multi-codebook system (*2lvl*) using full covariance Gaussian components. The use of diagonal covariances and the traditional semi-continuous system (*semi*) is addressed later.

**Number of Tree Leaves** The initial parameter set for the following experiments is a similar configuration as in [Ried 12], using 3 000 full covariance Gaussian components distributed over 200 codebooks (using $q = 1$, *cf.* Eq. 3.32) for the *2lvl* system. In a first step, the effect of the number of leaves of the acoustic-phonetic tree is analyzed with respect to the system WER. The large phonetic inventory and large set of distinct tri-phones provides sufficient data for a detailed phonetic modeling. Increasing the number of leaves, *i.e.*, the number of actual states in context that require an acoustic model, slightly increases the number of actual parameters in case of a semi-continuous system. Tab. 3.4 shows the WER using between 1 000 and 5 000 tree leaves. Clearly, a larger number of leaves, and thus a more detailed phonetic model, improves the recognition performance. However, the modeling capacity seems to be limited by the number of Gaussian components: using 5 000 instead of 4 000 leaves in combination with only 3 000 Gaussians leads to only a minor improvement.

**Number of Gaussian Components** This observation leads to the question how the number of Gaussian components is geared with the number of leaves. Tab. 3.5 shows the WER for the *2lvl* system when using a variable number of Gaussians but keeping the number of codebooks and tree leaves constant. Similar to the prior experiment, an increasing number of Gaussians leads to a better recognition performance, with the relative improvement diminishing if the number of Gaussians approaches the

| # codebooks | # Gaussians | # leaves | dev93 | eval92 | eval93 |
|---|---|---|---|---|---|
| 200 | 1 000 | 5 000 | 19.14 | 13.17 | 19.08 |
| 200 | 2 000 | 5 000 | 16.58 | 11.34 | 15.92 |
| 200 | 3 000 | 5 000 | 15.65 | 10.67 | 14.44 |
| 200 | 4 000 | 5 000 | 15.11 | 10.53 | 13.98 |
| 200 | 5 000 | 5 000 | **14.95** | **10.28** | **13.25** |

Table 3.5: WER in percent for two-level tree based systems using varying numbers of total Gaussian components and a constant number of codebooks and tree leaves.

| # codebooks | # Gaussians | # leaves | dev93 | eval92 | eval93 |
|---|---|---|---|---|---|
| 200 | 5 000 | 5 000 | 14.95 | **10.28** | **13.25** |
| 600 | 5 000 | 5 000 | **14.80** | 10.31 | 14.24 |
| 1 000 | 5 000 | 5 000 | 15.05 | 10.30 | 13.75 |

Table 3.6: WER in percent for two-level tree based systems using varying numbers of codebooks and a constant number of total Gaussian components and tree leaves.

number of tree leaves. At this point is is important to note that a similar number of Gaussians and leaves does not imply that there is about a single Gaussian component for each leaf. The average codebook size is determined by the number of Gaussians, codebooks, and the state occupancies from the initial training data alignments, while each tree leaf is associated with a set of weights matching the size of the respective codebook.

**Number of Codebooks** The remaining parameter, the number of codebooks the Gaussian components are distributed to, controls both the computational and modeling complexity. A large number of codebooks leads to a rather small average size while a small number of codebooks (or a single codebook in case of *semi*) leads to a large number of components per codebook. While a large codebook may in general allow more flexible acoustic modeling, the computational complexity increases – independent of the fact that the component weights may emphasize only a few individual components. Tab. 3.6 shows the WER of systems with a variable number of codebooks while keeping the number of tree leaves and Gaussian components constant. The results show only minor and notably inconsistent variation for all data sets, suggesting that the number of codebooks plays a less important role than the number of Gaussian components or tree leaves. This backs the original motivation of multiple codebook semi-continuous models that is to cluster these Gaussian components that would be active in a certain set of states, in order both speed up the computations and possibly make better use of modeling the Gaussian components as they are not distorted by concurrent states. This implies on the other hand that the number of codebooks can be increased to speed up the computations without necessarily affecting the recognition performance.

| *system* | # *codebooks* | # *Gaussians* | # *leaves* | *dev93* | *eval92* | *eval93* |
|---|---|---|---|---|---|---|
| 2lvl/full | 200 | 5 000 | 5 000 | **14.95** | **10.28** | **13.25** |
| 2lvl/diag | 200 | 5 000 | 5 000 | 17.61 | 12.33 | 17.08 |
| 2lvl/diag | 200 | 10 000 | 5 000 | 15.79 | 11.02 | 14.91 |
| semi/full | 1 | 600 | 5 000 | 26.28 | 19.16 | 26.89 |

Table 3.7: WER in percent for semi-continuous systems using full and diagonal covariance matrices; *semi* represents a traditional semi-continuous system, *2lvl* a two-level tree based one. The *semi* model uses the same initial codebook as *sgmm4c*.

| # *codebooks* | # *Gaussians* | # *leaves* | $q$ | *dev93* | *eval92* | *eval93* |
|---|---|---|---|---|---|---|
| 1 000 | 10 000 | 7 000 | 1.0 | 14.40 | 9.59 | 12.99 |
| 1 000 | 10 000 | 7 000 | 0.2 | 14.15 | 10.19 | 12.62 |

Table 3.8: WER in percent using the final parameter set for the *2lvl* system using different $q$ values to distribute the Gaussian components to the codebooks.

**Model Topologies** The above experiments focused on extending the prior experiments on two-level tree based systems in [Ried 12]. To complete that framework of experiments, Tab. 3.7 shows results for using diagonal instead of full covariances for *2lvl* based systems, as well as the traditional semi-continuous system *semi* with only a single codebook. The number of codebooks, Gaussian components and tree leaves follows the above experiment. The *semi* system uses the same (initial) codebook as the *sgmm4c* baseline system. Rows two to four of Tab. 3.7 compare the *2lvl* system with respect to full and diagonal covariances. The increase of WER for diagonal covariance when using the same amount of Gaussians can to some extent be compensated by increasing the number of Gaussian components. However, the performance using diagonal covariances is still clearly inferior despite twice the number of components. Increasing the number of components further may eventually lead to a similar performance but would negate the lower computational complexity of diagonal covariances. The traditional semi-continuous system performance is far behind the performance of the *2lvl* systems, confirming the observations in [Ried 12]. Although substantially increasing the number of densities may eventually result in a competitive result, the computational complexity would be unacceptable: for each feature observation, the complete codebook would have to be evaluated in both training and test. Paired with the prior observation that increasing the number of codebooks, and thus reducing the computational complexity, does not necessarily deteriorate the WER, the traditional semi-continuous model seems to be obsolete and are not further considered in the following experiments. A practical interpretation of multi-codebook semi-continuous systems is that only these Gaussian components are actually evaluated which are relevant to a certain state, which is typically only a small percentage in a large codebook.

| $q$ | *min* | *max* | *mean* | *std. dev.* |
|---|---|---|---|---|
| 1.0 | 3 | 792 | 10 | 31.97 |
| 0.2 | 7 | 22 | 10 | 1.19 |

Table 3.9: Minimum, maximum, mean and standard deviation of the *2lvl* system codebook sizes using different values of $q$.

**Distribution of Gaussian Components** The *tri4b* and *sgmm4c* systems are configured with a far higher number of Gaussians and subspaces, respectively, than the *2lvl* systems in the initial experiments. However, simply increasing the number of Gaussian components leads to larger codebooks which leads to larger computational complexity. For $q = 1$ this also implies that the absolute variance of individual sizes of the codebooks increases due to the linear relation between codebook occupancy and number of attributed Gaussian components, which can only partially be eased by increasing the number of codebooks. A more elegant way to avoid inadequately sized codebooks is to set $q < 1$ to enforce a rather homogeneous codebook size. The final parameters of the *2lvl* system are found based on the findings in the previous paragraphs: A larger number of tree leaves should increase the phonetic modeling capabilities. As, in contrast to *tri4b* and *sgmm4c*, the number of parameters only slightly increase the overall number of parameters, the number of tree leaves is set to 7 000. The sufficient modeling of the increased number of leaves requires a higher number of Gaussian components. Here the number of full covariance Gaussians is set to 10 000, following the observation that Gaussians components with full covariance show a higher modeling capability than components with diagonal covariance. To reduce the computational complexity of the system in training and test, and to avoid over-sized codebooks, the Gaussians are distributed to the 1 000 codebooks using $q = 0.2$ (*cf.* Eq. 3.32). This follows the rule used for the continuous systems to distribute the Gaussian components over the states; further values of $q$ are not considered as the idea is to show the advantage of homogeneous codebook sizes. A reference system with $q = 1$ is trained for comparison. Tab. 3.9 shows the minimum, maximum, mean and standard deviation of the resulting codebook sizes. For $q = 1$, the largest codebook contains 792 Gaussian components, *i.e.*, the most frequent states are represented by about 8% of the total Gaussians. This leads to a definite performance bottleneck as this codebook is very likely to be evaluated. For $q = 0.2$ on the other hand, the largest codebook contains 22 Gaussian components which may still be sufficient to model these frequent states using the component weights.

Tab. 3.8 shows the results in comparison for $q = \{1.0, 0.2\}$. Although using $q = 0.2$ leads to a slightly better performance on the development set, the results on the test sets are inconclusive. The fact that the training is about six times faster for $q = 0.2$ due to the absence of individual oversized codebooks suggests to favor homogeneous codebook sizes.

**Smoothing Techniques** Tab. 3.10 shows the WER for *2lvl* systems trained with varying smoothing coefficients $\rho$ (inter-iteration) and $\tau$ (intra-iteration). A quick glance at the table reveals that both smoothing types have only little effect on the

| $\rho$ | $\tau$ | *dev93* | *eval92* | *eval93* |
|---|---|---|---|---|
| 0.0 | 0.0 | 14.15 | 10.19 | 12.62 |
| 0.0 | 0.3 | 14.14 | 10.17 | 12.62 |
| 0.0 | 0.5 | 14.08 | 9.87 | 12.50 |
| 0.0 | 0.7 | 14.09 | 10.19 | 12.65 |
| 0.0 | 50 | 14.12 | 9.96 | 12.59 |
| 0.0 | 250 | 14.27 | 9.91 | 12.56 |
| 0.0 | 500 | 14.26 | 9.96 | 12.65 |
| **0.3** | **0.0** | **14.00** | **9.99** | **12.50** |
| 0.3 | 0.3 | 14.05 | 10.01 | 12.41 |
| 0.3 | 0.5 | 14.08 | 10.05 | 12.47 |
| 0.3 | 0.7 | 14.03 | 10.03 | 12.50 |
| 0.3 | 50 | 14.23 | 10.03 | 12.62 |
| 0.3 | 250 | 14.33 | 9.89 | 12.47 |
| 0.3 | 500 | 14.36 | 9.96 | 12.59 |
| 0.5 | 0.0 | 14.20 | 9.98 | 12.62 |
| 0.5 | 0.3 | 14.16 | 9.87 | 12.47 |
| 0.5 | 0.5 | 14.16 | 9.98 | 12.59 |
| 0.5 | 0.7 | 14.14 | 9.98 | 12.59 |
| 0.5 | 50 | 14.20 | 10.12 | 12.62 |
| 0.5 | 250 | 14.38 | 9.96 | 12.62 |
| 0.5 | 500 | 14.34 | 9.99 | 12.73 |
| 0.7 | 0.0 | 14.33 | 10.05 | 12.50 |
| 0.7 | 0.3 | 14.31 | 9.99 | 12.53 |
| 0.7 | 0.5 | 14.31 | 10.03 | 12.50 |
| 0.7 | 0.7 | 14.20 | 9.91 | 12.59 |
| 0.7 | 50 | 14.40 | 9.89 | 12.35 |
| *0.7* | *250* | *14.50* | *9.75* | *12.27* |
| 0.7 | 500 | 14.46 | 9.78 | 12.35 |

Table 3.10: WER in percent for the *2lvl* system using 7 000 tree leaves and 10 000 Gaussians distributed over 1 000 codebooks; $\rho$ controls the *inter*- and $\tau$ controls the *intra*-iteration smoothing. The row typeset in bold face indicates the best system based on the *dev93* result, the row typeset in italic indicates the best system based on the *eval92* and *eval93* sets.

| *system* | # *codebooks* | # *Gaussians* | # *leaves* | *dev93* | *eval92* | *eval93* |
|---|---|---|---|---|---|---|
| tri4b | — | 40 000 | 4 200 | 14.57 | 10.03 | 13.08 |
| 2lvl/a | 1 000 | 10 000 | 7 000 | 14.15 | 10.19 | 12.50 |
| 2lvl/b | 1 000 | 10 000 | 7 000 | 14.00 | 9.99 | 12.50 |
| 2lvl/c | 1 000 | 10 000 | 7 000 | 14.50 | 9.75 | 12.27 |
| sgmm4c | 1 | 600 (25 000) | 5 500 | 11.85 | 8.15 | 11.48 |

Table 3.11: WER in percent for the different systems in comparison. For *2lvl*, the Gaussians are distributed using $p = 0.2$. The *2lvl* systems use different smoothing parameters: *a)* $\rho = \tau = 0$, *b)* $\rho = 0.3, \tau = 0$, *c)* $\rho = 0.7, \tau = 250$.

WER for both development and test sets, a result also found in [Ried 12]. Considering the results on the *dev93* set, applying only inter-iteration smoothing ($\rho > 0, \tau = 0$) yields the best WER of 14.00% for $\rho = 0.3$. This observation fits the intuition that the inter-iteration smoothing helps to prevent the Gaussian weights from over-fitting and matches findings in [Ried 12] on a different data set. For the intra-iteration smoothing, the interpolation weight $\tau$ can be set to values larger than one to increase the amount of smoothing applied to states with low occupancies. However, this possibly results in a final interpolation weight larger than one if a state has less occupancy than $\tau$. Considering the first block of Tab. 3.10 where $\rho = 0$, $\tau = 0.5$ leads to the best improvements both on the development and test sets. For $\tau > 1$, the values are oriented on the actual occupancies: $\tau = 50$ is just below the minimum occupancy of all states, ensuring an interpolation weight below one, about 1% of all states have less occupancy than 250, and about 5% of all states have less occupancy than 500.[12] Independently of $\rho$, $\tau > 1$ leads to an increased WER. The only exception is $\rho = 0.7$ and $\tau = 250$, which leads to the best WER for both test sets. The rather inconclusive results for the intra-iteration smoothing suggest to restrict its application only in presence of limited data, as already noted in [Ried 12]. The state occupancies for the *si284* training set seem to be sufficient to compute reliable parameter estimates.

**Results in Comparison** Tab. 3.11 summarizes the results on the WSJ development and test sets for the continuous (*tri4b*), two-level tree based semi-continuous (*2lvl*) and subspace Gaussian mixture (*sgmm4c*) system. Without smoothing, the *2lvl* system shows an about 3% relative improvement over *tri4b* while using significantly less Gaussian components. With smoothing, the *2lvl* systems achieve a relative improvement between 4% and 6% depending on configuration and evaluation set. The SGMM system adds another 14% relative improvement, showing the competitive WERs of 11.85% on the development, and 8.15% and 11.48% on the *eval92* and *eval93* test sets. This confirms the observation from earlier experiments on different data sets [Ried 12] that a two-level tree based semi-continuous system can outperform a regular continuous system while having significantly less Gaussian components.

[12] In case of Viterbi training, the state occupancy equals the number of observation frames associated with that state.

| *system* | *# codebooks* | *# Gaussians* | *# leaves* | *LM* | *devel* | *test* |
|---|---|---|---|---|---|---|
| tri4b | — | 40 000 | 4 200 | small | 11.55 | 13.57 |
| | | | | p-all | 11.52 | 13.19 |
| | | | | p-tri | 11.81 | 13.87 |
| 2lvl/a | 1 000 | 10 000 | 7 000 | small | 11.34 | 13.65 |
| | | | | p-all | 11.21 | 12.75 |
| | | | | p-tri | 11.73 | 13.42 |
| 2lvl/b | 1 000 | 10 000 | 7 000 | small | 11.18 | 13.52 |
| | | | | p-all | 11.14 | 12.80 |
| | | | | p-tri | 11.65 | 13.38 |
| 2lvl/c | 1 000 | 10 000 | 7 000 | small | 11.57 | 13.75 |
| | | | | p-all | 11.40 | 12.93 |
| | | | | p-tri | 11.82 | 13.54 |
| sgmm4b | 1 | 600 (25 000) | 5 500 | small | 10.02 | 11.70 |
| | | | | p-all | **9.78** | **11.03** |
| | | | | p-tri | 10.26 | 11.61 |

Table 3.12: WER in percent for the development and test data of the LMELectures data set using different language models. The *2lvl* systems use different smoothing parameters found on the WSJ set: *a)* $\rho = \tau = 0$, *b)* $\rho = 0.3, \tau = 0$, *c)* $\rho = 0.7, \tau = 250$.

### 3.11.4 Experiments on LMELectures

The system configurations found to be optimal on the WSJ data sets are reused for the training and evaluation on the LMELectures data set. The initial alignments required for the training are obtained using the *tri4b* system where the lexicon was replaced by the lexicon of the LMELectures.

Although the amount of training data is a little less then half of WSJ, the number of parameters is not reduced: with only a single speaker in LMELectures, the data has less variability and should thus be sufficient to reliably estimate the parameters.

Tab. 3.12 shows the WER of the different systems. The decoding parameters are again calibrated on the development set. Overall, the results are similar to the results using the WSJ data set: the SGMM system shows the best performance on development and test set, followed by the two-level tree based semi-continuous and continuous system. An interesting observation is the performance based on the different language models. Surprisingly, the most detailed language model *small* does not result in the best recognition performance. Also, pruning the tri-grams while retaining the all bi-grams (*p-tri*) leads to an even worse WER. The best result is obtained when using the average sized *p-all* language model where both bi- and tri-grams are pruned, which can be explained by the fact that the language model is not re-trained specifically for the LMELectures data. The language model used for the experiments was designed for the lecture domain, however, the probabilities of the less likely bi- and tri-grams may be skewed or mismatch the LMELectures domain. Thus pruning away the less likely n-grams reduces recognition errors based on domain

mismatch. Furthermore, the reduced size of *p-all* also reduces the complexity of the decoding WFST and thus the computational load.

## 3.12 Discussion

The experiments emphasize the advantages of two-level tree based multi-codebook semi-continuous models compared to both traditional (single codebook) semi-continuous and continuous systems. The downside of traditional semi-continuous systems with large codebooks is that, both in training and test, all Gaussian components need to be evaluated for every frame, despite the fact that only a few might actually contribute due to the score due to the weights: typically, for each state, a few weights dominate the remaining ones. The two-level tree based model allows to tie these states that concentrate on the same Gaussian components, thus avoiding to evaluate components that do not contribute to the score. Still, the multi-codebook architecture has the benefits of a regular semi-continuous model, *i.e.*, that for related states, the components need to be evaluated only once. Also, the number of tree leaves can be significantly increased, as each leaf only requires a small number of weights depending on the associated codebook. This combines the strengths of a continuous and semi-continuous systems, that is individual Gaussian components for each state but a rather small number of Gaussian components and thus parameters to estimate.

The actual number of parameters is strongly linked to the choice of full or diagonal covariance matrices. The experiments show that modeling full covariance matrices is beneficial, but can to some extent be compensated by increasing the number of Gaussian components. The use of diagonal matrices also simplifies the computation of the likelihoods, thus the decision for full or diagonal covariance matrices should be based on run-time considerations. The actual number of parameters when using diagonal or full covariance matrices should be compared with care. Although the number of parameters is significantly higher for full covariances, the diagonal system requires more Gaussian components and thus the estimation of a significantly larger number of mean values, which are of course critical to the likelihood computation. The point is that when using diagonal instead of full covariances, the same amount of data is used to estimate a larger number of mean vectors and variances.

The inter-iteration smoothing of the parameters leads to a moderate improvement of the recognition scores for $\rho = 0.3$, an observation consistent with experiments in [Ried 12] on a different data set. This backs the assumption that the weights tend to converge faster than the Gaussian means and covariances, resulting in a possibly over-fitted solution, and that slowing the convergence of the weights indeed helps to find a better overall parameter estimate. The intra-iteration smoothing of the sufficient statistics shows inconsistent effects on the WER. While for some cases a moderate smoothing seems to help, choosing a wrong $\tau$ may lead to a higher WER. However, the inconsistent results for the interplay of both inter- and intra-iteration smoothing makes it hard to come up with a definite conclusion on how to apply both smoothing techniques. Taking results in [Ried 12] into account, the intra-iteration smoothing can be beneficial in absence of sufficient training data, and the inter-iteration smoothing helps for certain $\rho$.

The SGMM based system shows a consistently better WER for both the WSJ and LMELectures data sets. The advantage of the SGMM acoustic model is that the more complex model of computing the likelihood of the substates. In contrast to semi-continuous systems, the individual likelihoods are not only determined by individual component weights, but also by a transformation of the mean vector of the underlying component. Although this model requires far more effort for the likelihood computation compared to a continuous system, the increased modeling capabilities of the individual substates (or leaves) positively affects the WER. The best WER of 11.03% on the LMELectures test set is a solid base for the experiments in the following chapter.

# Chapter 4

# Key Phrase Extraction and Ranking

This chapter is about finding phrases in a transcribed spoken document such as an audio or video lecture that best represent the covered topics and facts, the *key phrases*. Clearly, not every phrase may be of equal importance, thus a notion of importance, or *salience*, is required to distinguish between useful and superfluous phrases. As a result, most algorithms consist of two steps. Possible candidates are first identified, and then ranked according to their salience. Parts of this chapter have been published in [Ried 08a, Ried 10, Grop 11].

## 4.1 Introduction

Automatic methods to identify key phrases can be categorized as supervised or unsupervised. Supervised methods need to be trained prior to their use. Given a representative sample, *e.g.*, a set of spoken documents with manually labeled key phrases, a model is learned that makes predictions for each phrase of a new document to be a key phrase. Supervised methods can achieve high accuracy, *e.g.*, for meeting [Liu 11] and lecture data [Chen 10], but are, like all machine learning methods, strongly dependent on in-domain training data. For example, a system trained on a political context might detect phrases such as "party," "referendum" and "decision" — words which are unlikely to appear or to be salient in the context of music. A further drawback is the availability of representative data. While it might be easy to collect data for a certain domain, the training stage requires manual labels that indicate the key phrases. This labeling process is typically done by human annotators, and is thus time-consuming and expensive.

Unsupervised methods are appealing as they do not require such annotated in-domain training data. Instead, they try to identify key phrases by their inherent attributes. Spoken language, in contrast to written language or broadcast news[1], as observed in lectures or meetings has interesting properties regarding key phrases. In written language, it is considered good phrasing to use synonyms, references and paraphrases. In spontaneous speech, important things are often literally repeated — a phenomenon especially apparent for lecture and meetings: In teaching, new, and thus important things are repeatedly called by the same name so that students are

[1]Broadcast news are essentially read from a teleprompter, thus their structure is more similar to written language than spontaneous speech.

less likely to get confused. Furthermore, a consistent nomenclature helps students to quickly familiarize with a new context. In conversations with two or more speakers, interlocutors mutually agree on certain phraseology to signalize a common ground and to make sure the other party understands what was meant. This phenomenon is known from psycholinguistics as *lexical entrainment* [Bren 96].

The methods presented in this thesis are selected to work independently of the domain of the spoken document. The interesting property of salient phrases along with the independence of in-domain training data, make unsupervised key phrase extraction the method of choice. The following sections describe the necessary preprocessing, and put the candidate phrase selection and subsequent ranking methods in context to related work. An evaluation with respect to human annotated key phrases shows the validity of the presented approaches.

## 4.2 Preprocessing

### 4.2.1 Disfluency Removal

Whenever people speak freely, such as in lectures, meetings or other dialogs, the speech is mainly spontaneous, *i.e.*, the utterances are neither read nor rehearsed ahead of time.

> *"'[...] as usual —um— a few comments on the literature. Um— there is one book I— I like very much, so if you want to learn about linear algebra and— and the practical issues —um— that —uh— come in if you deal with linear algebra, you should really have a look into this book by Trefethen and Bau on lin— numerical linear algebra."*

This excerpt from the LMELectures transcripts *(PA06 at 0:35:04)* shows some of the typical artifacts found in spontaneous speech, which should be removed prior to further processing. Most dominant among these are disfluencies such as hesitations and repairs. For example, *um*, *uh* and alike are fillers typically used to maintain the attention of the listener while thinking about the continuation of the sentence. Word fragments and repairs as in *"lin— numerical linear algebra"* are also frequently observed. More complex disfluencies such as *sentence* restarts or repairs, *e.g.*, "He wore a blue– no wait– a red shirt," are harder to detect and fix as they would require semantic understanding to some extent. Disfluencies can be removed using a variety of unsupervised and supervised methods, for example based on words, knowledge-based rules or probabilistic models [Wang 10]; the latter can be adapted to the current speaker [Hona 05]. The disfluency removal implemented for this thesis focuses on hesitations, word fragments and basic repetitions, which can be easily removed using a set of manually defined rules. For manual transcriptions, the hesitations and abortions need to be coded in a consistent way. Automatic speech recognition systems typically output a special symbol for word fragments and hesitations, or are able to produce a lexical form of the partial word [Ryba 09].

| *word* | *PoS tag* | *description* |
|---|---|---|
| I | PRP | personal pronoun |
| 'd | MD | modal |
| rather | RB | adverb |
| have | VB | verb, base form |
| a | DT | determiner |
| big | JJ | adjective |
| , | — | punctuation |
| greasy | JJ | adjective |
| sandwich | NN | noun, singular or mass |

Figure 4.1: "I'd rather have a big, greasy sandwich," annotated with part-of-speech (PoS) tags. A complete list of the tags can be found for example in [Sant 90].

### 4.2.2 Part-of-Speech Tagging

In a second step, each word is automatically attributed a certain part-of-speech category, such as *noun*, *verb* and alike, which is best illustrated by an example given in Fig. 4.1.

Although the tagging seems a trivial task, ambiguities forbid a simple lookup in a dictionary. For example, words like "fly" or "burn" may be either a noun or a verb in base form depending on the context. Statistical taggers try to solve possible ambiguities by considering words in context rather then individually. Current state-of-the-art systems are based on HMMs [Jura 08] or linear chain conditional random fields (LCCRF), a variant of the traditional HMMs. A LCCRF can be used to predict a certain label sequence (the PoS tags) $\boldsymbol{y}^* = \{y_1, y_2, \ldots, y_L\}$ given a certain input word sequence $\boldsymbol{w} = \{w_1, w_2, \ldots, w_L\}$ that maximizes the conditional probability $p(\boldsymbol{y}|\boldsymbol{w})$. Simply speaking, a LCCRF is a discriminatively trained HMM that can be decoded using the Viterbi algorithm, with the difference that the complete observation sequence is accessible at any time. A detailed introduction to (LC)CRF can be found in [Sutt 06], along with parameter estimation and inference algorithms.

Most implementations use a feature set described in [Bril 92] that includes the word itself, its neighbors, and previously assigned tags. More advanced features include the presence of certain suffixes, capitalization or hyphenation. CRF based taggers achieve an accuracy of about 95% for both written and spoken language [Thed 99, Huan 07, Zhan 09]. The tagger used in this thesis is the open-source CRFTagger which is distributed with English models trained on articles of the Wall Street Journal. It achieves an accuracy of about 97% [Phan 06].

### 4.2.3 Basic Lemmatization

As mentioned in the introduction, the frequency of words or phrases will play a major role for the identification of salient phrases. Phrases may, however, appear in slightly different forms, for example, "Johns Hopkins University" is often mispronounced as "John Hopkins University" or the use of plural and singular forms. In terms of phrase frequency, these surface forms should clearly be merged to one *lemma*. A typical

| *condition* | *rule* | | | *example* | | |
|---|---|---|---|---|---|---|
| — | SSES | → | SS | caresses | → | caress |
| $m > 0$ | IVENESS | → | IVE | decisiveness | → | decisive |
| $m > 0$ | FUL | → | $\emptyset$ | hopeful | → | hope |
| $m > 1$ | MENT | → | $\emptyset$ | adjustment | → | adjust |

Table 4.1: Example rules of Porter's suffix stemming algorithm [Port 80]; $m$ is a syllable-like measure based on the number of vowel/consonant alternation.

approach to unify the word surface forms is to map the word to their stem which is often implemented as suffix deletion or modification[2]. Tab. 4.1 shows some example rules of Martin Porter's suffix stemmer [Port 80] which is still probably the most popular algorithm despite its age. The drawback of such a suffix based algorithm is that the generalizations may be either not successful, *e.g.*, in case of irregular verbs such as run–ran, or too aggressive by stripping too much of the word. This is tolerable, as the later phrase extraction focuses on noun phrases whose components are less error-prone.

More sophisticated lemmatization such as acronym expansion, *e.g.*, "HMM" → "hidden Markov model," semantic unions, *e.g.*, "car, truck, vehicle," or reference resolution, *e.g.*, "[...] of the President. He */the president/* said [...]," are not (yet) considered, as they require extensive linguistic analysis which is out of the scope of this thesis.

### 4.2.4 Stop Words

A final step of the preprocessing is to mark *stop words*, *i.e.*, words that should be ignored later because they are too frequent or do not convey particular information. A long time standard procedure in text mining, the manually compiled list of stop words includes frequent words such as personal pronouns, frequent verbs such as "be," "are," and alike, and other frequent words, for example "because," "either," *etc.*. For spontaneous speech, this list needs to be extended by colloquial expressions and, if known, speakers habits. Typical examples of American English include "yeah," "'cause," "gonna," or, for the west coast, "like." For this thesis, a stop word list of about 600 words is used, including the examples mentioned before.

It is important to note that stop words are not simply deleted neither are candidate phrases containing stop words. It might, however, be beneficial to ignore candidate phrases that begin or end with a stop word [Witt 99], or to ignore phrases that consist only out of stop words.

[2]Suffix stemming is of course only valid if the morphological variation is at the end of a word, which is mainly the case for the English language.

## 4.3 Key Phrase Candidate Selection

The actual key phrase extraction is a two-stage process. In a first step, all possible candidates are extracted from the cleaned up transcripts which typically results in a rather long list. In a second step which is described in the subsequent section, the list of candidates is ranked and truncated to a certain length or by setting a certain minimum salience threshold. The candidate selection is usually based on either word connectivity or coherence, or related to quasi-syntactic structures such as PoS tags.

### 4.3.1 KEA: word Connectivity

The widely acknowledged Keyphrase Extraction Algorithm, KEA[3] [Witt 99], starts off splitting the textual input according to phrase boundaries such as punctuation marks, dashes, numbers and so on. From these initial phrases, every sub-phrase up to length three is extracted that does not begin or end with a stop word. The resulting set of phrases is then case folded and stemmed. While this works great for most kinds of written documents such as books, journals or even web pages, spontaneous speech often lacks these clear phrase boundaries. In fact, especially lecturers tend to speak without interruptions or at least perceptible sentence boundaries. Although lexical, acoustic and prosodic features have been successfully used to determine sentence boundaries automatically, it remains still an error-prone task ([Liu 07]. NIST RT'03[4]). Another drawback of KEA's candidate selection is that it results in a tremendous amount of phrases, with many of them being of questionable quality. Furthermore, the limitation to a phrase length of three might be a bad choice. Consider the following example utterance, again taken from LMELectures PA06:

> *"It computes the principal axes that's the one d axis that shows the highest spread of the points."*

Beside good phrases such as "principal axes" or "highest spread," KEA would extract "computes the principal" and "shows the highest" which are useless as the referenced noun is missing. However, "highest spread of the points" might be a good key phrase, too, which is missed due to the maximum phrase length.

### 4.3.2 Branching Entropy

A different method based on word cohesion is described in [Chen 10]. The authors argue for a "branching entropy," that is, possible key phrases (with a length greater than one) have strong cohesion (little branching entropy) within the sequence, but a weak cohesion (large branching entropy) at the start and the beginning. More

[3] `http://www.nzdl.org/Kea/`

[4] US NIST Rich Transcription Evaluation, Fall 2003. `http://www.itl.nist.gov/iad/mig/tests/rt/2003-fall/`

formally, the branching entropy $H^b(\varphi^{(i)})$ of some phrase $\varphi^{(i)}$, length $i > 1$, is defined as

$$p(\varphi^{(i+1)}) = \frac{\text{freq}(\varphi_j^{(i+1)})}{\text{freq}(\varphi^{(i)})} \tag{4.1}$$

$$H^b(\varphi^{(i)}) = -\sum_j p(\varphi_j^{(i+1)}) \log_2 p(\varphi_j^{(i+1)}) \tag{4.2}$$

where $\varphi^{(i)}$ is the examined phrase, *e.g.*, "spread," $\varphi_j^{(i+1)}$ are the possible child phrases, *e.g.*, "highest spread," and freq($\varphi$) is the frequency count if phrase $\varphi$. Expanding the phrase left or right results in separate branching entropies. The probabilities of higher order phrases are expressed as a frequency ratio regarding the generalizing phrase. If "highest spread" were to be a key phrase, not only the frequency count has to be large but most of the time "spread" has to be preceded by "highest". In other words, the left (and right) branching entropy is a measure of perplexity, *i.e.*, how many different words precede (and follow) a certain word or phrase. Valid key phrase candidates are those phrases $\varphi$, that have a higher left and right $H^b$ than average [Chen 10]. The idea of the branching entropy is in principal similar to measuring the mutual information between the final word of a phrase and the remaining preceding words, one of the features used in AT&T's *How May I Help You* call routing system [Gori 97].

Despite its interesting combinatoric motivation, the branching entropy approach has a few drawbacks. The algorithm has a rather high computational complexity as the branching entropies need to be explored to a desired n-gram length. Phrases of length one are not considered. The entropy values and their distribution strongly depend on the input data, thus setting a selection threshold to a mean value might not be the best choice[5]. Furthermore, stop words, which naturally have high branching entropies, are possible sources of error.

### 4.3.3 PoS Patterns

The candidate phrase extraction applied in this thesis follows a different motivation which is more tailored to speech found in meetings and lectures. Recall the previous example from PA06. This time, possible good candidates are underlined:

> *"It computes the principal axes that's the one d axis that shows the highest spread of the points."*

It is easy to see that the marked phrases are (extended) noun phrases, which are intuitive key phrase candidates: in most of the cases, it is most important to know about the "things" rather than the dependent verbs. For example, knowing that a text contains "president" and "federal budget bill" may be better than knowing that the text is about "vote" and "resign" as verbs indicate a certain action, *e.g.*, pass or reject the bill, and must thus be put in context with their dependent noun phrases.

[5] This holds especially for short [spoken] documents. The high accuracy of the key phrase extraction in [Chen 10] may be optimistic due to the fact that most reference key phrases were English technical terms embedded in Taiwanese language.

The noun phrases can be extracted using a regular expressions applied to the respective part-of-speech tags:

$$2^* 1^+ ( 3^+ 2^* 1^+ )^*$$

where

$$1 = (\text{NN} \mid \text{NNS} \mid \text{FW} \mid \text{CD} \mid \text{VBG} \mid \text{NNP} \mid \text{NNPS})$$
$$2 = (\text{JJ} \mid \text{JJR} \mid \text{JJS} \mid \text{VBN}), \quad 3 = (\text{DT} \mid \text{IN})$$

represent the components of a noun phrase: (proper) nouns, foreign words, numbers and gerunds form the main component (1), adjectives, as well as comparative and superlative, and past participles (2) can be placed ahead of the noun. Finally, more basic noun phrases can be appended using determiner and prepositions (3). Applied to the prior example, the regular expression successfully detects all the previously underlined words and phrases (*cf.* Fig. 4.2).

| It | computes | the | principal | axes | that | 's | the | one | d | axis |
|---|---|---|---|---|---|---|---|---|---|---|
| PRP | VBZ | DT | JJ | NNS | IN | VBZ | DT | CD | NN | NN |
| — | — | 3 | 2 | 1 | 3 | — | 3 | 1 | 1 | 1 |

| that | shows | the | highest | spread | of | the | points |
|---|---|---|---|---|---|---|---|
| IN | VBZ | DT | JJS | NN | IN | DT | NNS |
| 3 | — | 3 | 2 | 1 | 3 | 3 | 1 |

Figure 4.2: Transcription, part-of-speech tags, and categories. The regular expression $2^* 1^+ ( 3^+ 2^* 1^+ )^*$ extracts noun phrases.

This algorithm has two advantages. First, the PoS tagging is very reliable, thus most of the noun phrases are detected independently of the actual phrase length. Second, allowing only noun phrases helps to get rid of useless phrases at an early stage. Going back to the example, the greedy matching of the regular expression detects all noun phrases, thus redundant ones need to be eliminated at some point later in the processing chain. A good heuristic is to delete the shorter n-gram, if the enclosing n-gram has the same frequency count, thus the shorter n-gram only appears in the context of the longer one.

PoS tags also play a strong role in the detection of named entities (NE), that are locations, names, dates, and other categories of phrases. While the NE type is important for certain tasks in spoken language understanding (*e.g.*, [Levi 07]), it is dispensable at this stage.

## 4.4 Unsupervised Key Phrase Ranking

The following unsupervised ranking strategies mainly rely on two measures, the frequency $f_i$, or related empirical distribution $p(\varphi_i)$, and length $n_i$ of phrase $\varphi_i$. While $f_i$ is just a raw count, $p(\varphi_i)$ (or *term frequency* $tf_i$) is the empirical distribution, and thus dependent on the document:

$$p(\varphi_i) = tf_i = \frac{f_i}{\sum_i f_i} \tag{4.3}$$

It is important to note that the phrases, and thus the associated frequency counts, are referenced in the lemmatized form instead of the actual lexical surface form. That way, phrases such as "Markov model" and "Markov models" are merged. The length $n_i$ of phrase $\varphi_i$ is defined as the number of words in the PoS categories 1 and 2, thus all but determiners and prepositions. The following ranking strategies show some of the possible heuristics. While they all mainly rely on the above two criteria, they integrate different amounts of prior knowledge. The functions are designed so that a larger value suggests a higher salience.

### 4.4.1 No Language Model

The simplest heuristic does not require any prior knowledge and is based on the phrase frequency and length. While frequency is in principle a good indicator of salience, longer phrases are naturally inferior to shorter ones due to inclusion. For example, "house" will always have the same or larger count as "big house". A good means to compensate that effect is to boost longer phrases when computing the salience of phrase $\varphi_i$:

$$\text{f x len1}\,(\varphi_i) = \begin{cases} f_i & n_i = 1 \\ f_i \cdot (n_i + 1) & n_i > 1 \end{cases} \tag{4.4}$$

This linear re-weighting introduces a strong bias towards very long phrases, which makes sense to some extent: ha *et al.* [Ha 02] showed for Chinese and English text data, that bi-gram ($n_i = 2$) frequencies are about an order of magnitude larger than 5-gram frequencies. But a closer look at the data suggests that most of the key phrases have a length of two or three meaningful words, which can be modeled in an empirical way:

$$\text{f x len2}\,(\varphi_i) = f_i \cdot n_i \cdot \exp\left(-\frac{1}{5}\, n_i^{3/2}\right) \quad . \tag{4.5}$$

Fig. 4.3 shows the magnitude of the two different frequency scaling coefficients given the phrase length. A different motivation to boost long phrases is that the longer a repeated phrase is, the more likely it is that the repetition was on purpose, thus the phrase is of interest.

### 4.4.2 Corpus Specific Language Model

A common heuristic in information retrieval is to combine the term (phrase) frequency with with a different, corpus specific frequency. The *document frequency* [Spar 72] indicates, how may documents contain a certain phrase. The motivation is that if a phrase occurs in one document but rarely in others, it could be important. If a phrase occurs in about every document, it might represent some general fact and is thus unlikely to be important. Most commonly, the *inverse* document frequency

$$idf_i = \log \frac{\#\text{ documents}}{\#\text{ documents containing } \varphi_i} \tag{4.6}$$

is combined with the term frequency to form the well-known *tf x idf* (*e.g.*, [Baez 99]):

$$tf\text{-}idf(\varphi_i) = tf_i \cdot idf_i \tag{4.7}$$

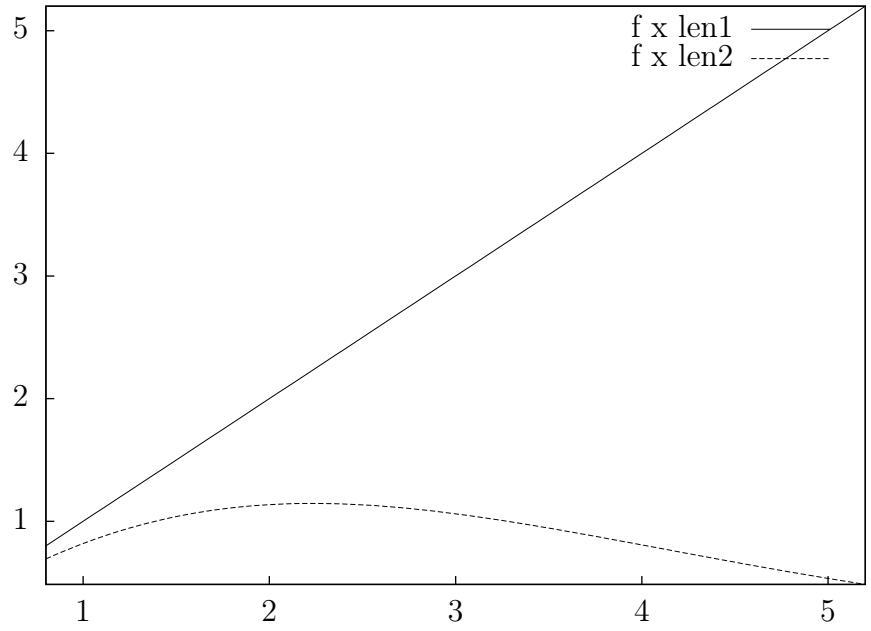



Figure 4.3: Frequency scaling components for f x len1 and f x len2 based on the phrase length $n_i$.

The *tfidf* values have proven to be useful for large corpora — they are a core component of large-scale indexers such as KEA or LUCENE[6]. The lecture corpora are typically small, and lecture series usually range in between 12 and 24 sessions, depending on the semester schedule of the university. Nonetheless, the *idf* values estimated from individual lecture series provide a solid base as they can be used to distinguish the lecture-specific terms from general terms of the covered topic. For spoken language, the *idf* values can capture certain idioms or ways of saying of an individual lecturer which is beneficial in case of a single lecturer for one series, but may lead to errors if a series is read by various speakers. Again, the $tfidf$ values are combined with the phrase length as

$$\text{tf-idf x len1}\,(\varphi_i) = \begin{cases} idf_i \cdot tf_i & i = 1 \\ idf_i \cdot tf_i \cdot (n_i + 1) & i > 1 \end{cases} \tag{4.8}$$

$$\text{tf-idf x len2}\,(\varphi_i) = idf_i \cdot f_i \cdot n_i \cdot \exp\left(-\frac{1}{5}\, n_i^{3/2}\right) \quad . \tag{4.9}$$

### 4.4.3 General Background Model

A similar but more general way is to compare how often a phrase appears in the target document compared to its frequency in some large background corpus, an idea which goes back to Edmundson *et al.* [Edmu 61]. The advantage is that the background model can be learned from a very large data set, and can thus provide a solid estimate of how often certain words or phrases "normally" appear. Edmundson *et al.* suggested

[6] `http://lucene.apache.org/`

to consider the ratio or differences of the relative frequencies [Edmu 61]. But a more natural way to compare the empirical distributions is the Kullback-Leibler divergence [Kull 51]

$$\mathrm{KL}\,(p,q) = \sum_i p(i) \cdot \log \frac{p(i)}{q(i)} \tag{4.10}$$

where the $p(\cdot)$ and $q(\cdot)$ are the relative frequencies of the phrases in the target document and the background model. Tomokiyo and Hurst [Tomo 03] suggested to use what they call the *point-wise* KL divergence to estimate the salience of a certain phrase $\varphi_i$ as

$$\mathrm{kl}\,(\varphi_i) = p(\varphi_i) \cdot \log \frac{p(\varphi_i)}{q(\varphi_i)} \quad . \tag{4.11}$$

Finding the relative phrase frequencies for the target document is straight forward, while finding the ones that correspond to the background model, unfortunately, is not. While it is very likely that every single word of the target document is also found in the background corpus, the longer the phrases are, the less likely they appear in both. An easy combinatorics exercise: suppose there are 50 000 words, there are theoretically $50\,000!/49\,997! = 124\,992\,500\,100\,000$ phrases of length three. Even if only a small share of these resemble meaningful phrases, the chances are high that a phrase of the target document does not appear in the background corpus.

In [Grop 10, Grop 11], the authors suggest to consider only words and phrases listed in WordNet [Fell 98]. The relative frequencies are estimated from the British National Corpus (BNC), a 100 million words collection of samples of written and spoken language [Burn 07], and add-one smoothing (*cf.* 3.8.2) is used for missing phrases. A drawback of this ranking function is that possibly important phrases containing certain technical terms may be sparse in both the target document and background corpus, so they will be missed.

### 4.4.4 Positional Heuristics

A simple heuristic is to find salient phrases based on their position within the document. For written documents, Edmundson and Wyllys [Edmu 61] note that phrases close below certain headings, *e.g.*, *Introduction*, *Conclusion*, tend to be more relevant than others. For spoken documents, this heuristic is hard to transfer if no topical segmentation exists. Edmundson *et al.* further observe, that relevant sentences, and thus phrases, tend to appear rather early or late within a document [Edmu 61]. But again, this is not necessarily transferable to spoken language. For example, meetings may have hot discussions, or lectures may introduce a new topic half way through the session.

However, analyzing the position, or better, the occurrences of a phrase is worthwhile. If a phrase is frequent but occurs somewhat regularly throughout the document, the phrase may be particularly useful. On the other hand, if it occurs frequent but in a short period of time, the phrase seems salient. Zhang *et al.* [Zhan 04] suggest to divide the document equally sized segments of some arbitrary small size, *e.g.*, ten words or 30 seconds, and compute the per-segment empirical phrase distribution

$$p(\varphi_i, s) = \frac{f_{is}}{f_i} \tag{4.12}$$

where $f_{is}$ is the frequency of phrase $\varphi_i$ in segment $s$. The $p(\varphi_i, s)$ can be used to compute the phrase entropy as

$$H(\varphi_i) = -\sum_s p(\varphi, s) \log p(\varphi, s) \quad . \tag{4.13}$$

The segmentation introduces another parameter that needs to be carefully tuned. To avoid that, Gropp suggests to make use of the empirical phrase distribution function [Grop 10]

$$\hat{F}(\varphi_i, t) = \frac{f_i^t}{f_i} \tag{4.14}$$

where $f_i^t$ is the frequency count of phrase $\varphi_i$ up to time $t$. $\hat{F}$ can now be compared to an ideal distribution $F$ using the Kolmogorov-Smirnov (KS) test (*e.g.*, [Knut 97])

$$\mathrm{ks}\,(\varphi_i) = \sqrt{f_i}\,\underset{t}{\mathrm{argmax}} \left| \hat{F}(\varphi_i, t) - F(t) \right| \quad . \tag{4.15}$$

It is sufficient to evaluate this formula for these times $t$ where the $\varphi_i$ occurs. Reconnecting to the prior motivation, the empirical distribution function is compared to the uniform one. A large value indicates a strong difference and thus a possibly salient phrase. While this measure can be used for single documents, the distributions could also be estimated for a sequence of recordings, for example a certain lecture series. That way, phrases which occur rather uniformly throughout one lecture but rarely occur in other recordings of that series can be correctly identified as salient.

The KL and KS based ranking strategies are both not augmented by the lengths as they consider the distribution rather then the plain frequency of a phrase. For the other strategies, the length is integrated to make longer phrases with naturally lower counts competitive to frequent short phrases.

## 4.5 Ranking Evaluation Measures

The quality of automatically extracted key phrases and their ranking is either directly assessed by human raters, or determined by comparison with lists of manually selected and ranked phrases. Both approaches involve human raters and thus introduce two problems. First, the annotators are subjective, *i.e.*, they tend to disagree on how many key phrases to select, and if general or specific phrases should be preferred. Second, rankings —in a strict sense— do not allow multiple items to be assigned the same rank which would be desirable for example for synonymous phrases. Furthermore, the actual value of two items might only be weakly related to their rank, for example, consider the two *greater-than* rankings 10-5-1 and 3-2-1. While the subjective bias of human ratings is typically eased by sourcing multiple annotators, the value *vs.* rank issue can be directly addressed using suitable evaluation measures.

### 4.5.1 Precision, Recall and F-Measure

The most straight forward evaluation of a (ranked) list of phrases is to verify for each phrase if it is also included in the reference list. The *recall* $R$ is defined as

$$R = \frac{\textit{number of correctly identified key phrases}}{\textit{number of reference key phrases}} \quad . \tag{4.16}$$

| | **A** | **B** | **C** | **D** | **E** | **F** | **G** | **H** | **I** | $R$ | $P$ | $F$ | $\rho$ | AP(4) | NDCG(4) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **HUM** | | | 0 | 1 | | | | 2 | 3 | | | | | | |
| **M 1** | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 1 | 0.5 | 0.5 | 0.96 | 0.42 | 0.36 |
| **M 2** | | | 1 | 0 | | | | 3 | 2 | 1 | 1.0 | 1.0 | 0.60 | 1.00 | 1.00 |

Table 4.2: Simple example of human (HUM) and machine (M) rankings; the letters A through I represent example phrases. $P$, $R$, $F$ and $\rho$ are measured with respect to the human ranking, AP and NDCG are limited to rankings of length four, the phrases C, D, H, I were assigned with salience one.

However, the recall can typically be boosted by increasing the number of extracted phrases. To account for that fact, the *precision* $P$

$$P = \frac{number\ of\ correctly\ identified\ key\ phrases}{total\ number\ of\ identified\ key\ phrases} \tag{4.17}$$

indicates how accurate the system is. If the recall were to be increased by extracting more phrases, the precision would severely decline. Finally, these two measures are typically combined in the *F-measure*

$$F = \frac{2 \cdot R \cdot P}{R + P} \tag{4.18}$$

to get a more unbiased performance indicator. However, P, R and F do not reflect the quality of the order of the list items which is a substantial part of the experiments in this thesis. They are listed for completeness but are not further considered in the experiments. Tab. 4.2 shows an example ranking and the respective scores of the machine rankings compared to the human one. Although machine 1 achieves perfect recall, the precision is rather due to the large number of selected phrases. The perfect recall and precision score of machine 2 does not reflect the unfavorable ranking.

### 4.5.2 Spearman's Rank Correlation Coefficient

Spearman's rank correlation coefficient $\rho$ [Spea 04] is closely related to the well known correlation coefficient $r$ [Pear 96]. In fact, the only difference is that the correlation coefficient is computed on the *ranks* of the items instead of their actual values. Given two sequences of values $x$ and $y$, *e.g.*, two different key phrase rankings, the correlations can be computed as

$$r(x, y) = \frac{\mathrm{cov}(x, y)}{\sigma_x \sigma_y} \tag{4.19}$$

$$\rho(x, y) = r(\mathrm{rank}(x), \mathrm{rank}(y)) \tag{4.20}$$

where $\mathrm{cov}(x, y)$ is the covariance of, and $\sigma_{x,y}$ are the standard deviation of $x$ and $y$. Note that in the case of ranked key phrases, the ranking is typically in descending

order regarding some phrase relevancy score. Unfortunately, $\rho$ can only be applied to lists of same lengths, and truncations typically lead to skewed values – recall the simple (already ranked) example in Tab. 4.2: machine 1 shows a significantly higher $\rho$ than machine 2, although the latter is clearly to be favored. As with precision and recall, $\rho$ is listed for the sake of completeness but is not further considered for the experiments in this thesis.

### 4.5.3 Average Precision

Precision and recall can be modified to incorporate the rank information and reflect quality of the ranking. Average Precision (AP) [Baez 99] evaluates how many relevant phrases are placed on top of the list. Using binary relevance judgments $\chi(\varphi) = \{0, 1\}$ of a phrase $\varphi$, the precision at the position $n$ of a key phrase list can be defined as the fraction of relevant phrases in the top $n$ items of that list as

$$p_n = \frac{\sum_{i=1}^{n} \chi(\varphi_i)}{n} \quad . \tag{4.21}$$

Considering the $N$ top items of the candidate list, the Average Precision is computed as

$$\mathrm{AP}(N) = \frac{\sum_{n=1}^{N} p_n \chi(\varphi_n)}{\sum_{n=1}^{N} \chi(\varphi_n)} \quad . \tag{4.22}$$

For the phrases annotated for the LMELectures data, the grade interval (1 to 6) is split in half, assigning $\chi(\varphi) = 1$ if phrase $\varphi$ was rated 1 to 3, and $\chi(\varphi) = 0$ otherwise. As with the regular precision, AP values range from 0 to 1, with 1 being the best possible value. The parameter $N$ that controls how many items of the ranking are actually considered needs to be set depending on the target application. Typically, $N$ is set to the minimum number of phrases to allow a complete computation of the average values with respect to all rankings. It is important to understand that AP is a combination of manually defined binary relevance judgements and the automatically generated ranking. Simply speaking, $\mathrm{AP}(N)$ is the average percentage of relevant phrases when considering the top $N$ phrases of an automatically generated ranking.

### 4.5.4 Normalized Distributed Cumulative Gain

Similar to AP, the Normalized Distributed Cumulative Gain (NDCG) as described by [Jarv 00, Jarv 02] rewards placing salient phrases at the top of the produced list, putting emphasis on the actual order of phrases. But instead of a binary measure, each phrase $\varphi_i$ is assigned a *gain* (high relevance equals high gain) that models the contribution of each phrase more detailed than just a binary decision. The gain is multiplied with a discount factor to incorporate the position within the list (less impact of low ranked phrases). A common choice for the gain function is to map the relevancy scores using an exponential function to emphasize top ranks. The phrases of the LMELectures data are annotated with grades from 1 to 6, thus the gain is set as

$$\mathrm{gain}(\varphi_i) = 2^{(6-\mathrm{grade}_{\varphi_i})/5} - 1 \quad . \tag{4.23}$$

Although the original discount function is the reciprocal logarithm, a common choice is to use $1/log(1 + \text{rank})$ to avoid a special treatment of the first item. The base of the logarithm controls how much the top ranks should be emphasized. The NDCG with respect to the top $N$ list entries is computed as

$$\text{NDCG}(N) = C \cdot \sum_{i=1}^{N} \frac{\text{gain}(\varphi_i)}{\log_2(1+i)} \tag{4.24}$$

where the normalization factor $C$ ensures that the ideal (perfectly sorted) list has an NDCG value of 1, and other values range within 0 and 1. Tab. 4.2 shows AP and NDCG in comparison to R, P, F, and $\rho$. The identical value of AP and NDCG for machine 2 is due to the identical salience values of all human phrases.

### 4.5.5 Multiple Annotators – Agreement and Performance

The key phrases in the LMELectures data were annotated by multiple human subjects which allows to study the inter-rater agreement, *i.e.*, to what extent do people agree when selecting and ranking key phrases, and the quality of automatically extracted and ranked key phrases. While the inter-rater agreement is an indicator for the complexity of the task, it can also be used as a performance measure when substituting a human annotator with a machine and tracking the change in inter-rater agreement. This helps to answer the question how good a machine can do *instead of* a human, *i.e.*, would one be able to tell the difference between automatically and manually extracted and ranked key phrases.

Traditional inter-rater agreement measures such as Cohen's $\kappa$ [Cohe 60] or Krippendorff's $\alpha$ [Krip 03] only consider pair-wise independent comparisons and thus neglect the contextual ranking information. The AP and NDCG values, however, incorporate both relevance and ranking information and can be computed for each human-human and human-machine pairing. The resulting values are then averaged, and will be listed separately for the human-human ("reference") and each automatic ranking. Tab. 4.3 illustrates which reference and ranking pairing contributes to which overall average score. Note that the lecturer's phrases are not considered due to the small number and lack of salience annotation.

It would be desirable for the average human-machine scores to be in a same range as the human-human scores, as this would suggest that the extraction and ranking algorithms produce likable rankings. An average human-machine agreement higher than the human-human agreement would indicate that the algorithms produce rankings that satisfy the average human more than an individual human ranking, which would also be an interesting observation.

## 4.6 Experiments and Results

### 4.6.1 Ground Truth, Manual and Automatic Key Phrases

The experiments focus on the manual transcripts of lecture *PA06*, which was manually annotated with a small set of key phrases by the lecturer, and set of about 20 phrases

| **Reference** | **Ranking** | **Contributes to Score** |
|---|---|---|
| Annotator 1 | Annotator 2 | *reference* |
| | Annotator 3 | *reference* |
| | ⋮ | ⋮ |
| | *f x len1* | *f x len1* |
| | *f x len2* | *f x len2* |
| | ⋮ | ⋮ |
| Annotator 2 | Annotator 1 | *reference* |
| | Annotator 3 | *reference* |
| | ⋮ | ⋮ |
| | *f x len1* | *f x len1* |
| | *f x len2* | *f x len2* |
| | ⋮ | ⋮ |
| ⋮ | ⋮ | ⋮ |

Table 4.3: Computing the overall evaluation scores by averaging every reference and ranking pairing.

each by five annotators (see the annotation description in Chapter 2). The lecturer's phrases are considered ground truth while the other more detailed key phrase rankings represent the subjective impression of the labelers. The left part of Tab. 4.4 shows which of the ground truth phrases were also listed by the human annotators. The very general phrase "motivation" is not listed by any annotator. The coverage of the other phrases varies with the labeler. The only consistently listed phrase is "linear regression," which is a recurring topic throughout the lecture.

The right part of Tab. 4.4 lists, for each key phrase ranking strategy, the rank of the ground truth phrase, if it was extracted. Similar to the human key phrase lists, all ranking strategies award a high rank to "linear regression." The fact that all ranking strategies find "motivation" but place it at a very low rank suggests that this is an artifact of the extraction algorithm: the word may appear just twice and is thus included as possible key phrase. The rankings "f x len2" and "ks" both list all ground truth phrases but the respective ranks are rather low. Although the automatic rankings show a rather consistent relative ranking of the phrases, "tfidf x len2" shows the best performance, listing four of the five important phrases in the top 50, similar to "f x len1" with the exception of "ridge regression" (ranked 59).

### 4.6.2 Average Human and Automatic Performance

Tab. 4.4 allows two further conclusions. First, human annotators are subjective and do not necessarily agree with each other or the lecturer on the key phrases. This is despite the fact that the annotators list about 20 phrases on average, and thus cover more detail. And second, if a key phrase appears more than once in the transcript it will be included in the ranking, but likely with a very bad rank. So when evaluating

| *Lecturer's Phrases* | *Annotator 1* | *Annotator 2* | *Annotator 3* | *Annotator 4* | *Annotator 5* | f x len1 | f x len2 | tfidf x len1 | tfidf x len2 | kl | ks |
|---|---|---|---|---|---|---|---|---|---|---|---|
| linear regression | • | • | • | • | • | 5 | 14 | 3 | 3 | 4 | 1 |
| norms | • | | • | • | ○ | 6 | 2 | – | 10 | 7 | 87 |
| dep. linear regression | | | ○ | | ○ | 30 | 93 | 8 | 23 | – | 379 |
| ridge regression | • | | • | | • | 59 | 115 | 65 | 50 | – | 252 |
| discriminant analysis | ○ | | ○ | ○ | | – | 290 | – | – | – | 399 |
| motivation | | | | | | 320 | 432 | 276 | 343 | 231 | 401 |

Table 4.4: Master key phrases of lecture *PA06* assigned by the lecturer, coverage indicators (•) for the human annotators, and phrase rank of the automatic rankings, if applicable. The empty bullets (○) indicate a partial match, *e.g.*, "linear discriminant analysis" satisfies "discriminant analysis."

the quality of the automatic rankings, the lists should be shortened to a reasonable length, and compared to each of the human annotators.

Fig. 4.4 shows the average AP and NDCG scores for comparing one human ranking to the remaining four, considering list lengths from one to 14 (the shortest human key phrase ranking). Note that the lecturer's phrases are not further considered due to the small number and lack of salience information. Both measures show a peak when considering only the top two phrases. For more than three phrases, AP decreases, while NDCG improves again after a local minimum when considering the top five phrases. This is due to the definition of the evaluation measures. AP awards score for placing valuable phrases on top of the list, however, the $\chi$ in both the nominator and denominator sum in Eq. 4.22 effects that the AP only changes if another valuable phrase is included. As a consequence, a ranking might have a rather high AP score, but in fact may only have a few valuable phrases on top followed by invaluable phrases. Furthermore, the $\chi$ function is implemented as a relevance threshold that does not reflect the nuances of the annotated values, which range from one to six. NDCG incorporates this relevance via the gain function (Eq. 4.23). Thus the quality of a ranking might indeed increase when considering longer lists.

Fig. 4.5 shows the average AP scores when considering up to 20 phrases of the automatic rankings. The average human performance (*reference*) is added for comparison. Beside the "kl" and "ks" measures, the automatic rankings show a similar performance as the *reference* ranking. For more then 7 phrases, the "len1" based measures show even higher performance than the average human, *i.e.*, the automatic rankings fit the average human ranking better than a human ranking, suggesting a notion of objectiveness of the rankings. The AP scores seem to level out for more than 10 phrases, which is due to the above mentioned definition of AP.

Fig. 4.6 shows the average NDCG scores when considering up to 20 phrases of the automatic rankings. Again, the average human performance (*reference*) is added for comparison. Although the overall performance figure is similar for both human

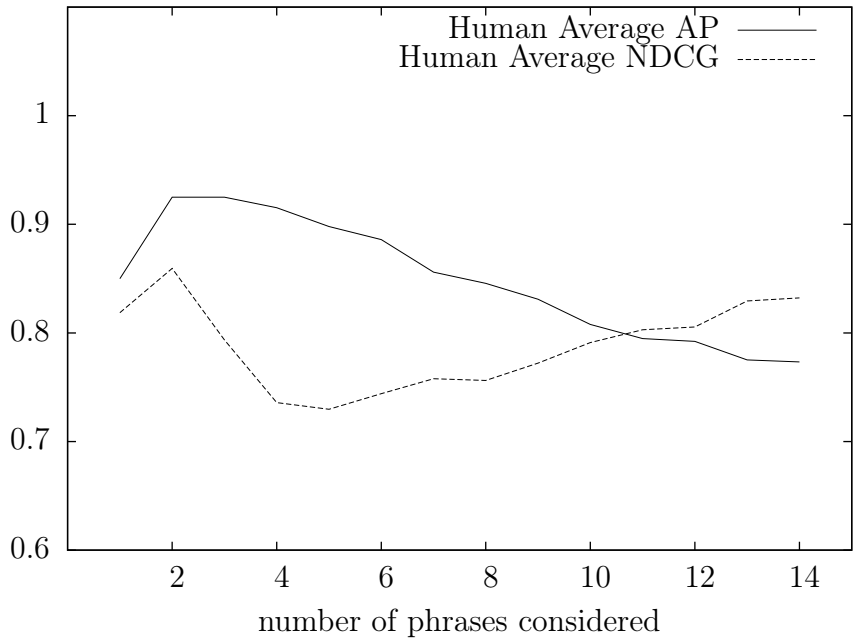



Figure 4.4: Average AP and NDCG values for comparing one human ranking to the remaining four (*reference*).

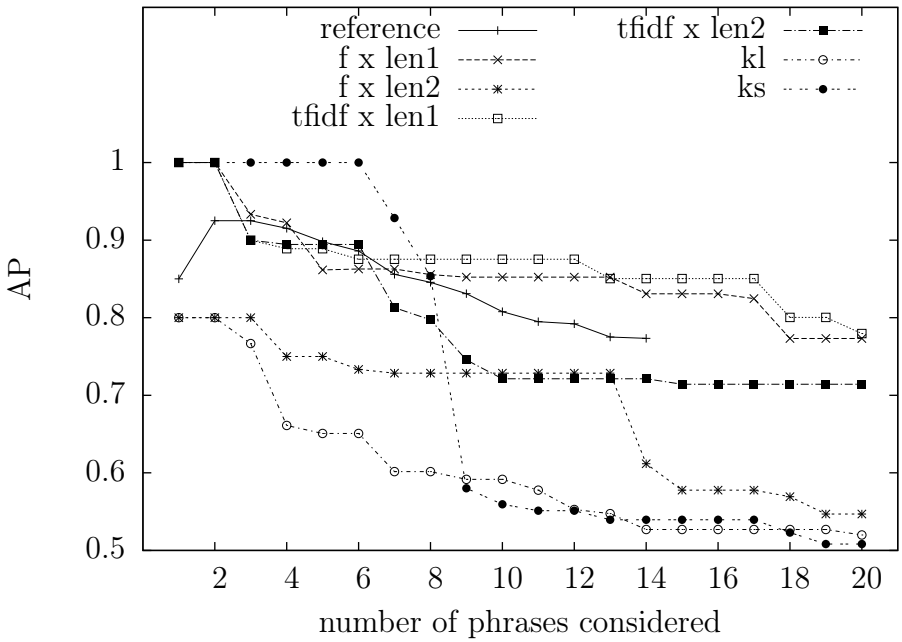



Figure 4.5: Average AP for the human ("reference") and automatic key phrases using the manual transcripts of lecture *PA06*.

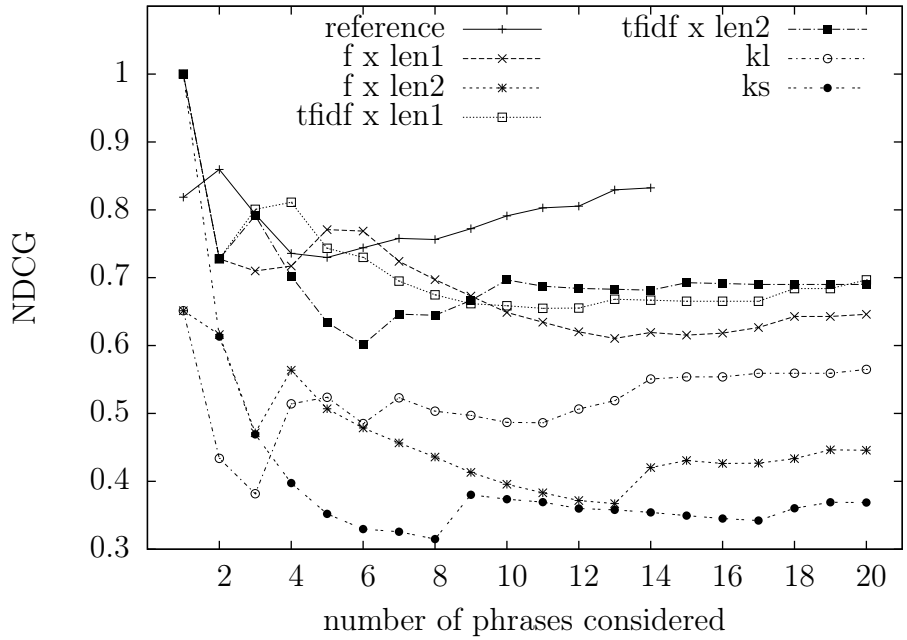



Figure 4.6: Average NDCG for the human ("reference") and automatic key phrases using the manual transcripts of lecture *PA06*.

and automatic rankings – a slight increase after an initial decrease – the "tfidf" based systems perform most similar to the average human.

Using automatic instead of manual transcripts of PA06 (*sgmm4c*, *cf.* Ch. 3.11.4) leads to an overall similar result, as shown in Figs. 4.7 and 4.8. The word error rate of about 10% mainly affects the "len2" and the distribution based "ks" measure. Beside plain recognition errors, phrases may be corrupted by erroneous word insertions (or deletions) which obviously directly affects the "ks" measure. The "len2" based rankings emphasize phrases of length two and three rather than a linear up-weighting. This asymmetric behaviour can skew the ranking both emphasizing "wrong" phrases but deemphasizing good phrases. The "len1" based measures remain rather unaffected by the recognition errors and show a competitive performance compared to the human reference scores.

## 4.7 Discussion

The main conclusions to be drawn from the experiments in the previous section is that the automatic rankings appear of similar quality as the human rankings, especially for a smaller number of key phrases such as five to eight. That indicates that the most salient phrases can be identified reliably but that both human and automatic rankings tend to disagree on the less salient ones. This subjectiveness, or personal preferences, is hard to satisfy without prior knowledge or, in case of supervised algorithms, adaptation to the target user, a fact that will be again discussed in the following chapters.

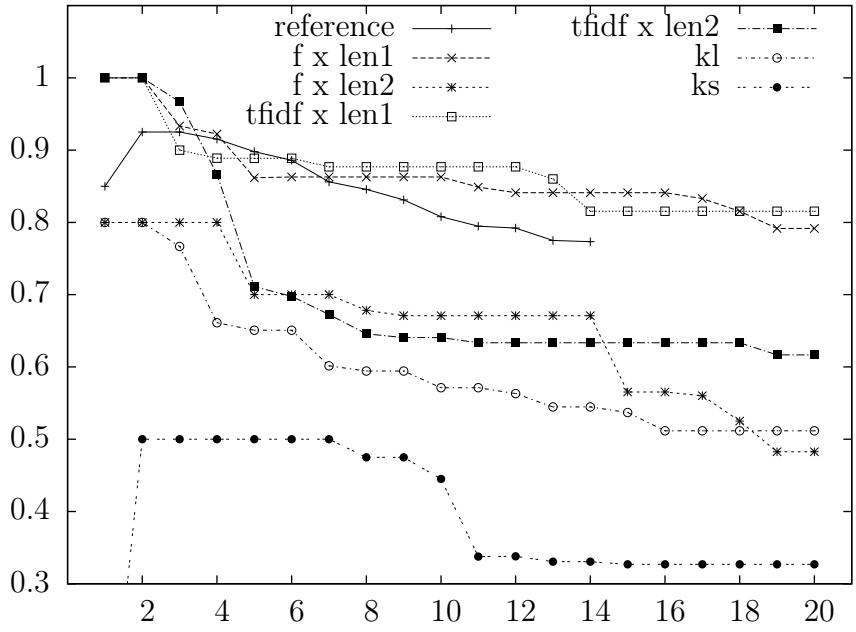



Figure 4.7: Average AP for the human ("reference") and automatic key phrases using the automatic transcripts of lecture *PA06*.

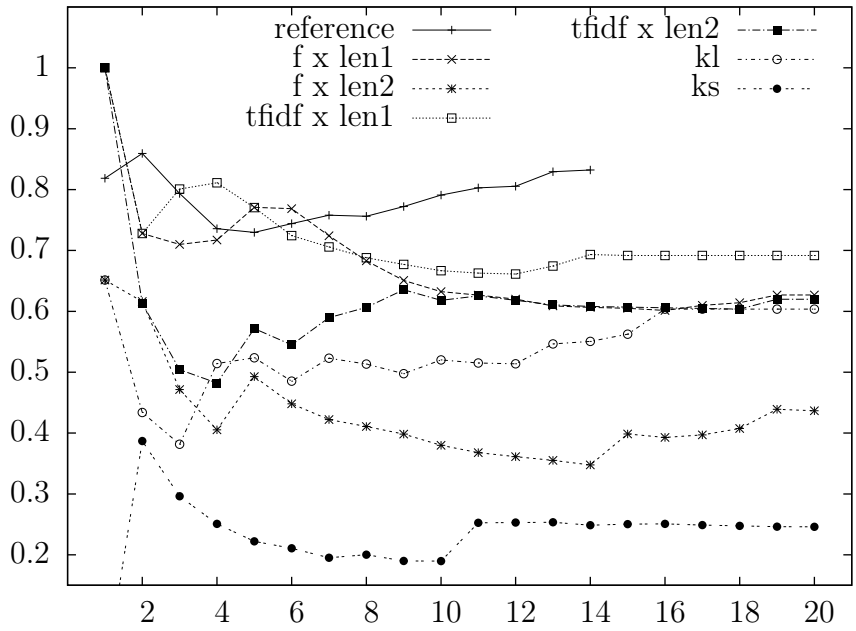



Figure 4.8: Average NDCG for the human ("reference") and automatic key phrases using the automatic transcripts of lecture *PA06*.

The results also show that using corpus (or domain) specific background knowledge, here based on the "tf-idf" values, can greatly enhance the ranking. The general background models ("kl") or positional heuristics ("ks") did not show convincing results, especially when considering longer rankings. The used background corpus (BNC) may be too general to distinguish regular from salient phrases, while the positional heuristics might require either longer documents or a different implementation that allows to incorporate synonyms and abbreviations when counting the occurrences.

The key phrase extraction and ranking is, to some extent, robust with respect to transcription errors. This is however not necessarily credited to the key phrase extraction or ranking strategy but to the fact that on the one hand important phrases are typically well articulated and on the other hand that the speech recognizer tends to fail for fast, hasty or colloquial expressions but shows high accuracy for clean, well-articulated speech. However, the simpler ranking strategies are less sensitive to recognition errors which is why they are favorable if a low recognition performance is expected.

In the following chapters, the key phrases and their rankings will be used to generate automatic summaries and to present the phrases in an interactive user interface.

# Chapter 5

# Extractive Speech Summarization

The previous chapter concluded that it is possible to extract salient key phrases from recorded speech and put them in a meaningful order using some weighting strategy. Although these phrases already give a good intuition on the content, the actual meaning and context can only be guessed. This chapter introduces to extractive speech summarization methods and describes how to generate summaries from speech recordings to put the phrases in context and give a compact representation of the recording. Parts of this chapter have been published in [Ried 08b, Gill 09, Ried 10].

## 5.1 Introduction

A summary of a textual, audio or video document is a document that contains the important aspects in a short, condensed way.

Automatic text summarization dates back to the early days of computer science. In the late fifties, Hans Peter Luhn published his fundamental article "The Automatic Creation of Chemical Abstracts" [Luhn 58] in which he laid the foundations of this field. In the following years, Luhn's approach was extended to incorporate syntactic information [Clim 61], position-based features and cue words [Edmu 69]. In the early seventies, Karen Spärck Jones put emphasis on multi-document summarization [Spar 72], which is still the dominant topic in current research. Since the new millennium, the United States National Institute of Standards and Technology (NIST) supports research on summarization by hosting the annual Document Understanding Conferences (DUC, 2001-2007) and Text Analysis Conference (TAC 2008-2012).

Compared to that, *speech* summarization is a fairly new topic that originates from porting methods from text summarization. It has been applied to various genres: broad cast news [Hori 02, Chri 04, Inou 04, Mask 05, Mroz 05, Zhan 07b], lectures [Furu 04, Mroz 05], telephone dialogs [Zech 02, Zhu 06] and meeting transcripts [Murr 05a, Liu 08, Ried 08b]. Speech summarization has also been part of large-scale research projects as for example the Cognitive Assistant that Learns and Organizes (CALO) [Tur 08, Tur 10], with technology transfer to Apple's Siri[1].

These genres can be very different in terms of type of speech and content presentation, and are thus best summarized using different approaches. For example, broad-

[1]Personal communication with former SRI International employees. Siri, Inc. was a 2007 spin-off of SRI International's venture group which was acquired by Apple in 2010.

cast news are very similar to textual news articles, as they are in fact well-structured non-redundant sentences read from a teleprompter. Meeting conversations are typically less structured, with spontaneous utterances that involve interaction between humans. As a result, meeting speech contains utterances with little information content such as backchannels (*e.g.*, aha, hm, yeah), sentence restarts and corrections, and a lot of redundancy. Lectures, on the other hand, have a similar style but are typically given by a single speaker that tries to form rather simple sentences to have students follow her thoughts. However, sentence boundaries are often unclear and utterances have unnatural pauses, especially if a blackboard is used while talking.

Research on meeting summarization took great advantage of the availability of two large and well-annotated corpora, namely the ICSI [Jani 03] and AMI meeting corpora [McCo 05]. This common ground stimulated a concentrated effort with reproducible and comparable results. Lecture summarization, however, lacks such data sets. Most research groups work independently from each other and on different —and differently annotated— data, producing results that are hard to compare:

- He *et al.* [He 99, He 00] work on a set of internal technical talks from Microsoft. Following simple heuristics, they extract segments based on slide transition, pitch information and slide visibility. The different algorithms are evaluated by human subjects based on questionnaire performance.

- Zhang and Fung [Zhan 07a] work on a corpus of conference presentations held by different speakers. The talks are in Chinese Mandarin and last about 15 min each. In addition to lexical and discourse features, they utilize the rhetorical structure to generate separate summaries for the introduction, main part and conclusion.

- Penn and Zhu [Penn 08] use computer science lectures recorded at the University of Toronto. The talks are given by different teachers and last about 50 min each. The lectures are annotated by three human subjects that were asked to mark utterances that correspond to lines in the presentation slides as well as main utterances that support each line. The lectures are then split into short eight to 15 minute long segments which are independently summarized.

Due to the very heterogeneous data, annotations and experiments, and due to the lack of human abstracts for the LMELectures data, the summarization algorithms presented in this thesis are evaluated on the well-studied ICSIMC. This also allows to compare the results with findings in related prior work. The speech in both the LMELectures and ICSIMC data sets is technical and spontaneous, thus the results are likely transferable. Furthermore, the assumptions underlying the key phrase extractions also hold for meeting data[2] — interlocutors tend to use the same phraseology for things they consider important, a phenomenon known from psycholinguistics as lexical entrainment [Bren 96]. To complete the overall picture, the results section contains an example summary of the LMELectures corpus.

Summarization evolved to a diverse set of tasks which can be categorized along several dimensions [Mani 01, Lin 09b, Tur 11]:

[2] In fact, early versions of the key phrase extraction were originally developed for meeting data.

**Extractive vs. Abstractive** A summary can be *abstractive*, *i.e.*, a freely formulated text, or *extractive*, *i.e.*, a concatenation of sentences extracted from the document[3] Abstractive, or generative, summarization is typically a combination of distilling domain specific information and generating the summary by filling templates with that knowledge. Although the idea of generative summarization is very appealing, the implementation is still a hard problem. Beside the problem of extracting and organizing the document knowledge, the generation of natural language still mainly relies on hand crafted rules and templates, and are thus very domain and language specific. Extractive summarization algorithms tend to be domain and language independent, and have received increasing attention in the past years. The length, or compression factor, is a crucial constraint. Typically, the summary length is required to be fixed, *e.g.*, 500 words, or a percentage of the length underlying document, *e.g.*, 10%. Instead of a sentence selection, some approaches compress sentences by removing superfluous words [Hori 02, Furu 04, Liu 09].

**Single vs. Multiple Documents** As briefly mentioned earlier, summaries can cover single or multiple documents. For the latter, the provided documents are typically centered around a similar topic, making redundancy a major issue.

**Type of Media** Summarizations can be based on textual data, *e.g.*, newspaper articles, audio data, *e.g.*, meeting recordings, or video data such as talk shows or video lectures. While this all regards the *input*, the *output* modalities can also differ. Although most summaries are in textual form, audible or visual summaries, *i.e.*, a concatenation of the salient audio or video snippets, can be constructed, the input data permitting. Also, visualizations of the data can also provide clues about the salient information.

**Generic vs. Specific** *Specific* summaries are based on a query or selected topic, providing a clear definition of what is supposed to be in the summary. Producing a *generic* summary can be a tricky task, as the definition of "generic" is very subjective. On the one hand, it is hard to judge the level of detail, as it depends on the world knowledge of the system or user. On the other hand, it is impossible to specify or quantify what amount of detail the user expects from a generic summary in advance.

**Interactivity** One way to address the issue of genericness is to move away from a single pass system that produces a final summary based on the initial request. Allowing the user to interact with and pass feedback to the summarization process can enhance the use of summarization systems. The user may identify actually salient sentences [Mies 07, Lin 10] or modify the weight of words or phrases which are used to build the summary [Ried 08a].

**Other Dimensions** More recently, the traditional summarization task has been extended to more specific applications such as *update* or *opinion summarization*

[3] As the presented algorithms can be applied to both text and speech, utterance and sentence as well as recording and document are used interchangeably.

(*cf.* NIST TAC). The former task is to produce a summary that indicates the new information given two sets of documents. The latter focuses on identifying *opinions* instead of facts, requiring a more in-depth analysis of presumably redundant sentences and a disambiguation between facts and quotes or opinions.

This chapter focuses on *extractive* speech summarization, *i.e.*, selecting sentences from automatic or manual transcriptions. The interesting possibilities of user interaction and visualization are discussed in the following chapter.

Utterance selection can be done either supervised or unsupervised. Supervised methods rely on a classifier, typically a support vector machine [Burg 98], that predicts a binary label (or probability) for each input sentence whether it should be included in the summary or not. Features include textual, structural and acoustic cues, such as term frequency, inverse document frequency [Spar 72] and alike from the information retrieval community, sentence position and length [Murr 06], and prosodic information like fundamental frequency and energy contour, speaking rate and pauses, and speech artifacts like disfluencies and repetitions [Inou 04, Mask 05, Zhu 06, Xie 09].

Although supervised approaches can achieve a high performance if task and domain are limited and suitable training data is available, they often fail when migrating to a new domain or task. Unsupervised methods are designed to be independent of topic, task and training data, and are thus more appealing. They consist mainly of algorithms ported from the text summarization community which can be categorized using the criteria below.

**Iterative vs. Global Models** Early algorithms such as Maximum Marginal Relevance (MMR) [Carb 98] build a summary step by step by selecting the sentence that is most relevant but least redundant to the previously selected sentences. This is repeated until the target length constraint is fulfilled. While this algorithm proved to be very successful, the shortcomings are easy to see. The greedy selection process can result in a suboptimal set as the currently best candidate might not be the best in the long run due to the pair-wise comparisons. This observation leads to the idea of finding an optimal selection from a global point of view. Instead of an iterative scheme, a global objective function is defined that can be optimized subject to some constraints. Although computationally more complex, a model of that kind can overcome the shortcomings of iterative models.

**Sentences vs. Concepts** Aside from the algorithm topology, summarization algorithms can be distinguished by their semantic granularity. Until a few years ago, most extractive summarization algorithms worked on the sentence level scoring, *i.e.*, the decision whether the sentence should be included in the summary is based on measures that regard the sentence as an independent unit. Concept based summarization assumes that the information is spread out over the sentences as little pieces, the concepts, that can be names, things, events and alike. The idea is now to find a selection of sentences that covers most of these concepts while satisfying the length constraint which is similar to finding a solution to the knapsack problem.

The remainder of this chapter is dedicated to the design and evaluation of global unsupervised summarization models. The objective functions for the sentence and

concept based models are implemented as integer linear programs (ILP) that can be efficiently optimized using solvers like `glpsol`[4].

## 5.2 Sentence Based Summarization

Most sentence based extractive summarization systems rely on measuring relevance and redundancy to produce a summary which is most informative while being least redundant. These two measures can and need to be balanced to control the content of the summary. The well-known MMR is, in its original formulation in [Carb 98], a greedy algorithm that iteratively selects that sentence that has the best trade-off between relevance to the summary and redundancy to the most similar sentence that is already included in the summary. Formally, the MMR score of a sentence $i$ is defined as

$$\mathrm{MMR}(i) = \lambda \,\mathrm{Rel}(i) - (1-\lambda) \max_{j \in S} \mathrm{Red}(i,j) \tag{5.1}$$

where $\lambda$ balances the relevance score $\mathrm{Rel}(i)$ of sentence $i$ and the redundancy penalty $\mathrm{Red}(i,j)$ when including both sentences $i$ and $j$ in the summary $S$. The algorithm terminates when the summary length constraint is reached.

Traditionally, relevance is measured as a similarity to a user defined query or, in case of a generic summary, a *document centroid*. However, both approaches are problematic. Consider the centroid as an "average sentence" of the document. In case of spontaneous speech, this centroid is severely skewed by colloquial expressions, idioms or just non informative words. A user defined query can strongly guide the summarization process, however, this is somewhat bound to a chicken-and-egg problem: to form a good query, the user needs to know what the text is about — but if this would be the case, no summary would be required in the first place.

To escape this vicious circle, relevance is modeled using key phrases (and their weights) [Ried 08a]. More specifically, the relevance of a sentence is the sum over the occurring key phrases. Redundancy is modeled as a normalized word overlap (ignoring stopwords). The stopword list contains about 500 manually selected words including pronouns, articles, particles and other frequent function words in order to not distort the redundancy score[5].

$$\mathrm{Rel}(i) = \sum_k \chi(\varphi_k, i) \cdot w_k \quad ; \quad \mathrm{Red}(i,j) = \frac{\mathrm{words}(i) \cap \mathrm{words}(j)}{\max(\mathrm{words}(i), \mathrm{words(j)})} \tag{5.2}$$

where $\chi(\varphi_k, i) = 1$ if phrase $\varphi_k$ is present in utterance $i$ and 0 otherwise, and $w_j$ is the key phrase weight (either a rank or the underlying salience score). The flaw that two sentences with the same key phrases but of different lengths yield the same redundancy score is compensated in the later optimization which inherently favors shorter sentences in presence of same scores.

As mentioned in the introduction to this chapter, the selection resulting from the greedy solution is likely to be suboptimal as a sentence, once selected, is not

[4] `glpsol` is part of the GNU Linear Programming Kit, GLPK, `http://www.gnu.org/software/glpk`

[5] Measuring redundancy in terms of key phrase overlap would lead to many similar or equal scores, depending on the number of phrases available.

reconsidered for inclusion in favor of other sentences. For example, a single longer sentence could be selected instead of two short ones to cover the same information with less words. This issue can be addressed by reformulating MMR as a global objective function

$$\text{Maximize:} \quad \textstyle\sum_i \left[\lambda \operatorname{Rel}(i) s_i - (1-\lambda) \max_j \operatorname{Red}(i,j) s_i s_j\right] \tag{5.3}$$

$$\text{Subject to:} \quad \textstyle\sum_i l_i s_i \leq L \tag{5.4}$$

where $s_i$ is a binary indicator of the inclusion of utterance $i$ in the summary, $l_i$ is the length of sentence $i$ and $L$ is the maximum allowed length of the summary.

Unfortunately, the max operator results in a non-linear 0-1 quadratic problem. One way of finding an approximate solution is to apply optimization techniques like Monte-Carlo search or genetic algorithms. A more elegant way is to change the MMR formulation so that it resembles a linear problem. Using additional constraints, an MMR-inspired approach can be written as an integer linear program (ILP) as found in [McDo 07]:

$$\text{Maximize:} \quad \textstyle\sum_i \left[\lambda \operatorname{Rel}(i) s_i - (1-\lambda) \sum_{j \neq i} \operatorname{Red}(i,j) s_{ij}\right] \tag{5.5}$$

$$\text{Subject to:} \quad s_{ij} \leq s_i \qquad \forall i,j \wedge i \neq j \tag{5.6}$$

$$s_{ij} \leq s_j \qquad \forall i,j \wedge i \neq j \tag{5.7}$$

$$s_i + s_j \leq 1 + s_{ij} \qquad \forall i,j \wedge i \neq j \tag{5.8}$$

$$\textstyle\sum_i l_i s_i \leq L \tag{5.9}$$

where the additionally introduced indicator $s_{ij}$ indicates the presence of the sentence *pair* $i$ and $j$ in the summary. The constraints 5.6–5.8 ensure that $s_{ij}$ equals 1 if and only if both $s_i$ and $s_j$ equal 1. The max in the redundancy term is replaced by a sum over the redundancy of all pairs of selected sentences which penalizes a sentence inclusion according to its average redundancy to the other included sentences.

While the approximation of McDonald is already superior to the greedy version, it is possible to formulate an ILP which is a global equivalent of the original MMR. The key aspect is to convert the inner working of the max operator to ILP constraints:

$$\text{Maximize:} \quad \textstyle\sum_i \left[\lambda \operatorname{Rel}(i) s_i - (1-\lambda) \sum_{j \neq i} \operatorname{Red}(i,j) m_{ij}\right] \tag{5.10}$$

$$\text{Subject to:} \quad \textstyle\sum_j m_{ij} = s_i \qquad \forall i \tag{5.11}$$

$$m_{ik} \geq s_k - (1 - s_i) - \textstyle\sum_{j:\, \operatorname{Red}(i,j) \geq \operatorname{Red}(i,k)} s_j \qquad i \neq k \tag{5.12}$$

$$m_{ij} \leq s_i \qquad \forall i,j \wedge i \neq j \tag{5.13}$$

$$m_{ij} \leq s_j \qquad \forall i,j \wedge i \neq j \tag{5.14}$$

$$\textstyle\sum_i l_i s_i \leq L \tag{5.15}$$

The binary variable $m_{ik}$ indicates that $\operatorname{Red}(i,k)$ is the maximum among the $\operatorname{Red}(i,*)$. The idea is to define constraints that help to explicitly compute which sentence is most redundant to which other sentence of the summary. Once a sentence is selected, there must be exactly one other selected sentence which is considered the most redundant one (Eq. 5.11). If a sentence $k$ is maximum redundant regarding sentence $i$, *i.e.*, $m_{ik} = 1$, no other sentence with higher redundancy to $i$ may be selected (Eq. 5.12). Of

course, both sentences $i$ and $k$ need to be included in the summary (Eqs. 5.13 and 5.14). The balancing of redundancy can lead to the interesting phenomenon that the resulting summary is way shorter than the target length, especially for small $\lambda$: the optimization may reach a point where selecting any more sentences would decrease the objective function value. This makes sense in a general setup where documents may be short and the summary length is more of an upper limit but the later experiments focus on fixed-length summaries in favor of a fair comparison. The following additional constraint ensures that the resulting summary length is within 50 words of the target length:

$$\sum_i l_i s_i \geq L - 50 \tag{5.16}$$

This formulation has significantly more constraints than the formulation by McDonald, but the lack of the linear approximation makes it an optimal solution to the MMR problem.

Although ILPs with large numbers of constraints can –in general– be solved efficiently, the experiments show that the optimization routines take an unacceptable large amount of time[6], requiring an early termination. The reason is the design of the objective function and the underlying data. The redundancy scores are very similar or even equal for a large number of sentences, leading to many solutions of similar or even equal value. This issue will be discussed in the experiments. Other approaches avoid this relevance-redundancy trade-off and the related issues by building a graph where sentences resemble nodes, and the vertices represent the similarity. Garg *et al.* [Garg 09] identify "hot" nodes as potential candidates for extraction and use a bag-of-words criteria to determine the redundancy between the candidate and the summary. Lin *et al.* [Lin 09a] compute sub-modular selections and add a redundancy penalty term to the graph cut function.

## 5.3 Concept Based Summarization

The major deficit of sentence level redundancy scoring is the limitation to pairs of sentences. As a result, redundancy caused by groups of more than two sentences is missed. Concept based summarization steps back from the sentence units and considers the actual bits of information spread over the utterances, for example, the underlined portions in Fig. 5.1. The idea is to find that set of sentences that covers the most concepts while obeying a certain lengths constraint. In the example, the three sentences, as a group, introduce redundancy that is not detected by pair-wise comparison. The proposed solution is to credit each concept that is included in the summary only once, thus finding the best solution is a natural way of avoiding redundancy while maximizing relevancy.

The notion of concepts has been around for some time, however they have been mainly utilized for evaluation purposes. Measures such as ROUGE [Lin 04], Pyramid [Nenk 04] or Basic Elements [Hovy 06], which will be discussed in Sec. 5.4, are based

[6]Up to several hours or even days on a high performance machine for small problems of about 500 sentences.

| | | $(c_1)$ | $(c_2)$ | $(c_3)$ |
|---|---|---|---|---|
| $(s_1)$ | You will see, this convex problem can be solved efficiently. | • | • | |
| $(s_2)$ | The convex problem requires Lagrange multipliers. | • | | • |
| $(s_3)$ | It be solved efficiently using Lagrange multipliers. | | • | • |

Figure 5.1: Redundancy introduced by a group of sentences. A pair-wise comparison will not reveal that (1) is subsumed by (2) and (3).

on overlap regarding n-grams, manually annotated salient fragments, or dependency parsing relations. Here, the concepts are represented by the previously extracted key phrases, and their rank or weight are used to guide the optimization routine to prefer important phrases.

The concept based model is to some parts analogous to the sentence based model. Variables indicate the presence of concepts and the inclusion of sentences in the summary while additional constraints ensure a coherent setting. The objective function to maximize is the quality of a possible summary defined as a sum over the positive weights $w_i$ of the included concepts. As with the sentence based model, the main constraint ensures that the summary length, *i.e.*, the sum over the lengths of the selected sentences, is shorter or equal a constant $L$.

$$\text{Maximize:} \quad \sum_i w_i c_i \tag{5.17}$$

$$\text{Subject to:} \quad \sum_j s_j l_j \leq L \tag{5.18}$$

$$s_j o_{ij} \leq c_i \quad \forall i, j \tag{5.19}$$

$$\sum_j s_j o_{ij} \geq c_i \quad \forall i \tag{5.20}$$

The binary variables $c_i$ and $s_j$ indicate the presence of concept $i$ in and the inclusion of sentence $j$ to the summary. Per definition, $c_i$ and $s_j$ are closely linked which needs to be incorporated in the ILP. If a sentence $j$ is included, all contained concepts are included. Vice versa, a concept $i$ can only be present if at least one sentence $j$ is selected that contains this very concept. This dependency is modeled using the constraints Eq. 5.19 and 5.20. In detail, Eq. 5.20 ensures that if concept $i$ is present, there is at least one sentence selected that covers it. Eq. 5.19, on the other side, assures that every active concept $i$ is incorporated in the objective function using a binary constant $o_{ij}$ which marks the occurrence of a concept $i$ in sentence $j$: if $s_j o_{ij} = 1$ then $c_i = 1$.

This model generalizes prior related work on text summarization. Filatova and Hatzivassiloglou [Fila 04] use an adaptive greedy algorithm to find a selection of sentences that maximizes the coverage of what they call events. These events are basically pairs of named entities ("relations") and the connecting words ("connectors"). After a reduction using some external knowledge base (in their case WordNet), the concepts are weighted by their normalized relation and connector frequency. Independently from the model described in this section, Takamura and Okumura [Taka 09] introduced an ILP formulation very similar to a previously published version of the above summarization system for textual (news) data [Gill 08]. As concepts, they use words and the associated weights that are either computed using an unsupervised generative model, or a supervised model using a prior trained logistic regression model.

Their Maximum Coverage Problem with Knapsack Constraints formulation is in fact very similar, however they miss the constraints Eq. 5.19. As a result, the objective function can be skewed in a way that there might be concepts present in the summary which are not included in the score. This possible ambiguity might also be the reason why they could not show that exact solutions are superior to greedy ones.

**Remarks Regarding the Implementation** The implementation of the summarization system is straight forward but is worth mentioning a few details.

- The key phrase assignment, *i.e.*, the decision which key phrase appears in which sentence, is independent of singular or plural forms; *e.g.*, "Markov model" will also trigger "Markov models."
- Sentences that are shorter than five words are discarded as they typically result from abandoned utterances or are meaningless if taken out of context.
- The key phrase weights from the different ranking strategies may greatly vary in terms of their numeric values as some depend on raw phrase counts and others on logarithmic ratios. To minimize the effect of these numeric differences, the key phrase weights for each document are scaled to a range of $(1, 5)$.
- The relevance and redundancy measures of the sentence based systems are normalized to a have a maximum of one to have comparable values and allow a fair trade-off.

## 5.4 Summarization Evaluation

Evaluating the quality of summaries is a difficult task due to the diversity of the possible candidate summaries which can be short or long, detailed or abstract, focused or general, to serve different purposes – in short, "it is impossible to evaluate summaries properly without knowing what they are for" [Spar 98]. In general, candidate summaries may be evaluated according to *intrinsic* or *extrinsic* criteria [Spar 96]. Intrinsic criteria relate directly to the quality of the presented summary. Measures such as ROUGE [Lin 04], Pyramid [Nenk 04] or Basic Elements [Hovy 06] compare the target summary to multiple reference summaries. These intrinsic measures are of great value for the design and validation of summarization algorithms as the reference summaries need to be generated only once for each task. In contrast to that, extrinsic evaluations aim on the *use* of a summary. Typically, these evaluations are rather expensive and subjective: human subjects have to perform certain tasks, *e.g.*, a decision audit [Murr 09], with either the full information or the summary in question. Although extrinsic evaluations seem to be the only thorough way to analyze the actual use of a summary, the required resources and the lack of reproducibility suggest to apply it only in selected cases, and use intrinsic measures to track the system performance in day-to-day research.

If the human reference summaries are extractive, *i.e.*, humans marked certain parts as salient, extractive summaries can be evaluated using precision, recall and F-measure as introduced in Sec. 4.5.1 of the previous chapter. However, similar

problems arise. Humans disagree on which and how many sentences to extract, and precision and recall may substantially depend on the target summary lengths. A main problem, especially for speech summarization, is redundancy. Consider two utterances A and B that convey the same information. The human rater may indicate A as salient and neglect B as redundant, while the machine decides exactly complementary. Although both summaries convey the same information, the recall would be zero. Due to this pitfall, these measures are not further considered in this thesis as they are of doubtful meaning for summarization and typically only applied in a supervised scenario when learning sentence salience.

A problem specifically for speech summarization is the mismatch between the automatic extractive summary and the human abstracts. While the latter are typically well-formed sentences with low redundancy, the extractive summary is a concatenation of pieces of spontaneous speech – which is typically neither well-formed nor non-redundant. Methods such as disfluency removal or utterance compression or rewriting can ease this structural difference, but are outside the scope of this thesis.

### 5.4.1 ROUGE

ROUGE, the Recall-Oriented Understudy for Gisting Evaluation [Lin 04], is a software toolkit to assess the quality of a summary with respect to a set of (typically human) reference summaries. The general idea is to measure the word or phrase (n-gram) overlap between two summaries while ignoring frequent stop words. The most commonly used version is ROUGE-2, which measures bi-gram overlap. Other popular versions include ROUGE-1 (uni-grams) and ROUGE-SU4, that measures uni-gram plus skip bi-gram[7] overlap. As the name already suggests, a target summary receives high scores for containing many n-grams from the reference summaries. Beside the recall, precision and F-measure can be computed as well. Known limitations of ROUGE include missed credit due to the lack of semantics, *e.g.*, reference resolution or detection of synonymous words, and extra credit attributed to overlap in non-informational n-grams [Sjob 07].

### 5.4.2 Other Evaluation Measures

The Pyramid method [Nenk 04] is in principal similar to ROUGE, but accounts for summary content units (SCU) instead of n-grams to avoid discredit due to non-informational overlap. These SCUs are manually identified from the set of reference summaries. SCUs may include single words, noun or verb phrases, or even partial sentences. These extra annotations require time and experience, and also introduce further subjectiveness.

Also similar to ROUGE, Basic Elements (BE) [Hovy 06] quantizes overlap between target and reference summaries. Instead of frequent n-grams, BE measures the overlap of so called *basic elements*. These basic elements include the head noun major syntactic constituents, or a relation between a head and a single dependent. The BEs can be defined manually or automatically using a syntactic parser and subsequent cutting operations on the parse tree. BE is rarely used to evaluate speech

[7] A bi-gram with interleaving words, here up to 4.

summarization, mainly due to the different structure of spoken (target) and written (reference) language and the unreliable parsing performance for spontaneous speech.

The extra annotations and complexity of Pyramid and BE along with the moderate annotation requirements may be the main reasons why ROUGE is still the most widely used evaluation measure for text summarization (*cf.* DUC/TAC proceedings). This also holds for speech summarization, although the validity of the measures is questionable [Murr 05b, Nenk 06, Gall 06]: the high correlation between human judgments and ROUGE scores for text summarization could not be confirmed for speech summarization, possibly due to the lack of large amount of data. However, ROUGE is still considered an indicator of speech summarization performance, and is thus the measure of choice for the experiments in this thesis. Of course, ROUGE and alike can only measure content. Coherence or linguistic quality are hard to evaluate automatically and are thus mainly evaluated subjectively.

### 5.4.3 Framing Results: Baselines and Oracle

The plain result of a complex evaluation measure like ROUGE, Pyramid or BE is hard to interpret without some reference values. An elegant way to display evaluation results in a meaningful way is to frame them by the results of simple baselines and a theoretical upper bound, the *oracle*, given the chosen metric [Ried 08b].

**Baselines** A simple way to generate an extractive summary is to simply select the next longest utterance until the target summary length is reached (*baseline1*). The intention is that longer utterances convey more, and more coherent information than short ones. The second baseline, *baseline2*, is the traditional greedy solution to MMR with the relevance and redundancy measured using the cosine distance to a simple term frequency based document centroid and the already selected utterances.

**Oracle** The idea of an oracle is to find the summary that yields the maximum possible evaluation score — in a cheating experiment. Recall the basic scoring principle of ROUGE: the more overlap between the target and reference summaries, the higher the score. The oracle solution *max-r* is obtained by applying the same ILP as with the concept based summarizer, but attributing weights proportional to the n-gram frequencies in the reference summaries. The result is a summary that shows the largest possible n-gram overlap with the reference summaries while satisfying the length constraint. As this is also the best solution that could be produced by the actual extractive summarization system, this is considered the oracle summary, and the associated ROUGE score the upper limit.

## 5.5 Experiments and Results

In this section, the sentence and concept based summarization models are analyzed, compared, and put into context with the baseline and oracle results. The experiments are conducted on the ICSIMC corpus due to the lack of reference summaries in the LMELectures corpus and to put the results in context with prior work. The following systems are implemented:

- *mmr/greedy* The original iterative (greedy) MMR algorithm using key phrase similarity as relevance and normalized word overlap as redundancy measure.
- *mcd/ilp* ILP for global inference as proposed by McDonald [McDo 07].
- *mmr/ilp* The ILP to solve the global formulation of MMR as described in Sec. 5.2 using the same relevance and redundancy measure as above.
- *concepts/ilp* Solution of the concept based summarization problem using weighted key phrases as introduced in Sec. 5.3. In case of enclosing key phrases, *e.g.*, *problem* in presence of *difficult problem*, only the longest matching phrase is considered.
- *n-kp* Instead of extracting utterances, output the top $n$ phrases as a textual summary. The idea is to generate a more compact representation with similar meaning or content.

The experiments are split in three parts. In the first part, the systems are compared and analyzed when computing summaries of a fixed length relating to the average length of the human reference summaries, on the manually transcribed meeting data. The explored parameters include the relevancy trade-off $\lambda$ (for MMR-based systems), the way of accounting for key phrases that may be enclosed by longer phrases, and the key phrase ranking strategies from the previous chapter. Furthermore, the sentence based ILP systems have a rather high computational complexity due to the very large number of constraints which can be addressed by either pruning the number of candidate sentences or restricting the computation time of the solver.

In the second part, the performance of selected system configurations is compared for a number of different length constraints to give an idea how the summary length and ROUGE score are related. In the last part, these systems are applied to the automatic transcriptions and respective key phrases, to shed some light on the performance in presence of erroneous transcriptions. This chapter concludes with a few example summaries and a discussion of their usefulness.

Throughout the remainder of this chapter, summarization scores are given as ROUGE-1 R (recall) as it reduces the bias when comparing written text with spontaneous speech. The ROUGE 2 and SU4 scores are in a lower range, but change very similar to the respective ROUGE-1 scores (*cf.* [Ried 10]). The recall measure puts emphasis on the coverage of important aspects in summaries of approximately equal length. The respective P (precision) and F values are typically in the same range but are more of interest if the compared systems may produce summaries of different lengths. For a trivial example, listing the whole document as a "summary" would lead to the highest possible recall, however the precision will be very low, while a system that produces a short summary may have a lower recall at a much higher precision. References to significant differences in scores are with respect to a 95% confidence interval.

### 5.5.1 Fixed Length

The first and major part of the experiments is dedicated to producing summaries of a length of 400 words which is about the average length of the human summaries for the

test set. This roughly corresponds to 5% of the overall spoken words. Unless stated otherwise, the experiments are conducted on the test set. Optimizing parameters on the test set is generally considered as bad practice: a system should be calibrated using additional development data to get a realistic estimate on the performance on unseen data. However, the meetings of the training set are only annotated with a single human summary which makes the ROUGE scores less reliable than scores computed with respect to three summaries as available for the test set. Furthermore, all methods are unsupervised as no data is required to train the summarization models so the performance is expected to be similar on the training set.[8]

### Computational Complexity

The *mmr/greedy* and *concepts/ilp* systems show a rather little complexity even for large number of concepts and utterances. On the machine used for these experiments, an Intel(R) Xeon(R) CPU X5450 at 3.00GHz, the computation time is only a few seconds and mainly relates to input/output operations.

The major downside of the global sentence based models is the large number of constraints that grows exponential with with the number of candidate sentences. The optimization is further complicated by the fact that many sentences will have similar redundancy scores (zero in absence of non-stopword overlap) leading to a large portion of the search space to be associated with very similar objective function values. In fact, earlier experiments showed that the optimization is not feasible in presence of several hundred sentences [Ried 10]. This section explores two possibilities to speed up the optimization. First, the number of sentences can be pruned prior to the formulation of the ILP to reduce the number of constraints. This was also suggested in [McDo 07]. This is done by pruning the list of candidate sentences to a certain number after sorting them by their relevance score. Second, the optimization can be terminated prematurely after a certain time thus leading to a sub-optimal solution. This is justified by the demand of a certain responsiveness of the summarization system.

Tab. 5.1 shows the summarization scores and number of optimal solutions (out of six) of the global sentence based models for different numbers of candidate sentences and computation time limits. The key phrases were ranked using "f x len1" and the relevancy parameter $\lambda$ was set to 0.9. Both settings are based on a previous similar experiment on the same algorithms and data set [Ried 10]. Furthermore, the "f x len1" based key phrase rankings showed a high agreement to human rankings on the LMELectures corpus (see Chapter 4)).

The scores of the greedy MMR and concept based models are added as a reference to show the effect of pruning candidate sentences prior to the optimization. The first observation is that there is no significant change in score for the greedy MMR and concept based systems when pruning the number of candidate sentences. This is best explained by the fact that the relevancy based pruning leads to a list of highly relevant

[8] Strictly speaking, some systems rely on the corpus specific *idf* values which are in this case estimated on the complete data set. However, these values could be estimated from documents taken from a similar domain.

| | | *mmr/greedy* | *mcd/ilp* | | *mmr/ilp* | | *concepts/ilp* |
|---|---|---|---|---|---|---|---|
| *# cand* | *t* | R-1 R | *# opt* | R-1 R | *# opt* | R-1 R | R-1 R |
| *no constraint* | | 0.15 | *(optimization not feasible)* | | | | 0.20 |
| 50 | 120 | 0.15 | 0.19 | 4 | 0.18 | 6 | 0.19 |
| | 360 | | 0.19 | 5 | 0.18 | 6 | |
| | 480 | | 0.19 | 5 | 0.18 | 6 | |
| | 600 | | 0.19 | 6 | 0.18 | 6 | |
| 75 | 120 | 0.15 | 0.19 | 0 | 0.18 | 2 | 0.20 |
| | 360 | | 0.18 | 1 | 0.18 | 4 | |
| | 480 | | 0.18 | 1 | 0.18 | 5 | |
| | 600 | | 0.19 | 1 | 0.18 | 5 | |
| 100 | 120 | 0.15 | 0.03 | 0 | 0.18 | 0 | 0.20 |
| | 360 | | 0.19 | 0 | 0.18 | 2 | |
| | 480 | | 0.19 | 0 | 0.18 | 3 | |
| | 600 | | 0.18 | 0 | 0.18 | 4 | |
| 150 | 120 | 0.15 | 0.00 | 0 | 0.00 | 0 | 0.21 |
| | 360 | | 0.02 | 0 | 0.10 | 0 | |
| | 480 | | 0.06 | 0 | 0.16 | 0 | |
| | 600 | | 0.12 | 0 | 0.16 | 0 | |
| 200 | 120 | 0.15 | 0.00 | 0 | 0.00 | 0 | 0.20 |
| | 360 | | 0.00 | 0 | 0.00 | 0 | |
| | 480 | | 0.00 | 0 | 0.03 | 0 | |
| | 600 | | 0.03 | 0 | 0.06 | 0 | |

Table 5.1: ROUGE-1 R (R-1 R) scores and number of optimal solutions (*# opt*, out of six) for the sentence based ILP models. The list of candidate sentences was pruned to a number of *# cand* prior to formulating the ILP, the computation time of the solver was limited to $t$ seconds on a . The *mmr/greedy* and *concepts/ilp* system are added to see how they would be affected by the same pre-pruning.

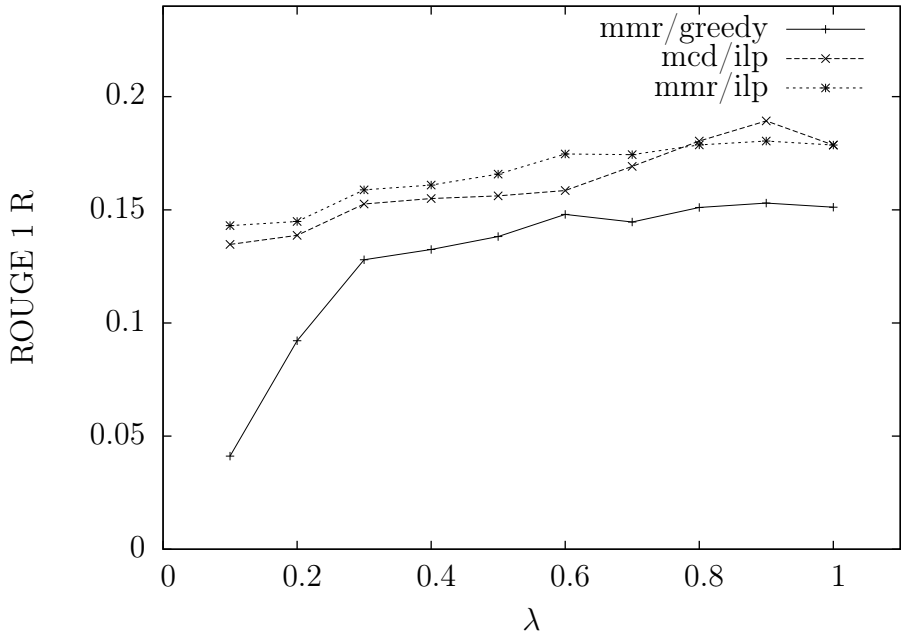



Figure 5.2: ROUGE-1 R scores for the MMR based systems using different $\lambda$ values and a target summary length of 400 words for the test set.

and thus often long sentences with more than ten words which are by nature the best candidates to be included in a summary, and the rather short target summary length.

The global sentence based systems react similar to each other: the more candidate sentences, the less likely is an optimal or near-optimum solution. However, independent of the number of candidate sentences, both optimum and near-optimum solutions result in similar summarization scores. This indicates that the optimization routine quickly identifies the most important utterances before getting stuck in finding suitable sentences for the remaining few words to fill. This is confirmed by the fact that the solver reported sub-optimal solutions within a few percentage points (in value) of the estimated maximum objective function. The outliers, *i.e.*, scores below 0.18 are due to empty outputs of the summarizer in case no integer solution to the ILP was found in time, thus no sentence was selected.

Based on these findings, the number of candidate sentences and the computation time will be limited to 50 and 360 seconds for the global sentence based models.

### Relevance-Redundancy Trade-Off

The sentence based models strongly depend on $\lambda$ balancing relevancy and redundancy. Naturally, this parameter depends on the data. While sentences found in the news may show little redundancy, spontaneous speech often shows a large amount of redundancy, making it an important factor. Here, $\lambda$ is systematically explored for the affected systems *mmr/greedy*, *mcd/ilp* and *mmr/ilp* by sampling values from $\lambda = (0.1, 0.2, \ldots, 1.0)$.

The results in Fig. 5.2 show a maximum performance using $\lambda = 0.9$ for all systems. The key phrases were again ranked using "f x len1" and the results confirm prior findings on the ICSIMC and another similar data set [Ried 10]. The rather strong emphasis on the relevance should not necessarily interpreted as an indicator of strong redundancy in the data, but as a balancing factor for two types of values which have a different numerical range. The relevance is per sentence and regarding the whole document, while the redundancy values are computed for sentence pairs. The extreme values of $\lambda = \{0, 1\}$ and thus ignoring the relevance or redundancy are conflict the motivation of MMR. While for $\lambda = 0$ the summarization score is indeed almost 0, the performance for $\lambda = 1$ is worse than for $\lambda = 0.9$, confirming that the relevance is a factor to be considered: less redundant content also means more room for additional information. The global MMR formulations appear less sensitive to especially small $\lambda$ than the original greedy formulation which is best explained by the optimization process. In the greedy approach, the decisions on including a sentence in a summary is final for each step, but the unreliable measure of redundancy in the early iterations due to the little number of already selected sentences may lead to sequence of bad decisions, especially if the relevance is only weighted little. The global model is able to modify the selection of summary sentences at every step to achieve an overall best solution and is thus less prone to unfavorable local decisions.

Based on these findings, the relevance-redundancy trade-off $\lambda$ will be set to 0.9 for the MMR based models. The results furthermore show that the global models outperform the greedy approach which will thus not be considered for further parameter explorations.

### Key Phrase Assignment

For both the sentence and concept based systems, the key phrase assignment, *i.e.*, the decision which key phrases appear in each phrase, is a critical step. The point is that longer phrases may entail shorter ones, for example, "hidden Markov model" and "Markov model." Here, two types of assignment are analyzed. The first and basic approach is to account for every phrase that appears, including entailed ones (*all*). While this is the most natural way, a closer look reveals that this leads to skewed relevance scores for the sentence based systems and limits the selection choices of the concept based system as selecting a sentence with a longer phrase automatically marks the entailed phrases as selected. Following this observation, the second type of assignment is to account only for the longest matching phrase to ease the bias in relevance scores and give the concept based model the utmost flexibility in selecting sentences (*longest match*).

There are two more strategies, namely to account for the shortest matching phrase or for the more relevant phrase. However, both are essentially a truncation of the key phrase list in terms of weight or n-gram length, which should be handled in the ranking process but is not desirable at this stage.

Tab. 5.2 shows the summarization scores of the global models, again using the "f x len1" ranked key phrases. The decline in performance when only accounting for the longest match is insignificant for the concept based model, similar as in [Ried 10], but significant for the sentence based models. That implies that those candidates with redundant phrases actually benefit from their artificially increased relevance,

| | *mcd/ilp* | *mmr/ilp* | *concepts/ilp* |
|---|---|---|---|
| all | 0.19 | 0.18 | 0.20 |
| longest match | 0.16 | 0.16 | 0.19 |

Table 5.2: ROUGE-1 R scores for different key phrase assignments using both sentence and concept based global systems.

| | *mcd/ilp* | *mmr/ilp* | *concepts/ilp* |
|---|---|---|---|
| f x len1 | 0.19 | 0.18 | 0.20 |
| f x len2 | **0.20** | 0.18 | 0.20 |
| tfidf x len1 | 0.19 | 0.17 | 0.19 |
| tfidf x len2 | 0.19 | **0.19** | **0.21** |
| kl | 0.19 | **0.19** | 0.19 |
| ks | 0.19 | 0.18 | 0.20 |

Table 5.3: ROUGE-1 R scores for different key phrase weightings using both sentence and concept based global models. The bold numbers denote the best weighting strategy for each system (column).

which corresponds to an implicit up-weighting of longer phrases. In fact, the this performance decline is even stronger and also significant for *concepts/ilp* when using the weighting "f x len2" which puts less emphasis on longer phrases than "f x len1." Thus, all phrase occurences are considered for the following experiments.

### Different Key Phase Weighting Strategies

In the previous chapter, six different key phrase weighting strategies were defined and evaluated with respect to human annotated phrases. Here, the effect of different weightings on the summarization performance is analyzed which can also be interpreted as an application based evaluation of these rankings.

Tab. 5.3 shows the summarization scores of the global models, using the six different ranking strategies. Although the systems tend to perform best for the "len2" based weights, the score differences are insignificant. This suggests that as long as the key phrases are reasonably well chosen and ranked, the resulting relevant utterances are either the same or similar. This does however not implicate that the different key phrases are equally important regarding the human summaries which can be shown by using only the key phrases as a textual summary.

Fig. 5.3 shows the summarization performance of system *n-kp* when using the different strategies and an increasing number of top $n$ phrases to form the textual summaries. With more than 75 phrases, the overlap with the human summaries is significantly higher than of the respective extractive summaries for selected rankings while having less words. The better performance in terms of ROUGE-1 R has to be judged with precaution as these "summaries," in contrast to the extractive ones, do not contain any verbs or context and may thus be misleading. Furthermore, a large $n$ will lead to a high ROUGE-1 R score simply by chance. Consider the extreme case

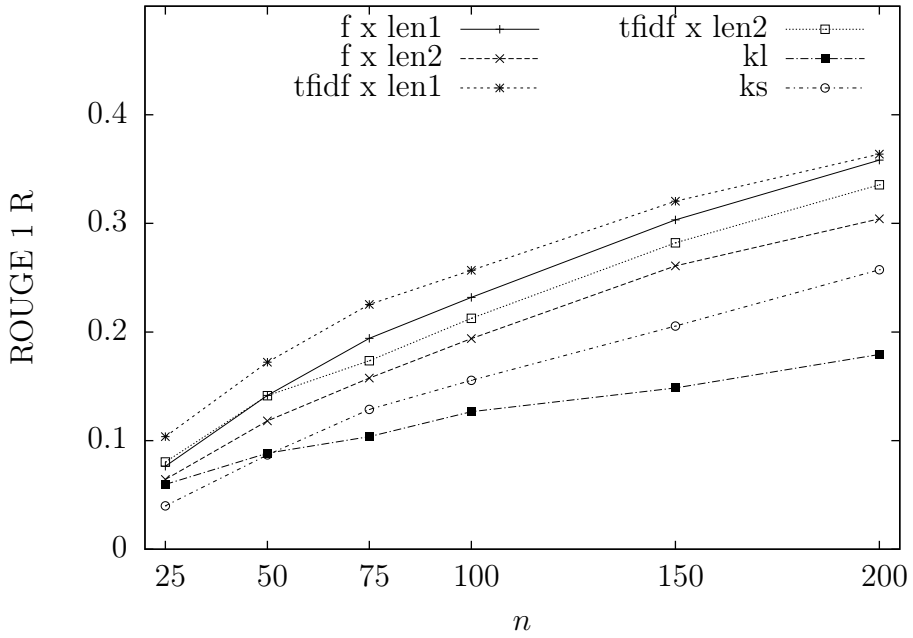



Figure 5.3: ROUGE-1 R scores using *n-kp*, the different key phrase weighting strategies, and an increasing number of phrases as summaries.

of including all phrases which typically results in 500 to 600 words. This "summary" contains every noun phrase that appears at least twice in the document. As ROUGE-1 R is a recall oriented measure, its value will be unnaturally high as it is very likely that this long list of noun phrases covers any noun phrase of the reference summaries. However, the results show that extractive summarization, in its current definition, seems to have reached the upper limit, which will be discussed in the next section. Following the trend in Tab. 5.3, the "tfidf x len2" ranking will be used in the following experiments.

### Results in Context

The previous sections focused on different aspects of the various systems. To conclude these experiments, the tuned systems are put in comparison and framed by the baseline and oracle results. The 400 word summaries are computed using $\lambda = 0.9$, the key phrases weighted using "tfidf x len2," and accounting for all key phrases, as suggested by the prior findings. Where applicable, the candidate sentences are pruned to 50 and the optimization time limit is 360 seconds. For the *n-kp* system, the top 100 key phrases form the textual summary. This number is a compromise between choosing the most relevant phrases but generating a rather short summary. The resulting number of words is typically less than 200 and thus much shorter than the target length.

Tab. 5.4 shows all systems in comparison. The first observation is that the scores differences between training and test set are insignificant, with the exception of the *max-r* oracle, which is due to the three human abstracts instead of a single one as

| | *train* | *test* | *all* | # | *signif. better than* |
|---|---|---|---|---|---|
| baseline1 | 0.12 | 0.12 | 0.12 | 1 | — |
| baseline2 | 0.13 | 0.11 | 0.12 | 2 | — |
| mmr/grd | 0.16 | 0.15 | 0.16 | 3 | 1,2 |
| mcd/ilp | 0.19 | 0.19 | 0.19 | 4 | 1,2,3 |
| mmr/ilp | 0.18 | 0.19 | 0.18 | 5 | 1,2,3 |
| concepts/ilp | **0.20** | **0.21** | **0.20** | 6 | 1,2,3,4,5 |
| max-r | 0.35 | 0.30 | 0.34 | 7 | all |
| *n-kp* ($n = 100$) | *0.20* | *0.21* | *0.20* | *8* | *1,2,3,4,5* |

Table 5.4: Comparison of ROUGE-1 scores for summaries of 400 words for manual transcriptions using "tfidf x len2" ranked phrases. Numbers are given for the training, test and complete set. The last column lists the system # that are significantly worse than the row system regarding the complete set.

| annotator | ROUGE-1 R | avg. length |
|---|---|---|
| 1 | 0.26 | 230 |
| 2 | 0.45 | 700 |
| 3 | 0.20 | 230 |

Table 5.5: ROUGE-1 R scores for the test set if comparing one human annotator with the remaining two.

for the training set. A further general observation is that the global models indeed outperform the greedy solutions, despite the pruning and possible early termination of the optimization. The concept based system is significantly better than all other extractive systems, in addition to its low computational effort. The *max-r* oracle result is however still far ahead. However, the oracle summary is not necessarily of high quality, but simply optimizes the n-gram overlap with the human abstracts. This raises the question how well a human annotator could do. Using the three reference summaries for each of the test meetings, a three-fold leave one annotator out experiment is conducted, evaluating the summaries of one annotator with regard to the remaining two. Tab. 5.5 shows the resulting ROUGE-1 R scores. The rather high recall of subject 2 is explained by the average length of the summaries, which is more than twice the length of the others. Extrapolating the values for 400 words, the *max-r* score of 0.34 seems realistic.

Using solely key phrases as a textual summary (*n-kp*) shows a surprisingly high score despite the rather short length of 165 words on average. If all key phrases were to be used, the ROUGE-1 R score even significantly exceeds the *max-r* oracle at an average length of 550 words per summary. This is again due to fact that ROUGE-1 R is recall based and the complete set of key phrases is basically the set of noun phrases that appear at least twice in the document, and that this set most likely covers all noun phrases of the reference summary. Similar as with the oracle, the higher score does not necessarily indicate a high quality summary: the key phrases

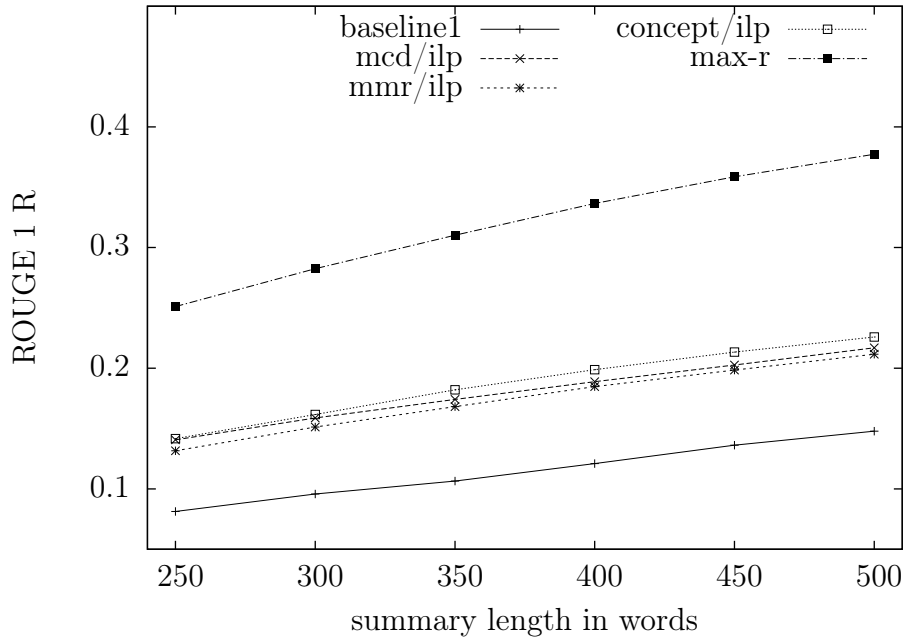



Figure 5.4: ROUGE-1 R scores of the *baseline1*, *mcd/ilp*, *mmr/ilp*, *concepts/ilp* and *max-r* system generating summaries of different lengths for all meetings using manual transcripts.

are a good means to describe the general topics and to point to important concepts, but are of questionable reliability without context. In general, the motivation of displaying the phrases instead of an extractive summary is to express similar or the same information in a more compact way.

### 5.5.2 Results for Different Summary Lengths

The previous experiments extensively analyzed the different systems and their parameters regarding the performance when generating fixed-length summaries. This section focuses on the effect of different summary lengths between 250 and 500 words on the summarization performance using the parameters found in the previous sections. The system *n-kp* is excluded from these experiments as it does not produce extractive summaries and due to the reasons stated above. Furthermore, it already achieves better ROUGE-1 R as the other systems at an average length of 165 words.

Fig. 5.4 shows the summarization performance of the three global models *mcd/ilp*, *mmr/ilp* and *concepts/ilp*, framed by the *baseline1* and *max-r* results. Intuitively, the recall generally increases with increasing summary length. The ranking of the systems as seen in Tab. 5.4 remains the same, regardless of the summary length. The *max-r* oracle performance remains far ahead of the automatic systems, which will be further discussed at the end of this chapter.

| | *train* | *test* | *all* | # | *signif. better than* |
|---|---|---|---|---|---|
| baseline1 | 0.09 | 0.09 | 0.09 | 1 | — |
| baseline2 | 0.09 | 0.09 | 0.09 | 2 | — |
| mmr/grd | 0.12 | 0.12 | 0.12 | 3 | 1,2 |
| mcd/ilp | 0.14 | 0.15 | 0.14 | 4 | 1,2,3 |
| mmr/ilp | 0.13 | 0.14 | 0.13 | 5 | 1,2,3 |
| concepts/ilp | **0.14** | **0.15** | **0.14** | 6 | 1,2,3,4,5 |
| max-r | 0.31 | 0.26 | 0.29 | 7 | all |
| *n-kp* ($n = 100$) | *0.14* | *0.15* | *0.14* | *8* | *1,2,3,4,5* |

Table 5.6: Comparison of ROUGE-1 R scores for summaries of 400 words for automatic transcriptions using "tfidf x len2" ranked phrases. Numbers are given for the training, test and complete set. The last column lists the system # that are significantly worse than the row system regarding the complete set.

### 5.5.3 Results using Automatic Transcripts

The experiments on fixed and variable summary lengths can also be conducted on automatic instead of manual transcripts. The automatic speech recognition output was provided by SRI International using their conversational telephone speech recognition system and has a word error rate of about 37% [Zhu 05].

Tab. 5.6 shows the summarization scores of all systems using automatic instead of manual transcripts. The overall decrease in score can be explained by the rather high word error rate– the automatic summaries are still compared to the human abstractive summaries. The system ranking in terms of performance remains about the same, with the concept based system showing the best performance of the extractive systems. Note that the decrease in oracle performance is about 5%, which is consistent with the performance of the other systems.

Fig. 5.5 shows the system performances for different summary lengths using automatic instead of manual transcriptions. As with the manual transcriptions, the overall system ranking remains the same. The performance gap between the automatic and *max-r* systems increased, which is explained by the transcription errors and their propagation to the ley phrase extraction and ranking.

### 5.5.4 Example Summaries

The following paragraphs show example summaries for a meeting and a lecture, both using the manual transcripts. The automatically extracted key phrases are typeset in bold, the extracted transcript are augmented by punctuation to increase the readability.

#### Meeting

The given example summaries of meeting *Bro023* include the manual abstract and automatic summaries of about 400 words, computed using the *concept/ilp* ("tfidf x len2," match all phrases) and oracle *max-r* systems.

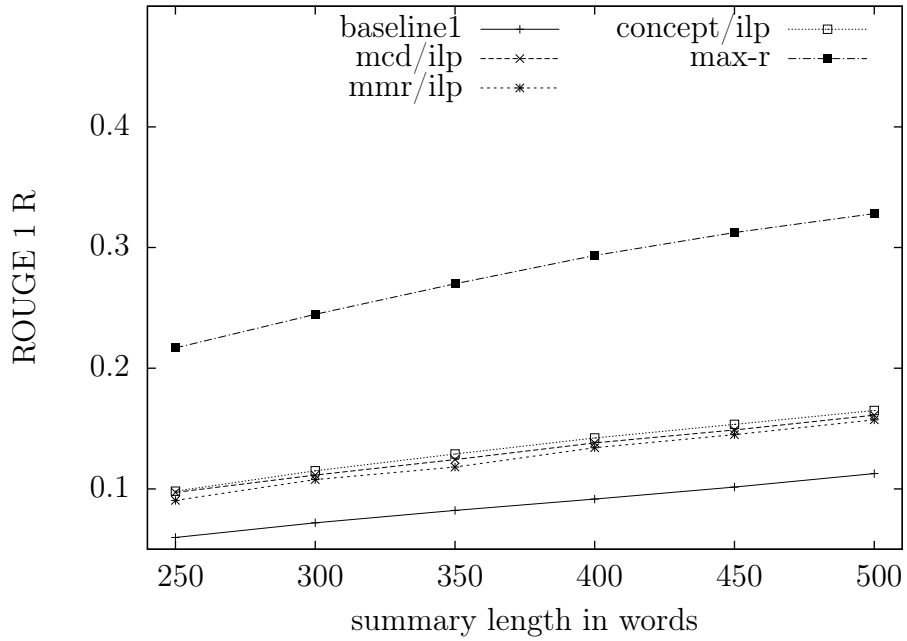



Figure 5.5: ROUGE-1 R scores of the *baseline1*, *mcd/ilp*, *mmr/ilp*, *concepts/ilp* and *max-r* system generating summaries of different lengths for all meetings using automatic transcriptions.

**Human Abstract** The ICSI Meeting Recorder Group of Berkeley met for the first time in two weeks. Group members reported their progress in the areas of **spectral subtraction**, **Wiener filtering** and **noise estimation**. They also discusses topics relating to the **rules** and preferences of the project they are working on, including single vs. **multiple passes**. A number of the group also took time to explain the basics of their approaches to the group. There are hopes that a **visitor** coming for three weeks, may lead to a **longer** [term] **collaboration**. The **visitor** works on **spectral subtraction**, so speaker me026 will make sure he talks to him. Speaker mn007 agreed, at me013's suggestion, to try his **noise compensation scheme** in compensation with the prior work on **spectral subtraction**. In implementing **smoothing** to the **spectral subtraction**, **latency** has been increased. While some feel this is nothing to worry about, others feel it is better to worry now, in case it turns out to be something to worry about. Speaker me026 has been experimenting with **spectral subtraction** using different data window sizes. One possible idea is to use increasing windows as more data becomes available. Speaker mn049 has been working on **Wiener filtering**, and testing with just the base system provides 30 **% improvement**. Using a second stage of filtering led to even more improvement. Speaker mn007 is working on **spectral subtraction**, still with minimal results. **Smoothing** seems to help, and implementing alongside the **neural net** should also be positive. He has also been working on **noise estimation** with an **energy** minima approach that does not require the **voice activity detector**.

**concept/ilp** Th– that is his **spectral subtraction group**. Just– there is just **inter word silences**. So if your second **pass** takes a **millisecond** who cares. Improves over

the **base line MFCC system**. Which is a **standard wiener filter**. That when I said 30 **% improvement** it is like **MFCC baseline**. Forty percent is the **high mismatch**. This is the **single stage wiener filter**. And the rest is like the LDA plu– and the on **line normalization** all remaining the same. Which is like **final filter** is acting on the input **noisy speech** rather than on the cleaned up. I mean that is like a **cheating method**. Which is the **channel** zero. So sf– is a **clean speech spectrum power spectrum**. And $n$ is the **noisy power spectrum**. You get an estimate of the **clean power spectrum**. And you will multiply that **noise spectrum times** some **constant** and subtract that. We just wanted to have a few **noise production compensation techniques**. Yeah I mean there is car– carmen is working on another, on the **vector taylor series**. So with a **bump** around one **kilohertz**. Yep there is all **sorts** of **deviations** from the **ideal** here. But you end up reducing some **neighboring frequency bins** in the average, right. Which is the **true estimate**. But instead of **double stage wiener filtering** it is this **smoothed spectral subtraction**. What is it the **France telecom system** uses. Th– the just **noise compensation technique** is a **variant** of **wiener filtering**. This is in the **mel frequency bands**. It could be seen as a f– a **smoothing** in the **frequency domain** because I used in **ad mel bands** in **addition** and then the other **phase** of **smoothing** in the time domain. That is the **musical noise**. If you do this in the **FFT bins** then you have **spots** of **energy** randomly distributing. So it is the **center recursion**. The mean **estimation** has some delay. Is not it the TCP **buffer** some. And **first frame** has a twenty **framed latency**. What is done is that these **minima** are computed based on **high resolution spectra**. In this **tile** appears like the **harmonics** if you have a **voiced sound**. I also implemented a sp– **spectral whitening idea**. Because during the **silence portions** which are below the **threshold** of **voice activity probability** w– you would have some kind of **dummy frame**. Well **Hans-Guenter** will be here next **week**.

**max-r** Th– that's his **spectral subtraction group**. Yeah we met him in **Amsterdam**. We didn't meet last week, Morgan. And that **increasing past** twelve **seconds** didn't seem to help. But other people didn't and had **multiple passes**. So if it turned out to be a **problem** that you didn't have enough speech because you need a **longer window** to do this processing th– n– one **tactic** is, you know, looking at the **larger system** and not just at the front end stuff is to take in the speech with some **simpler mechanism** or **shorter time mechanism**. I'm talking about **Italian**. This is the **single stage wiener filter**. That's that 's just a **rule**. The **Wiener filter** it's like you try to minimize. But it works reasonably well. It's all pretty related. And **spectral subtraction** is one approach to it. But y– but there you make different **approximations**. We just wanted to have a few **noise production compensation techniques**. Because after **subtraction** you can have **negative energies**. When you add the negative to the positive value. Basically that's this idea. But do you have numbers in terms of word error rates on **italian** so just so you have some sense of **reference**. It 's similar in the **smoothing**. So one would hope presumably that the **neural net** part of it would improve things further as they did before. In **fact** we had **visitors** here who did that. Why is that **delay** coming. I would worry about it a little. And first **frame** has a twenty **framed latency**. What is done is that these **minima** are computed based on **high resolution spectra** . Which is not the case if you rely on the **voice activity detector**. So I have to test it. But the **spectral subtraction scheme** that you reported on also re– requires a **noise estimate**. So I have like some **experiments**

running I don't have the results. I also implemented a sp– **spectral whitening idea**. What what point does the system stop recording. Yeah for this I think we can maybe try to train the **neural network** for **voice activity detection** on all the data that we have **including** all the **speechdat car** data.

**Lecture**

The following summary of lecture *PA06* was generated by the *concept/ilp* system, using the "tfidf x len2" ranking and considering all phrases of each utterance, as in the example above.

So welcome to the **Tuesday afternoon session pattern analysis, yesterday**. The **postal offices** to read the **addresses** and. Applied to **current systems** and **current problems**. And computes the principal **axis**. And this can be used for shape **representation** in medical **engineering** applications this can also be used for face **modeling** faces **human faces** or even for human **body shapes**. This is the problem of **overfitting**. That the variables you are estimating are discrete **categorial variable**. So the **difference** between regression and classification is in the one **case** we estimate real **valued parameters**. There are so called **relaxation** techniques where you do so like you know **classes** are represented by **real numbers** and then you have a **relaxation** algorithm where you more and more **force** the solution to the discrete **class numbers**. **Straight line** here into the **linear function** just our feature vector x. Lets say its our linear decision boundary. Specific **scenario** but basically you get a **signed distance**. We have defined the **ideal decision** function or **ideal decision** boundary. And you get a **closed form solution**. The L2 **norm** I get basically a **quadratic function** a quadratic **convex function**. Why the **hell** should I prefer the euclidean **norm**. Can be incorporate **prior knowledge** and **linear regression** so can we incorporate some priors into **linear regression**. The alpha vector here to have **unit length**. Thats a regularized **regression approach**. Gets an **additive component** that is governed by the **identity matrix**. **Terms** like this **regularizer** here I modify the **original estimator**. Introduced to **statistics** and in the pattern **recognition community**. I mean we're having now a very good understanding of the **relationship** between Bayesian classifiers, Gaussian **classifiers**, **nearest neighbor classifiers**. What was the **Mahalanobis distance**. We now know about **principal component** analysis. Few **comments** on the **literature**. Okay then my **web page** I told you, matrix **cook book**. Trevor **Hastie**, **Tibshirani** and **Jerome Friedman**, hey they are all three at at **Stanford** in the statistical. **Statistical department** and they wrote a book on the **elements** of **statistical learning** data **mining inference** and **prediction**. **Chair** of pattern recognition and check our our **publication list**, in **December** last **year** we had one **contribution** at the **international conference**. On pattern recognition in **Florida**, by the way that was the **trip** when they took my **Adidas** one shoes out of my **suitcase**. What **topics** are **reconsidering** in in **diploma** thesis and master **thesis projects**.

## 5.6 Discussion

The experiments show that the global formulations of MMR show a better performance than the original greedy formulation — but at a high computational cost. The

number of constraints in the ILP grow exponentially with the number of sentences, making the optimization problem infeasible even for short documents. Although the presented optimal MMR has more constraints then McDonald's formulation, the strict translation of the *max* operator (*cf.* Eq. 5.1) into ILP constraints (Eqs. 5.11–5.14) leads to a faster converging optimization. Interestingly, the performance of the global MMR-based systems is still better than the greedy algorithm despite the required pruning of the candidate sentences and the limitation of the computation time. Beside the computational complexity, the biggest disadvantage of MMR-based systems is the required calibration of the relevance parameter $\lambda$. Its effect on the summarization performance is reduced by the global optimization, but it remains a data-dependent variable that needs to be set application specific or to match the user's preferences. Compared to McDonald's formulation, the presented global MMR ILP leads to results that do not differ significantly in terms of ROUGE scores. This may be explained by the rather small number of actually valuable candidate utterances, but also by to the aforementioned pruning, that has a strong impact on the redundancy term of the optimization problem.

The global concept based system steps away from the sentence-based relevance and pair-wise redundancy measure by modeling summarization as a knapsack problem: the summary should contain as many unique concepts (here: key phrases) while satisfying a certain length constraint. This has two major benefits. First, the number of constraints is linear in the number the available concepts thus making it an easy optimization problem that can be solved in seconds. Second, the objective function only accounts for every contained concept only once, thus indirectly countering relevance. Although the summarization performance is only little (but significantly) better, the concept based model does not require a pre-pruning of the candidate utterances or a calibration of the relevance parameter, and should be favored over sentence based models.

Differently weighted key phrases did not show a significant effect on the summarization performance. This leads to the conclusion that any decent ranking will lead to a similar set of candidate sentences that are relevant for a summary. Minor differences in phrases and respective ranks diminish throughout the summarization process due to the limited choice of sentences to extract. However, this does not implicate that each key phrase ranking is similarly good with respect to the reference summaries. Summaries formed out of the top 100 key phrases of each ranking show a quite diverse performance in comparison to the human abstracts. In other words, not every strategy places actually valuable phrases on top, but helps to promote salient utterances in general.

With ROUGE being a recall-based measure, the system performance is partly governed by the summary length. Experiments sampling various summary lengths suggest that the relative performance of the different models remains similar. Using automatic instead of manual transcriptions naturally affects the performance as the errors from the automatic speech recognition are propagated to the key phrase extraction, ranking and subsequent summarization. However, the relative system performance remains, similar as with different length constraints.

The example summaries in the previous section illustrate the core problem of extractive summarization. Despite good ROUGE scores, utterances of spontaneous

speech are often meaningless if taken out of context, or show only one view of the conversation. The rather large score gap between the best automatic system and the *max-r* oracle can be explained by the oracle selecting meaningless utterances which still contribute to the ROUGE score. In the above meeting example, the oracle summary includes sentences such as "but it works reasonably well," "it's all pretty related," or "basically that's this idea," which are obviously meaningless without the context. The human annotators are able to compress way more information and context into a single sentence without reducing the readability. Longer extracts of spontaneous speech may convey the same information, but are a lot harder to read.

The example summary of the lecture *PA06* shows that these observations also hold for the lecture data. While some of the extracted utterances can provide valuable information, *e.g.*, "the L2 norm I get basically a quadratic function a quadratic convex function," others are meaningless without the context, *e.g.*, "applied to current systems and current problems" or "this is the problem of overfitting."

The experiments and example summaries suggest to draw the following conclusions. Extractive summaries are prone to be inconsistent by design unless an extra constraint is introduced to only allow to extract larger chunks of subsequent utterances to retain the context to a certain extent. This constraint however would result in the models selecting segments of the document that show a high combined salience, and might thus reduce the coverage of topics. The key phrases themselves are valid and useful, but should be presented in a more useful way than embedding them in utterances that might lack context. Furthermore, the experiments comparing only the human abstracts showed that even human have different taste in terms of length and detail of summaries. The next chapter elaborates this issue and introduces to interactive approaches to make key phrases and summarization more useful.

# Chapter 6

# Interactive Approaches to Video Lecture Assessment

## 6.1 The Gordian Knot of Summarization

The experiments in the previous chapter show that it is possible to generate extractive summaries which compare well against human abstracts — based on ROUGE results. However, the example summaries in Sec. 5.5.4 reveal what is wrong with extractive summarization of spontaneous speech. First, the summaries are built around the automatically extracted and weighted key phrases which might contain misleading or ill-weighted terms. This is not a problem *per se*, but the phrases and their weights, and thus the resulting summary, might not reflect the user's personal preferences. Second, the extracted utterances often lack context even in correct temporal order. Even if references were resolved, *e.g.*, names instead of pronouns, and spontaneous speech artifacts such as hesitations, repetitions or colloquial expressions were removed, there might just be a single utterance in the summary to reflect a complex topic within the document. This may be appropriate in some cases, but the user might also perceive that sentence as either too detailed or vague.

Given these thoughts, forming the ideal summary seems like solving the legendary Gordian Knot: only the user herself can come up with a perfect summary — after having studied all of the available material. Most of the current research on speech summarization focuses on improving certain aspects such as linguistic quality or coherence with respect to a small number of human, and thus subjective, summaries. This effort certainly helps to improve the methods and models in general, however, the focus is always on a "one-shot" system that produces a single, one-fits-all summary.

In this chapter, summarization is viewed from a different angle — its *use*. The original purpose of a summary is to inform someone about the important things, or, in other words, to provide one with everything considered important without going through all of the original material in order to save time. However, importance can only be defined by the user, and even that definition might change while reading a summary. For example, the user might read a generic summary and find topic $A$ interesting. She then reads that $A$ is linked to some topic $B$, which might not be part of the summary. Thus, she would prefer a summary which also covers topic $B$. This observation especially applies to video lectures: a user presented with the video

might first appreciate a general overview of the lecture before digging into certain topics which may induce further questions.

The aim of this thesis is to cut this Gordian Knot by stepping away from traditional extractive summarization. The user is not provided with a single summary that might or might not match her expectations, but instead with interactive tools that help to assess the information in the video. A key issue to back this change of paradigm is to show that the proposed interface is actually of better use.

The remainder of this chapter is organized as follows. First, aspects and criteria of user interface design are elaborated that are used to evaluate the quality of user interfaces. The following sections describe two interactive systems that help the user to extract all the salient parts from the data.

## 6.2 User Interface Evaluation

### 6.2.1 Three Major Aspects of User Interface Evaluation

User interface design and evaluation go hand in hand. *Usable* interfaces are defined in terms of learnability, efficiency, memorability, error reduction, and user satisfaction [Shne 04]. Evaluated is typically the *usability* as defined in ISO 9421-11[1]:

> **Effectiveness** — *Accuracy and completeness with which users achieve specified goals.*
>
> **Efficiency** — *Resources expected in relation to the accuracy and completeness with which users achieve goals.*
>
> **Satisfaction** — *Freedom of discomfort, and positive attitudes towards the use of the product.*

Depending on the maturity of the interface development process, the evaluation might focus on different usability aspects. For example in an early prototype state, the effectiveness is the first parameter to optimize. Later, closer to the release of the product, user satisfaction might be an important issue. Often, only the first two aspects are considered in evaluations because they refer to the function of the program and measurable facts instead of subjective opinions. But user satisfaction is a crucial feature: if the users dislike the interface they are unlikely to use it in the future. Although these three aspects seem to be closely related, the correlation among them is not necessarily high [Horn 07]: a program may be very effective, a rough work flow may result in low efficiency. But the subjective satisfaction may completely vary. The following paragraphs introduce to the three main evaluation criteria, and how they are typically evaluated.

**Satisfaction** The user *satisfaction* is traditionally measured using questionnaires that contain statements and questions to be answered using Likert scales [Like 32]. These psychometric scales often distinguish between five, seven or nine nominal values, for example shown in Fig. 6.1. An early attempt to standardize the user interface

[1]ISO 9241-11, 1998: Ergonomic Requirements for Office Work with Visual Display Terminals (VDTs): Part 11: Guidance on Usability

*The interface is intuitive to use.*
- ○ I strongly agree.
- ○ I agree.
- ○ I neither agree or disagree.
- ○ I disagree.
- ○ I strongly disagree.

Figure 6.1: Example of a statement and respective five point Likert scale.

evaluation with respect to user satisfaction is the commercially available Questionnaire for User Interface Satisfaction (QUIS) [Chin 88] which provides a large set of questions regarding different aspects of the interface, such as overall reaction, appearance on screen, learning or system capabilities. The questions are rated on a ten point scale with opposite adjectives at each end. For example: "System speed; too slow (0) — fast enough (9)," or "Performing tasks is straight forward; never (0) — always (9)."[2] Beside QUIS, a number of other catalogs have been established, for example the Perceived Usefulness and Ease of User (PUEU) [Davi 89], the Computer System Usability Questionnaire (CSUQ) [Lewi 95], the Purdue Usability Testing Questionnaire (PUTQ) [Lin 97], and the (free of use) USE Questionnaire [Lund 01]. The catalogs mainly differ in the wording and detail of the questions, the range of the (Likert) scales, and the aspects covered. Unfortunately, most of the questionnaires have to be commercially licensed and are tailored towards large-scale software products rather than short assessment of prototypes.

**Efficiency and Effectiveness** The *Efficiency* is often assessed by measuring the time required by the user to fulfill some required task, independently of the actual success rate. Other factors include the use of the resources, such as how many or how complex interactions were facilitated. Measuring the *effectiveness* of spoken document browsers is more complicated, and often tailored to the domain the browser is used in. A few examples are given in the next section.

### 6.2.2 Related Work on Browser Evaluation

For meetings, Wellner *et al.* [Well 05] suggest to "define the task of browsing a meeting recording as an attempt to find a maximum number of observations of interest in a minimum amount of time." Their Browser Evaluation Test (BET) is based on deciding on correct statements from a set of manually compiled alternatives, for example "the device should be red," or "the device should be green." However, these observations of interest refer to single points of occurrence and can thus be solved using a simple key word search, which is available with most browsing interfaces. A deeper understanding of or more extensive search within the spoken document is not required.

[2] An integer scale with a description at each end is not a Likert scale in its strict sense, but the resulting values can be used to compute correlations or agreement measures.

Stepping away from simple question answering, Kraaji and Post suggest to measure the effectiveness of a [meeting] browser by task based evaluation (TBE) [Kraa 06]. The users were provided with a series of group meetings and should finalize decisions based on decisions and thoughts. This scenario is not only very domain specific, but also depends on meeting content that allow the design of such tasks. This method is thus hard to apply in a general case.

Murray *et al.* find a compromise between the simple BET and TBE. Their meeting browser is evaluated using what they call a decision audit task [Murr 09]. The user is given the task of identifying decisions and the related arguments for and against them, which is manually validated by subjects familiar with the topic and meeting. Additionally, a questionnaire is used to determine how confident the user was in determining the answer.

For lectures, the vast majority of the proposed browsing interfaces remain unevaluated. A notable exception is presented by Huber *et al.*: the proposed mobile browser is evaluated in terms of usability as previously defined [Hube 10]. The users are provided with either of two browsers, and solve visual (slide-based) and textual fact finding tasks. The success rate and elapsed time relate to the effectiveness and efficiency. The user satisfaction is determined using a questionnaire.

In summary, there are two broad categories of user evaluations of efficiency and effectiveness, both usually followed by a questionnaire to survey the user satisfaction.

**Comprehension Task** Given the browser and a spoken document, the user is asked to answer certain questions about facts and their context. While this is possibly the best way to compare two or more interfaces in terms of their usability, it is possibly problematic for (technical) lectures. If the questions are too easy they may be answered based on prior knowledge or common sense. If the questions are too hard the user may not be able to answer properly, because the topic or question was not understood. Also, if the questions may be answered with a freely formulated text, the assessment of the effectiveness is a time-consuming manual task, that has the same pitfalls as the evaluation of summaries.

**Localization Task** Given the browser and a spoken document, the user is asked to mark regions that contain or support certain facts or statements. On the good side, the localization task is rather independent of prior knowledge. On the bad side however, the efficiency and effectiveness will be very high if a key word search or similar function is available, but rather low otherwise, which makes finding occurrences of facts a task too obvious for a reliable evaluation.

The following sections introduce to two user interfaces, one for summarization and one to visualize key phrases, and their user evaluations.

## 6.3 Interactive Summarization

### 6.3.1 A User Study

Thinking straight forward, the easiest way for the user to interact with the summarization system is to guide the summarization process. The key phrase based models

| | F | R | P |
|---|---|---|---|
| *baseline1* | 0.15 | 0.13 | 0.20 |
| *mmr/greedy* auto | 0.20 | 0.15 | 0.21 |
| *mmr/greedy* refined | **0.21** | **0.20** | **0.25** |
| *max-r* | 0.31 | 0.30 | 0.24 |

Table 6.1: ROUGE-1 (F)-measure, (R)ecall and (P)recision of the automatic summaries using automatic (*auto*) and manually refined (*refined*) key phrases. Summary length is 5% of the respective meeting word count.

presented in the previous chapter provide a natural way to do so: the idea is that better key phrases lead to a better summary. In a small user study on the test set of the ICSIMC, three human subjects were asked to refine a list of 50 automatically extracted key phrases. After reading the human meeting abstracts, they were asked to remove useless items from the automatically extracted key phrase list such as "thing," or "planner" in presence of "action planner." The automatic summaries with a word count of about 5% of the words of the original meeting were generated using the *mmr/greedy* (*cf.* Sec. 5.5) system and the ROUGE-1 scores are framed by the results of the *baseline1* and *max-r* (oracle) system (*cf.* Sec. 5.4.3).

The results in Tab. 6.1 confirm that better key phrases indeed lead to better summaries in terms of ROUGE-1[3]. Although the summaries are compared to three human abstracts (by different annotators) and were no further evaluated by the key phrase annotators, the result suggests that the user should be able to control the key phrases to guide the summarization process.

This experiment is similar to a recent study on user guided text summarization. Lin *et al.* propose to guide the iterative MMR algorithm by letting the user decide which sentence to extract next, given a ranked set of alternatives [Lin 10]. This way, the user is the final authority to relevance and redundancy for each step, instead of the user refining the relevance measure prior to the regular MMR. Similar to the results in Tab. 6.1, the authors could show that the human guided summaries were of higher quality.

The possibly first attempt to interactive speech summarization (for meetings) is documented in [Mies 07]. The user can give salience feedback for extracted utterances, which allows to retrain supervised systems systems and adapt the weights of unsupervised systems. The proposed summarization system was however not evaluated.

### 6.3.2 Interface Description

The fact that human refined key phrases can be used to compute more reliable summaries suggests to integrate the user in the summarization process. By giving the user control over the key phrase weights, the resulting summary is more likely to match the users' expectation.

[3]Further details of the experimental setup can be found in [Ried 08a]

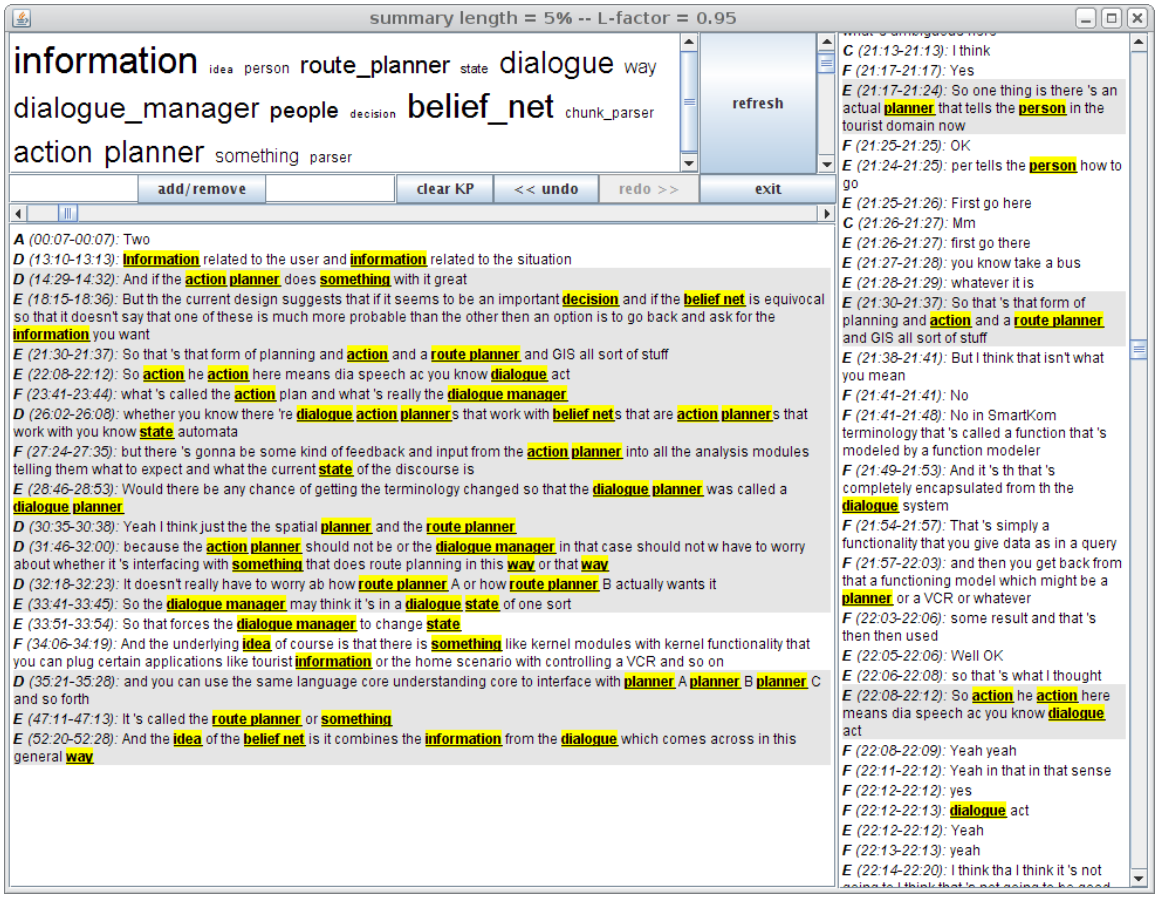



Figure 6.2: Screenshot of the implemented interactive summarization tool. The top left field contains the key phrases with the size representing the weight. The bottom left field lists the extracted sentences; the right field contains the complete transcript. Key phrases are marked yellow, utterances included in previous summaries are shaded.

A screenshot of the implemented interactive summarization tool is shown in Fig. 6.2. The top left area displays an initial set of automatically extracted key phrases. Their weight is expressed in their font size. The bottom left area displays the latest summary, the right column displays the complete transcript of the meeting or lecture. Key phrases are highlighted (underlined and yellow) and sentences which appeared in previous summaries are shaded gray.

### 6.3.3 User Interactions

The interface allows the following user interactions:

- Use the text field below the key phrase area to add or remove key phrases. Phrases that are added but do not appear in the document are rendered in red color in the key phrase area.

```
Mon Feb 06 10:18:41 CET 2012 UAChangeKeywordWeight (do) planner => 30.0
Mon Feb 06 10:18:52 CET 2012 CALL mmr_baseline(l=0.95)
Mon Feb 06 10:19:47 CET 2012 UAChangeKeywordWeight (do) route_planner => 17.0
Mon Feb 06 10:20:39 CET 2012 UAToggleKeyword (do) kind
Mon Feb 06 10:21:03 CET 2012 UAChangeKeywordWeight (do) chunk_parser => 21.0
```

Figure 6.3: Example user interaction protocol of the meeting summarization interface.

- Change the weight of a key phrase by pointing on a phrase and turning the mouse wheel up (more weight) or down (less weight).
- Control the summarization parameters length and relevance factor $\lambda$ using the sliders above the summary area and right of the "refresh" button. Refresh the summary using the current parameters, key phrases and respective weights.
- Undo or redo any action using the respective buttons below the key phrase area.

Beside the actual summary, the user can make use of the yellow phrase makers to identify hot spots in the document. The gray shading of sentences that were already included in a previous summary help to concentrate on new information between subsequent summary generations. The above user interactions are logged for further analysis. Fig. 6.3 shows an example interaction protocol.

This interface is not further evaluated. Beside the fact that it is an early prototype, the summarization tool does not resemble a full meeting browser but addresses the specific problem of generating custom tailored summaries.

## 6.4 Interactive Key Phrase Visualization

The example summaries in the previous chapter show that the main problem with extractive summarization of spontaneous speech is the little readability (or linguistic quality) of concatenated utterances. The extracted utterances often lack context, and may contain errors in case speech recognition was used. The underlying key phrases, however, already convey most of the information and give clues about the topics. Thus, a graphical representation that puts phrases in context and shows them on a time line should allow the user to quickly get the gist of the video.

### 6.4.1 Visualization

A simple, straight forward visualization of phrase occurrences is shown in Fig. 6.4. The progress bar of the video player is augmented by an indicator field that shows a red bar for each time a given phrase occurs. The controls can then be used to navigate within the video. Although it is possible to show multiple phrases in either a single field using different colors, or in multiple, stacked fields, this visualization quickly gets confusing or too roomy.

A more intuitive visualization scheme is based on *StreamGraphs* [Byro 08], an extension of *ThemeRiver* [Havr 02], which was originally used to display thematic changes over time in large scale textual news corpora. The example in Fig. 6.5 shows

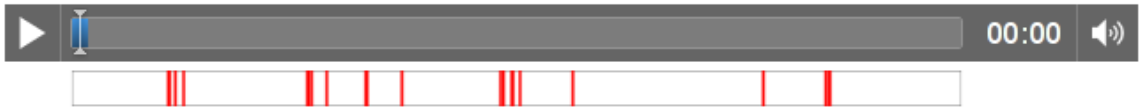



Figure 6.4: A basic indicator bar showing the occurrences of the phrase "decision boundary" in the lecture PA06 along the time bar.

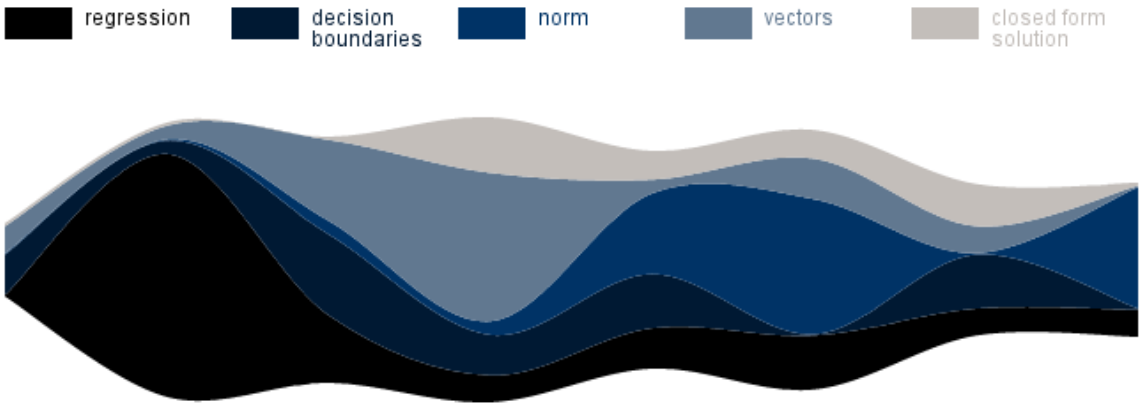



Figure 6.5: Example stream graph for lecture PA06. The X axis represents the time line, the width of the color coded stream indicates the significance of the phrase at the given time.

a colored stream for each of the five phrases, where the X axis represents the time line. The width of the stream indicates the significance of the respective phrase, which will be introduced later on. Similar to the occurrence indicator field, the width stream graph can be scaled to match the player controls, to use it to navigate through the video.

Stream graphs consist of several individual shapes that are stacked on top of each other using certain rules, where each of the shapes is formed using a cubic spline interpolation. Additional constraints ensure that the spline function remains positive at all times in order to have a positive stream width. The required mathematical formulations are described in the original work by Havre *et al.* [Havr 02]. Details on spline interpolations can be found for example in [Wahb 90].

The data points to be interpolated by the spline function are the sampled phrase significance or dominance values, which are determined by a simple heuristic. The document is split in equally sized segments, for example 5 minutes. Then, for each phrase, the number of occurrences in each segment is counted and used as a significance value, as shown in Fig. 6.6.

The interesting part is how to stack the splines to form an aesthetic graph. In general, stacking graphs is achieved by adding the previous, underlying graphs to the current one. Or more precisely

$$g_n(t) = g_0(t) + \sum_{i=1}^{n} f_i(t) \tag{6.1}$$

| | *whole recording* | | | | | |
|---|---|---|---|---|---|---|
| *segments* | | | | | | |
| *phrase 1* | 0 | 2 | 4 | 0 | 0 | 2 |
| *phrase 2* | 0 | 0 | 1 | 2 | 2 | 1 |

Figure 6.6: The whole recording is split in equally sized segments to count the per-segment phrase occurrences.

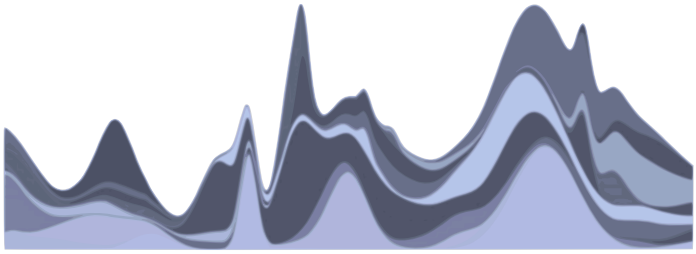


Figure 6.7: Stacked graphs with flat baseline ($g_0 = 0$). Image taken from [Byro 08].

where $g_0(t)$ is a baseline, $g_n(t)$ is the stacked version of $f_n(t)$ on top of $f_1(t), \ldots, f_{n-1}(t)$. Fig. 6.7 shows an example for a fixed baseline $g_0(t) = 0$. Depending on the spline values, the resulting graph tends to form spikes, making it rather hard to read. Havre *et al.* suggest to center the graph around a baseline, setting

$$g_0(t) = -\frac{1}{2} \sum_{i=1}^{N} f_i(t) \tag{6.2}$$

where $N$ is the overall number of splines to stack [Havr 02]. Fig. 6.8 shows the resulting symmetric stream graph. Byron *et al.* additionally consider "wiggle," or curliness, of the streams by looking at the sum of squares of the graphs' first derivatives [Byro 08]. The baseline function is chosen so that the

$$\text{wiggle}\,(g_0) = \sum_{t} \sum_{i=0}^{N} \left(g_i'(t)\right)^2 = \sum_{t} \sum_{i=0}^{N} \left( g_0'(t) + \sum_{j=1}^{N} f_j'(t) \right)^2 \tag{6.3}$$

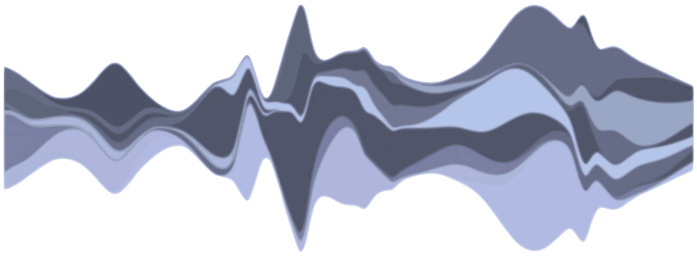


Figure 6.8: Stacked graphs with center baseline resulting in a symmetric shape. Image taken from [Byro 08].

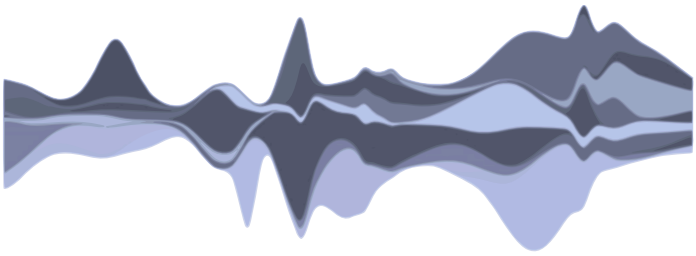


Figure 6.9: Stacked graphs, minimum "wiggle;" image taken from [Byro 08].

is minimized. The resulting smooth graph is shown in Fig. 6.9. Byron *et al.* describe a simple solution to numerically integrate $g'_0$ [Byro 08].

### 6.4.2 Interface Description

The interface of the lecture browser is shown in Fig. 6.10. The far right is a list of available key phrases, in a descending ranking. The top-left shows the video player with controls. Just below the canvas, the red bars indicate the occurrences of the currently selected phrase (blue highlight, far left). Further below is the stream graph that contains one colored stream for every selected phrase (column between video and available phrases list). The stream graph axes are the time (left to right, beginning at zero) and the current phrase dominance. Each stream is color coded regarding the index just above it. The bigger the stream is at a given time, the more dominant is the respective phrase. The stream graph is thus not a plain occurrence indicator, but combines the temporal occurrence with a notion of importance in comparison to the other displayed phrases. The comparison of two or more streams can shed some light on the context and relation of phrases. For example, if a phrase never occurs together with another one, they are unlikely to be related. The other way around, phrases that regularly co-occur may be related, which helps the user quickly assess certain topics. The interface is rounded off with a phrase search, found below the selected phrases list.

### 6.4.3 User Interactions

The possible user interactions are split in two groups, where the first relates to the pure browser functionality and the second to data collection for future research. The browser related interactions include:

- Alter the selection of phrases that are displayed in the stream graph by dragging a phrase from the right into the left for inclusion, and vice versa for exclusion.
- Select a single phrase to update the phrase occurrence indicator bar just between the video player and the stream graph.
- Click the occurrence indicator bar to set the video playback position to the requested position regarding the horizontal click position.

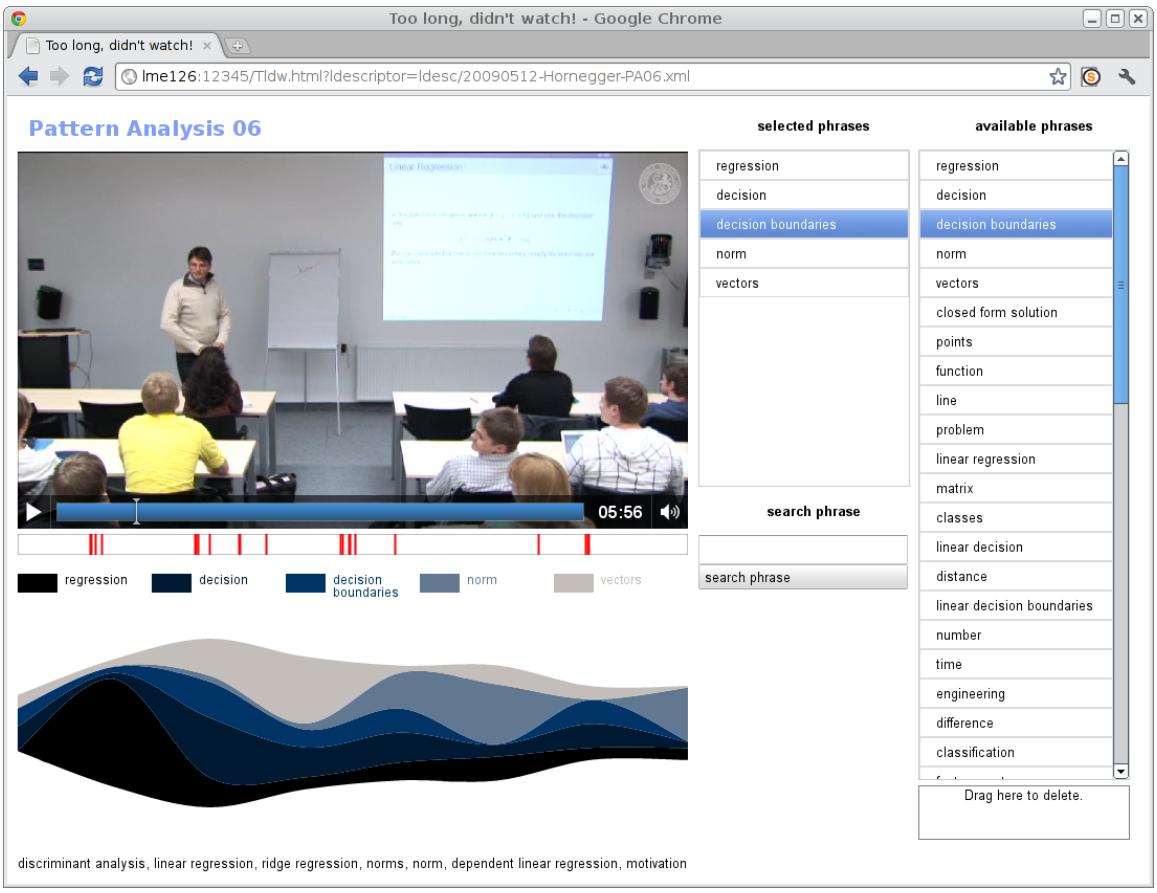



Figure 6.10: Screenshot of the interactive lecture browser. The video is on the top left, followed by selected phrase occurrence indicator (red bars) and the stream graph. The right two columns list the selected (left) and available (right) key phrases.

```
USERACTION 1330624252532 0 GraphClick exact=977.449 corrected=986.521
USERACTION 1330624354422 1 GraphClick exact=1495.150 corrected=1496.969
USERACTION 1330624952584 2 ExcludePhraseFromDisplay 466
USERACTION 1330624954175 3 ExcludePhraseFromDisplay 194
USERACTION 1330624954970 4 SelectPhraseForDisplay 230 3
USERACTION 1330624958178 5 SelectPhraseForDisplay 5 4
USERACTION 1330625012444 6 GraphClick exact=455.885 corrected=469.298
USERACTION 1330625188857 7 ModifyRank 328 5
USERACTION 1330629442342 8 DeletePhrase 133
```

Figure 6.11: Example user interaction protocol of the lecture browser interface.

- Move the mouse over a phrase in the stream graph legend or a stream to highlight both the stream and the related legend element.
- Click on a stream to set the current playback position a few seconds prior to the closest occurrence of the respective phrase regarding the horizontal click position.

For future research on user-tailored key phrase rankings, the following interactions are available:

- Alter the key phrase ranking by re-ordering phrases in the right list using drag-and-drop.
- Delete a phrase by dragging it to the designated area below the right list.
- Search a phrase using the input field below the left list. Any hit will be included in the display and put on top of the right list.

As with the summarization tool, any user interaction is logged for future analysis. Fig. 6.11 shows an example protocol. For all interactions, the time and running number are stored together with the interaction details. For the stream graph clicks, both actual horizontal click position and the phrase occurrence driven correction are recorded.

### 6.4.4 Implementation Details

The browser interface is implemented as a client-server application, as shown in Fig. 6.12. The server provides the data, namely the video, transcript and initially weighted key phrases. The user, on the other side, uses the interface, and the usage data is transmitted to the server using asynchronous remote procedure calls, *i.e.*, the client calls a function that is actually executed on the server — hidden to the client. The following paragraphs introduce to the technology and toolkits that are used for the prototype.

The first step in the processing pipeline is to apply speech recognition to the audio track of the video. The KALDI toolkit training and decoding routines are described in Chapter 3. The Apache 2.0 license would also allow commercial use.

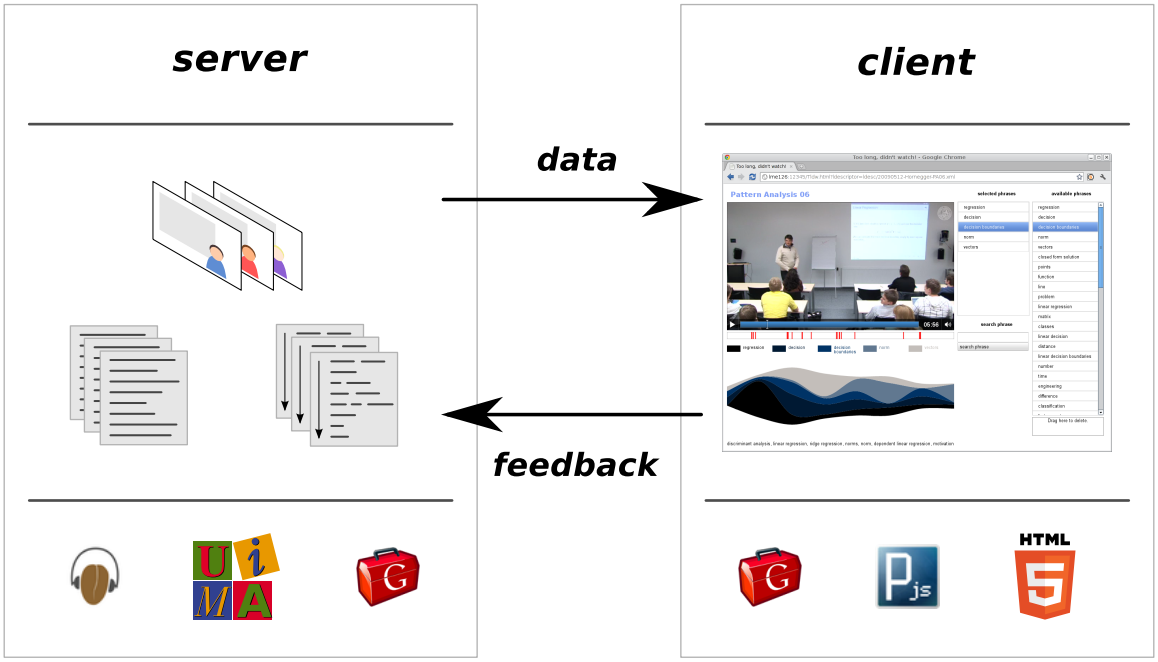



Figure 6.12: Overview of the client-server architecture of the lecture browser system.

The preprocessing, candidate selection and key phrase ranking algorithms are implemented in the Unstructured Information Management Applications (UIMA) framework[a] [Grop 10]. UIMA provides a framework where small, possibly independent modules can be easily put together to an extensive processing pipeline. The used JAVA implementation of the framework is platform independent and supports the automatic parallelization of independent steps such as PoS tagging and lemmatization.

[a] `http://uima.apache.org/`

The Google Web Toolkit[a] (GWT) plays a central role in the client server application. On the client side, the GWT provides methods to implement the click, drag, and drop functionality. The beauty of the GWT is that it allows the developer to work as with regular JAVA components such as Objects, ListBoxes or sorted lists, however, the application is compiled into JAVASCRIPT to be executed by the client browser.
Furthermore, the GWT provides a framework to seamlessly start synchronous, *i.e.*, blocking, and asynchronous routines on the server side. Beside the initial data retrieval, this mechanism allows to send feedback to the server. These feedback calls are modeled as asynchronous calls. That way, the interface does not freeze, if the server is temporarily unreachable. On the server side, the GWT is used to model the receiver of the clients' asynchronous calls.

[a] `http://code.google.com/webtoolkit/`

`Processing.js`[a] is a JAVASCRIPT port of the `Processing` toolkit. It provides the routines to compute and update the stream graph visualization. The `StreamGraph.js` package[b], and has been slightly adapted to allow to update the stream labels and data depending on the user interactions.

[a] `http://processingjs.org/`
[b] `https://github.com/jsundram/streamgraph.js`, retrieved Mar. 2, 2012.


The video, and to some extent the drag-and-drop, can be implemented using the hypertext markup language version five (HTML5). HTML5 introduced many convenient browser elements such as an audio/video player using the `<video>` tag, and a drawable canvas using the `<canvas>` tag. In combination with Cascading Style Sheets rev. 3 (CSS3) and JAVASCRIPT, websites can be built that have the look and feel of regular desktop applications, including key board short cuts and mouse inputs such as clicks and drag-and-drop.

### 6.4.5 Evaluation

The evaluation is performed on the lecture *PA06*, extending the experiments on key phrase extraction and summarization. It is split in two parts, namely an assessment task and a post-use questionnaire. To get an idea of how useful the tool actually is, the subjects are split in two groups. The *control* group performs the task with just the video, the *test* group using the proposed tool. Naturally, the post-use questionnaire is only given to the *test* group that actually uses the tool. Here, a group of ten computer science graduate students was split in five students for each *control* and *test* group.

#### Task Description

The subject is provided with a list of three briefly described topics that are covered in the lecture, and is asked to mark those three minute segments on a time line that correspond to these topics. The task is designed in a way that it can be solved by watching the whole video without actually comprehending the context. A simple key words search, as possible with the tool, will however lead to false hits, as the phrases occur in several topical segments as well as independent from them. Fig. 6.13 shows the task description and the paper form to mark the relevant segments.

#### Post-use Questionnaire

After the subjects finished the above described task, they are asked to complete a questionnaire regarding their impressions of the lecture browser. The statements are divided in four categories and are to be rated on five point Likert scales that read "strongly agree, agree, neutral, disagree, strongly disagree." Fig. 6.14 shows a copy of the questionnaire. The statements are selected from the USE [Lund 01] and PUTQ [Lin 97] catalogs, and slightly adapted to match the lecture browser scenario. Although some of the questionnaire items sound rather similar or redundant they

| *group* | *tp* | *tn* | *fp* | *fn* | *R* | *P* | *F* | *t* |
|---|---|---|---|---|---|---|---|---|
| control | 34 | 144 | 21 | 11 | 0.76 | 0.62 | 0.68 | 30 |
| test | 34 | 151 | 14 | 13 | 0.68 | 0.71 | 0.69 | 21 |

Table 6.2: Task performance of the two groups for detecting the salient segments within the video; *tp* and *tn* indicate true positive and negative decisions, *i.e.*, marking a segment as in the reference, *fp* and *fn* indicate the false positive and negative decisions, *i.e.*, marking a segment other than in the reference. $R$, $P$ and $F$ are the recall, precision and F-measure based on the previous quantities; the average time $t$ required to solve the task is measured in minutes.

aim for distinct aspects. An interesting question that reflects the overall perceived value of the interface regards the willingness to pay for it.

### Results

Each task sheet is compared against a reference solution by a person familiar with the topic, lecture series and particular class, using both video and slide material. Both control and test group show about the same performance in terms of detecting the relevant segments for each topic. Tab. 6.2 shows a detailed analysis of correct (true positive/negative) and false (false positive/negative) decisions and the resulting performance measures recall, precision and F-measure. The control group has a higher recall in detecting the relevant segments, but the lower precision indicates that the subjects tend to be overly sensitive. Similarly, the test group has a slightly lower recall at a higher precision, leading to almost identical F-measure values for both groups.

The interesting observation is the time required on average by each subject. The overall video duration is 41 minutes, which can be considered an upper bound for the task. While members of the control group used on average 30 minutes to complete the task, the members of the test group were able to achieve about the same result using only about 21 minutes on average, *i.e.*, the task could be performed about 29% faster when using the interface.

Tab. 6.3 shows the average ratings on the post use questionnaire after quantizing the attributes "strongly agree" to "strongly disagree" as 1 to 5. Although the users had a neutral attitude towards whether the tool to increased their productivity or efficiency, most subjects emphasized its usefulness to work with video lectures and its value as supplemental study material. The positive feedback regarding the ease of use and satisfaction statements confirm that the interface design and possible interactions are intuitive and easy to learn. The rather negative response to a possible commercial version is most likely due to web 2.0 zeitgeist that favors free access to information and software. For similar reasons, even large-scale web platforms for video hosting such as YouTube[4] or Vimeo[5] rely on secondary fund-raising using advertisements or separate premium features only available to paying customers. The key phrases are

[4] `http://www.youtube.com`
[5] `http://www.vimeo.com`

# Assessment Task

Participant No. _____

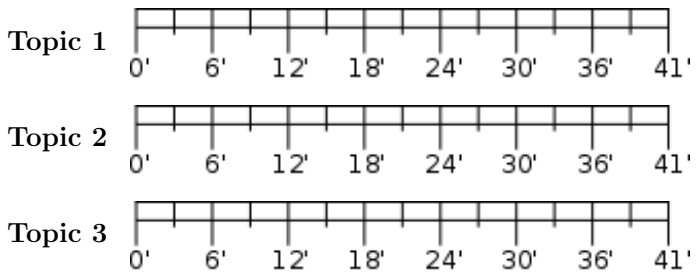
Time In: _____

Time Out: _____

Thank you for participating in this user survey for the LME Lecture Browser. The presented video lecture covers three main topics:

1. Difference between regression and classification; relation of regression line and decision boundary.
2. Introduction to *linear* regression, definition of the model, and derivation of the closed form solution.
3. Introduction to *ridge* regression, difference to linear regression.

Your task is to identify those three minute segments of the video that support each of these three topics, and mark them in the respective time lines below. The topics may overlap or re-occur throughout the lecture.




*Thank you!*

Figure 6.13: Task description and form used in the lecture browser evaluation.

# Questionnaire

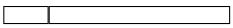
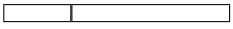
Participant No. _____

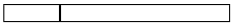
Thank you for participating in this user survey for the LME Lecture Browser. Please indicate your opinion of the browser by judging the following statements. Your feedback will help us to improve the browser, which is called "it" in the following.

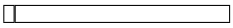
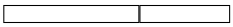
| | *strongly agree* | *agree* | *neutral* | *disagree* | *strongly disagree* |
|---|---|---|---|---|---|
| ***Perceived Usefulness*** | | | | | |
| Using it in my studies would enable me to find important facts quickly. | □ | □ | □ | □ | □ |
| Using it in my studies would increase my productivity. | □ | □ | □ | □ | □ |
| Using it in my studies would enhance my effectiveness. | □ | □ | □ | □ | □ |
| Using it would make it easier to work with the video lectures. | □ | □ | □ | □ | □ |
| I would find it useful in my studies. | □ | □ | □ | □ | □ |
| ***Ease of Use*** | | | | | |
| It is easy to use. | □ | □ | □ | □ | □ |
| It is intuitive to use. | □ | □ | □ | □ | □ |
| I learned to use it quickly. | □ | □ | □ | □ | □ |
| ***Satisfaction*** | | | | | |
| I like the interface. | □ | □ | □ | □ | □ |
| I would recommend it to a friend. | □ | □ | □ | □ | □ |
| I would pay for it. | □ | □ | □ | □ | □ |
| ***Browser Features*** | | | | | |
| The key phrases are helpful. | □ | □ | □ | □ | □ |
| The key phrases are accurate. | □ | □ | □ | □ | □ |
| The phrase visualization is helpful. | □ | □ | □ | □ | □ |
| The phrase visualization is accurate. | □ | □ | □ | □ | □ |
| Phrases and visualization give a good overview of the video lecture. | □ | □ | □ | □ | □ |

*Thank you!*

Figure 6.14: Post-use questionnaire used in the lecture browser evaluation.

| *Perceived Usefulness* | 1 | 5 |
|---|---|---|
| Using it in my studies would enable me to find important facts quickly. |  | |
| Using it in my studies would increase my productivity. |  | |
| Using it in my studies would enhance my effectiveness. |  | |
| Using it would make it easier to work with the video lectures. |  | |
| I would find it useful in my studies. |  | |

| *Ease of Use* | 1 | 5 |
|---|---|---|
| It is easy to use. |  | |
| It is intuitive to use. |  | |
| I learned to use it quickly. |  | |

| *Satisfaction* | 1 | 5 |
|---|---|---|
| I like the interface. |  | |
| I would recommend it to a friend. |  | |
| I would pay for it. |  | |

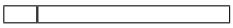
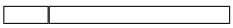
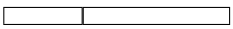
| *Browser Features* | 1 | 5 |
|---|---|---|
| The key phrases are helpful. |  | |
| The key phrases are accurate. |  | |
| The phrase visualization is helpful. |  | |
| The phrase visualization is accurate. |  | |
| Phrases and visualization give a good overview of the video lecture. |  | |

Table 6.3: Average human ratings for the questionnaire statements quantizing "strongly agree" through "strongly disagree" as 1 to 5.

found to be both helpful and accurate, and to give, together with the visualization, a good overview of the video lecture. The visualization itself seems, however, less helpful and accurate, leaving room for further improvements.

## 6.5 Summary

The intention of the user interfaces presented in this chapter is to step away from traditional summarization or key phrase presentation where a system produces a single-shot summary or key phrase list that might or might not satisfy the user. The first interface allowed the user to re-define or re-weight the key phrases that are used to generate an extractive summary. Although not thoroughly evaluated, the experiments with manually refined key phrases show that better phrases lead to better summaries in terms of ROUGE scores. The presented summarization tool allows the user to not only interact with the summarization algorithm, but generate multiple summaries with different focus. The gray-shading of utterances that were already presented in earlier summaries helps the user to concentrate on unseen utterances.

The experiments with the key phrases and the interactive summarization sparked the idea that the key phrases themselves might already provide sufficient information despite the lack of context. Furthermore, the extracts of spontaneous speech tend to be unstructured and hard to read. Thus, instead of using the phrases go generate a summary, the phrases are used as a navigation aid for the underlying document, be it an audio or video recording. That way, the phrases can be embedded in the actual context, without the possible pitfall of selecting the wrong or wrongly recognized utterances. The presented user evaluation of the interface confirmed that such a navigation aid is indeed useful: the participants were able to perform a certain localization task about 29% faster and at about the same success rate as participants provided solely with the raw video.

The experience with the interactive browser and the positive user feedback encourages to extend the work on visualization and interaction in order to provide a *useful* tool to assess the information content of spoken documents.

# Chapter 7

# Outlook

The experiments presented in this thesis focus on the four steps that are joined in the interactive tools presented in the previous chapter. The following paragraphs list suggestions how to improve or extend each of these steps, that are automatic speech recognition, key phrase extraction and ranking, and summarization. A final paragraph lists ideas on how to extend the interactive lecture browser to enhance the user interface and to improve its acceptance by the users.

**Automatic Speech Recognition** The two-level tree based multi-codebook semi-continuous system showed competitive numbers despite the relatively small number of Gaussian components. However, the subspace Gaussian mixture model (SGMM) based system is clearly superior in terms of modeling potential. A combination of the multi-codebook architecture with the SGMM framework will reduce the computational complexity while maintaining the modeling potential. The MLLT, as implemented for the continuous and SGMM models, needs to be modified to work within the multi-codebook architecture. The transformations can be implemented globally, codebook-specific, or based on acoustic similarities of the codebooks. Beside improvements to the acoustic model training, improving the capabilities of the recognition systems in terms of words, pronunciation alternatives, and language model are required to apply the system to lectures of other domains. For technical lectures, both correct and wrong pronunciations of technical or domain-specific words need to be integrated into the lexicon to increase the robustness regarding common pronunciation errors. If external sources such as presentation slides or related textbooks are available, the lexicon and language model need to be augmented and adapted to increase the recognition performance. The use of prosodic features instead of just speech pauses can improve the segmentation of the lectures. The resulting speech segments are expected to be more natural, *i.e.*, preserving the actual phrase boundaries. This helps not only the speech recognition by a likely better fit of the language model, but benefits all tasks that depend on the segmentation and transcription.

**Key Phrase Extraction and Ranking** The presented algorithms in Chapter 4 are based on the plain transcription, *i.e.*, the best word chain obtained by a manual or automatic transcription. To compensate for errors in the automatic transcription, the key phrases can be extracted from an n-best list that contains the best $n$

hypotheses, or preferably from a word lattice that shows the various hypotheses in form of a graph. The candidate selection implemented in this thesis is based on a regular expression on top of part-of-speech tags. A more flexible solution is to apply a statistical shallow parser that identifies noun phrases on top of the above-mentioned word lattices. Finally, the ranking strategies that are used in this thesis are unsupervised, or make only little use of prior (external) knowledge. Supervised methods can be used to generate user-specific rankings based on prior feedback collected with the lecture browser interface. They can be categorized in point-wise, pair-wise and list-wise models, with typically increasing complexity. Similar to the unsupervised ranking, supervised point-wise ranking algorithms are typically based on a regression model that predicts a numeric salience value for each key phrase. Pair-wise models are used to determine the ranking based on a sequence of binary decisions whether or not one phrase is more salient than the other. List-wise ranking strategies produce the complete ranking based instead of individual per-phrase decisions and thus require a training that consists of a set of ranked lists. Prosodic features can enhance the performance of both candidate extraction and subsequent ranking by identifying phrase boundaries or stressed words or phrases.

**Summarization** In text summarization, the automatic compression of sentences, *i.e.*, the removal of superfluous information or redundant segments, helps to increase the summarization performance. A good way to incorporate this into the global concept based speech summarization system is to generate several alternative versions of a sentence where each version has a possibly different length or covers different concepts. Additional constraints in the ILP are required to prevent a simultaneous presence of different versions of the same utterance. An improvement specific to speech summarization is to, on the one hand, determine a proper utterance segmentation to avoid the extraction of certain utterances without their context, and, on the other hand, to resolve references. That way, extracted utterances gain both readability and improved context. In case of multiple speakers, the summarization model can exploit this information to improve the reliability of the salience estimate and to enhance the readability of the extracts.

**User Interface** The interactive lecture browser presented in the previous chapter is a first prototype to study how users interact with the software and make use of the phrases and visualization. As it is often faster to read a transcript instead of listening to the audio, a text field showing the transcription and highlighting the currently spoken utterance may help the user to find the segments of interest faster. Similarly, the presentation slides, if available, can be displayed alongside the video and transcription. A further enhancement of the interface is the integration of a voice based search, allowing the user to search for certain terms, not only based on their textual match, but on the acoustic similarity, reducing the effect of misspellings of the search term or errors of the automatic speech recognition. A step towards an end-user ready product is the option of personalization based on harvested user interactions. If users can be tracked by a unique user id, their interactions can be harvested to both train and evaluate the key phrase extraction and ranking algorithms. Although the presented usability study shows a clear trend, it covers only ten participants and

a single video lecture. A larger study with more participants and video lectures is required to get a better insight to how users interact with the tool and what further interface improvements are required.

**Feedback Between the System Components** Currently, the system is implemented as a strict feed-forward process where the user interaction only applies to the very last step. However, the intermediate results of each component, and in particular the user feedback, can be propagated back to the individual steps to iteratively improve the overall outcome. For example, the feedback on salient phrases can be used to modify the language model of the speech recognition system to favor words of the related domain. The phrase ranking can be altered by re-weighting the phrases based on their relation to the user's choice. In case of multiple speakers, the user can provide a notion of credibility for each speaker that can be integrated into the phrase ranking and summarization algorithms.

# Chapter 8

# Summary

A growing number of universities and other educational institutions provide video recordings of lectures as supplemental study material. Unfortunately, the resulting videos are typically far from a good e-learning resource because of longer periods of silence or inactivity, unnecessary repetitions or reformulations, or corrections of prior errors or mistakes. Instructional films such as found in a distant learning curriculum are scripted to ensure a consistent, error-free and fluent presentation, and post-processed to correct possible shortcomings in recording the session and to enhance the quality of the video. However, this effort is time-consuming and thus expensive, which is the reason for many institutions to provide only the raw recording of the particular classes without further ado. While this is surely a first step towards better and distant teaching, the availability of the unedited videos is clearly not exploiting their potential. The presented work strives to make these videos a more valuable resource for students. The goal is to provide students with a tool to find the important information in the video lecture without the need of watching the whole video. The proposed interactive tools for summarization and video lecture browsing make use of automatic speech recognition, key phrase extraction and ranking, summarization, and visualization, which are the four major topics addressed in this thesis.

The recurrent theme in the experiments in this thesis is a newly acquired corpus of academic spoken English. The LMELectures, two series of 18 computer science lectures each, were recorded in high-definition audio and video, manually transcribed, and, for one recording, annotated with key phrases by five human subjects. The recordings were automatically segmented into 23 857 chunks with an average duration of 4.4 seconds based on the output of a phoneme recognition system. The about 29 hours of speech were transcribed using BLITZSCRIBE2, a new program designed for the rapid transcription of speech implemented on top of the Java Speech Toolkit. Using BLITZSCRIBE2, the transcription process could be sped up to about five times real time, from typically ten to 50 times real time. In total, about 300 500 words were transcribed with an average of 14 words per segment. The vocabulary consists of about 5 400 words, excluding foreign (mostly German) and mispronounced words. In addition to the transcriptions, the data set features the presentation slides in machine readable format and, for one series, a small set of key phrases for each session specified by the lecturer. In contrast to other corpora such as the British Academic Spoken English corpus or the Michigan Corpus of Academic Spoken English, the

LMELectures data set features a single speaker, a constant recording environment, and two consistent lecture series, making it an ideal base for experiments on automatic speech recognition, key phrase extraction and experiments with the interactive lecture browser.

The experiments on automatic speech recognition focus on the comparison of different types of acoustic models with respect to the number of parameters and recognition performance. To put the experimental results in the context of prior work, the models are first evaluated on the Wall Street Journal (WSJ) corpus, a corpus of about 60 hours of speech of professional speakers reading newspaper articles. The base for the experiments are standard continuous and subspace Gaussian mixture models implemented in the KALDI speech recognition toolkit. KALDI is extended by a traditional implementation of semi-continuous models, *i.e.*, using a shared single codebook of Gaussians with individual weights for each state, and a two-level tree based multi-codebook semi-continuous model that uses a number of smaller codebooks for acoustically similar states, again with individual weights per state. The results on WSJ show that a multi-codebook semi-continuous system achieves a moderately better word error rate (14.00 % WER, development set) than a continuous system (14.57 % WER) while using only a fourth of the number of Gaussian components. The relatively small number of components and the fact that they are shared among several states allows the use of full covariance matrices which results in a more accurate statistic model of the data. The subspace Gaussian mixture model based system shows the best performance (11.85 % WER) due to the additional per-state transformation of the Gaussian means in comparison. Although the transformations result in a higher computational complexity, the extended modeling capabilities allow to use a relatively small number of Gaussian components compared to a continuous or semi-continuous system, leading to an overall computational complexity similar as of the multi-codebook semi-continuous system. The performance of the traditional semi-continuous system is disappointing: beside the 26.28 % WER, the computational complexity is unacceptably higher than with the other models because of the rather large number of Gaussian components and the fact that the complete codebook needs to be evaluated for each observation. The presented inter- and intra-iteration smoothing techniques for semi-continuous models have to be applied with care: a moderate inter-iteration smoothing ($\rho = 0.3$) of the Gaussian weights shows a consistent improvement, however, the intra-iteration smoothing leads to inconclusive results. Prior experiments suggest that the smoothing of the sufficient statistics is beneficial in case of limited training data.

For the LMELectures data, a pre-computed language model covering about 65 000 words was reduced to the 5 383 actually occurring words *(small)* and further pruned to reduce the number of bi- and tri-grams and thus the size of the decoding WFST. The experiments showed an overall result similar to the results on the WSJ data: the SGMM system yields a WER of 11.03 % on the test set, followed by the multi-codebook semi-continuous system (12.75 % WER) and the continuous system (13.19 % WER). Interestingly, the best WER are achieved using a pruned version of the reduced language model, indicating that the n-gram probabilities of the original language model do not necessarily reflect the n-grams found in the LMELectures data.

Both summarization and visualization rely on the extraction of key phrases that are a good representation of the topics covered in the lectures. The algorithms presented in this thesis are unsupervised, *i.e.*, no prior learning step is required. The key phrases are extracted using a two step algorithm: in a first step, possible candidates are extracted, using regular expressions on top of part-of-speech tags. In a second step, the candidates are ranked according to heuristics based on frequency, length, position, or presence in a background corpus. The quality of these rankings is evaluated on a single recording, using manual phrase rankings by five human annotators. An evaluation that compared both human against human and automatic against human rankings using Average Precision (AP) and Normalized Distributed Cumulative Gain (NDCG) shows that the frequency and length based automatic rankings are indeed very similar to the human rankings. Interestingly, even human annotators only agree on a rather small number of key phrases (*e.g.*, five to eight) but disagree on items further down the ranking. The ranking that combines phrase frequency, phrase length and inverse document frequency shows the overall best agreement with the human rankings for both manual and automatic transcripts. It combines the raw phrase frequency with the inverse document frequency of the lecture series, and emphasizes longer phrases. The rather poor performance of the Kullback-Leibler ranking that compares the document's phrase distribution against a background corpus is best explained by the fact that longer phrases are likely to be missing in the background corpus, especially in the case of very technical lectures and rather general background material. The results based on automatic instead of manual transcriptions show that the frequency and length based ranking strategies are fairly robust against recognition errors and may, to some extent, even profit from errors: utterances with low recognition confidence are often less carefully articulated, and might thus carry less important details in the lecture setting.

The experiments on automatic summarization focus on global unsupervised models to extract these sentences that best represent a document or recording. Due to the lack of reference summaries for the LMELectures data set and to show the performance of the algorithms in the context of prior work, the experiments are conducted on the ICSIMC, a corpus of about 75 naturally occurring technical group meetings. Traditionally, summarization algorithms such as the well-known maximum marginal relevance (MMR) iteratively select the next best sentence for inclusion to a summary based on the relevance and redundancy to the previously selected sentences. Here, an integer linear program (ILP) is used to express the iterative MMR as a global optimization problem to avoid a possibly local optimum of the iterative solution. The resulting summaries show a relative improvement of 12.5 % in terms of ROUGE compared to the iterative solutions. The key idea of concept based summarization is to assume that the information is spread out over the sentences in little pieces, the concepts, and that a good summary covers as many concepts as possible. Here, the concepts are the automatically extracted key phrases together with the salience scores from the chosen ranking strategy. The sentence selection is modeled as a knapsack problem: the best selection of sentences covers the most distinct concepts while satisfying a certain textual length constraint. While MMR models the relevance and redundancy trade-off explicitly, this model considers relevance and redundancy indirectly. A large number of key phrases in a sentence indicates a higher

relevance potential, but including the same key phrase more than once does not increase the objective function value. Thus the optimization finds the most relevant selection while focusing on distinct concepts and thus low redundancy. The resulting summaries lead to an improvement of 11 % over the sentence based models in terms of ROUGE scores. Further advantages of the concept based system are the lack of a manually tuned relevance-redundancy trade-off factor and a significantly lower computational complexity: the number of ILP constraints is mainly governed by the number of unique concepts instead of the number of candidate sentences. The similarity in terms of technical language and spontaneous speech suggests that the results are transferable to the LMELectures data set.

An interesting observation from the summarization experiments is that human summaries yield about the same ROUGE score as the best automatic summary when compared to the remaining human abstract. In other words: the summary may be good, but does not match the expectations of the users. Experiments where user refined key phrases resulted in better ROUGE scores stimulated the implementation of an interactive summarization tool. The user can control which key phrases should be used to build the summary, and how they should be weighted. In an iterative process, the user can add, remove or re-weight certain phrases to obtain a new summary. Sentences that were displayed in earlier summaries are shaded gray to help the user skip over previously read sentences. Another interesting observation from the summarization experiments is that using the top 100 key phrases instead of a textual summary of 400 words leads to similar ROUGE score as both human and automatic abstracts, suggesting that the key phrases give a good overview. Also, the example summaries illustrate that extracts of spontaneous speech are often hard to read, especially if they lack context. Instead of a summary, the interactive lecture browser displays the ranked key phrases next to the video. Below the video, a stream graph is used to show the occurrences and dominance for a number of selected key phrases. Each phrase is represented by a colored stream where the horizontal direction represents the timeline and the vertical breadth indicates the phrase dominance at the given time.

The usability of this lecture browser is evaluated in a user study that aims at the use case of the browser: a student that prefers to watch only the parts of the video that cover a certain topic. For a given lecture and a short description of the covered topics, the subjects are asked to identify those three minute segments of the video that correspond to those topics, and to mark them on a time line. For two groups of five students each that are given the same video and task, the group using the lecture browser finished the task on average 29 % faster than the group that was presented with the video only, while achieving about the same accuracy in identifying the salient segments. A subsequent questionnaire collecting feedback regarding the user interface showed that most users find the lecture browser easy to use and helpful, and would recommend it to others. The key phrases were considered to be both helpful and accurate, and to give, together with the visualization, a good overview of the lecture. The users' interactions with the tools can be harvested to build up a database for future experiments on user-specific summaries and rankings. In particular, the addition, deletion and re-ranking of phrases provides valuable feedback to improve both candidate extraction and ranking algorithms.

# Appendix A

# The LME Corpus of Academic Spoken English Addenda

## A.1 Detailed Lecture List

| # | *duration* | *file name* |
|---|---|---|
| 1 | 00:57:15 | 20090427-Hornegger-IMIP01 |
| 2 | 01:16:33 | 20090428-Hornegger-IMIP02 |
| 3 | 00:48:27 | 20090504-Hornegger-IMIP03 |
| 4 | 00:45:06 | 20090511-Hornegger-IMIP05 |
| 5 | 01:25:15 | 20090512-Hornegger-IMIP06 |
| 6 | 01:23:42 | 20090519-Hornegger-IMIP08 |
| 7 | 00:42:23 | 20090525-Hornegger-IMIP09 |
| 8 | 01:26:41 | 20090526-Hornegger-IMIP10 |
| 9 | 01:25:14 | 20090609-Hornegger-IMIP12 |
| 10 | 00:36:11 | 20090615-Hornegger-IMIP13 |
| 11 | 01:24:35 | 20090616-Hornegger-IMIP14 |
| 12 | 01:27:53 | 20090623-Hornegger-IMIP16 |
| 13 | 00:41:23 | 20090629-Hornegger-IMIP17 |
| 14 | 01:27:22 | 20090630-Hornegger-IMIP18 |
| 15 | 01:23:17 | 20090707-Hornegger-IMIP20 |
| 16 | 00:33:12 | 20090713-Hornegger-IMIP21 |
| 17 | 00:35:08 | 20090720-Hornegger-IMIP22 |
| 18 | 01:11:04 | 20090721-Hornegger-IMIP23 |
| | 19:30:49 | *total playing time* |

Table A.1: List of *IMIP* lectures; held out *IMIP{04,07,11,15,19}* for technical reasons such as a different presenter, German language, or issues with the recording quality.

| # | *duration* | *file name* |
|---|---|---|
| 1 | 01:23:04 | 20090427-Hornegger-PA01 |
| 2 | 00:43:06 | 20090428-Hornegger-PA02 |
| 3 | 01:30:56 | 20090504-Hornegger-PA03 |
| 4 | 00:47:50 | 20090505-Hornegger-PA04 |
| 5 | 01:28:19 | 20090511-Hornegger-PA05 |
| 6 | 00:41:00 | 20090512-Hornegger-PA06 |
| 7 | 01:25:07 | 20090518-Hornegger-PA07 |
| 8 | 00:44:40 | 20090519-Hornegger-PA08 |
| 9 | 01:24:56 | 20090525-Hornegger-PA09 |
| 10 | 01:29:24 | 20090615-Hornegger-PA13 |
| 11 | 00:42:50 | 20090616-Hornegger-PA14 |
| 12 | 00:49:34 | 20090623-Hornegger-PA15 |
| 13 | 00:42:45 | 20090630-Hornegger-PA17 |
| 14 | 01:24:03 | 20090706-Hornegger-PA18 |
| 15 | 00:35:27 | 20090707-Hornegger-PA19 |
| 16 | 01:07:04 | 20090713-Hornegger-PA20 |
| 17 | 01:27:34 | 20090720-Hornegger-PA21 |
| 18 | 00:42:52 | 20090720-Hornegger-PA22 |
| | 19:10:40 | *total playing time* |

Table A.2: List of *PA* lectures; held out *PA{10,11,12,16}* for technical reasons such as a different presenter, German language, or issues with the recording quality.

# A.2 Transcription Guidelines

## Guidelines for Manual Transcriptions

Korbinian Riedhammer
korbinian.riedhammer@informatik.uni-erlangen.de

### 1 Introduction

Manually transcribed speech is used to train and evaluate automatic speech recognition systems. To obtain reliable results, the transcriptions have to be *as accurate as possible*, i.e., you need to write down *exactly what was said* and not *what the speaker meant to say*. Though the task seems simple, it requires a great deal of discipline and concentration.

### 2 Guidelines

To obtain consistent transcriptions, we kindly ask you to follow some guidelines.

- Capitalization is irrelevant.
- Umlauts are transcribed in `tex` style; e.g., `sch"one H"auser`.
- You *may* insert commas and colons, but it is not required.
- Words originating from *and pronounced in* a foreign language are appended a suffix following the ~character; e.g., `she grilled me a bratwurst~deu deluxe~fra`.
- Individual letters (e.g., when spelling a word) are transcribed as the letter following a dollar sign; e.g., `a large $c $c $d screen`.
- Numbers are transcribed as said; e.g., `one thousand five hundred`.
- Elision of phones as in "it's" or "you'd" are transcribed as spoken using the; apostrophe, e.g., `just 'cause it's been a hard day I do not give up`.
- Mispronounced and interrupted words are transcribed with a leading asterisk and in a phonetic way; e.g., `she made me a *pee *peana peanut butter *sundwich`.
- Unknown words are transcribed in a phonetic way with a leading question mark; e.g., `he suffered from a severe ?anorism`
- Typical hesitations are transcribed as words following a `#`; e.g. `#ahm, #ah, #ehm, #hm`.
- Acoustic events not covered by the above items are transcribed as follows.

| *type* | *token* | *type* | *token* |
|---|---|---|---|
| laughing | `#laugh` | coughing | `#cough` |
| (audible) breathing | `#breath` | sneezing | `#sneeze` |
| background noise | `#noise` | background speech | `#bs`, `#bsNumWords` |
| empty recording | `#nothing_said` | silence (> 1sec) | `#sil` |
| unitelligible word | `#ui` | microphone overload | `#mo` |

## 3 Examples

- `good morning welcome to the first of five lectures today we will`
- `the first topic is #ahm $s $v $m #sil so what's it all about`
- `yes marcus #bs15 yes that's correct #sil any other questions`

## 4 List of Technical Terms

$c, $c $t, $c plus plus, $d $p, $e, $e $m, $f, $f $f $t, $g, $h $m $m, $l $d $a, $m $l, $m $r $i, $p $c $a, $p $d $f, $s $s $d, $s $v $d, $x, $y, $z, (something) prime, (something) transposed, Bark, Baum-Welch, Bayes, Bayesian, Fourier, Haar, Hartley, Hess, Hessian, Hounsfield, Karhunen, Markov, Mel, Newton, Raphson, Tucker, Viterbi, algorithm, alpha, aneurysma, angio, apriori, axis, beta, bifurcation, calcification, calibration, carrotis, catheter, cerebral, classifier, clique potential, collimeter, compute, conditional probability, constraints, convex, coordinate system, cost function, covariance, criterion, decision boundary, deformation, deleted interpolation, denominator, density function, derivative, derive, differential equation, discrete, discretize, eigenvalue decomposition, eigenvector, encode, entropy, epsilon, exponential, extrapolation, extrinsic, functional, gallbladder, gamma, gradient descent, gray scale, harmonic, hidden Markov model, homogeneous, identity matrix, image plane, integral, interface, internship, interpolation, intrinsic, iterative, kernel, laproscopic surgery, linear programming, liver, log, logarithm, logistic, magnetic resonance imaging, manifold, marginal, marginalize, matlab, matrix (matrices), maximum likelihood estimation, metabolism, method, metrics, nominator, normalization, normalize, null space, numeric, numerical, objective function, orthogonal, orthographic projection, outlier, pacemaker, parametrization, pattern recognition, penalty function, perpendicular, phi, pi, pixel, point correspondence, posterior probability, prior probability, probabilistic, propagate, quaternion, radiation, random field, rank deficiency, recursive, registration, regression, rendering, rigid, rotation, sch"auferla deu, segmentation, sigma, simulated annealing, singular value decomposition, skew, slide, square root, stenosis (stenoses), sum of squared distances, support vector machine, tensor, texture, thorax, thyroid, time of flight, transposed, triangular, ultrasound, variance, vascular, vector, vessel, visualization, wavelet, xi, zeta.

# A.3 Key Phrase Questionnaire

Pattern Analysis, 2009-05-12

### Schlüsselphrasen

Wählen Sie aus der folgenden Liste diejenigen Begriffe aus, die den Inhalt der Vorlesung am besten wiedergeben. Übertragen Sie diese Begriffe in die untenstehende Tabelle und ordnen Sie sie nach ihrer Relevanz. Berücksichtigen Sie dabei ggf. auch die Zeit, die die entsprechenden Themen beansprucht haben. Eventuelle Synonyme können Sie einfach nebeneinander eintragenn.

Bewerten Sie bitte auch die Relevanz der einzelnen Phrasen mit einer Note von 1 (sehr relevant) bis 6 (nutzlos).

Sollte es im Text eine Phrase geben, die Ihrer Meinung nach in dieser Liste fehlt, tragen Sie sie einfach trotzdem ein und kennzeichnen Sie sie.

book, boundaries, case, classes, classifier, closed-form solution, covariances, decision, decision boundaries, distance, engineering, error, estimating, Euclidean norm, function, highest spread, ideal decision, instance, line, linear algebra, linear decision, linear decision boundaries, linear programming, linear regression, Mahalanobis distance, matrix, mechanical engineer, minimum, minus, modeling, norm, norm-dependent linear regression, number, optimization, overfitting, points, problems, question, regression, residual vector, ridge regression, scenario, set, slides, solution, spread, support vector machines, telling, terms, time, training, vector machines, vectors, webpage

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1. | ..... | 1 ☐ | 2 ☐ | 3 ☐ | 4 ☐ | 5 ☐ | 6 ☐ |
| 2. | ..... | 1 ☐ | 2 ☐ | 3 ☐ | 4 ☐ | 5 ☐ | 6 ☐ |
| 3. | ..... | 1 ☐ | 2 ☐ | 3 ☐ | 4 ☐ | 5 ☐ | 6 ☐ |
| 4. | ..... | 1 ☐ | 2 ☐ | 3 ☐ | 4 ☐ | 5 ☐ | 6 ☐ |
| 5. | ..... | 1 ☐ | 2 ☐ | 3 ☐ | 4 ☐ | 5 ☐ | 6 ☐ |
| 6. | ..... | 1 ☐ | 2 ☐ | 3 ☐ | 4 ☐ | 5 ☐ | 6 ☐ |
| 7. | ..... | 1 ☐ | 2 ☐ | 3 ☐ | 4 ☐ | 5 ☐ | 6 ☐ |
| 8. | ..... | 1 ☐ | 2 ☐ | 3 ☐ | 4 ☐ | 5 ☐ | 6 ☐ |
| 9. | ..... | 1 ☐ | 2 ☐ | 3 ☐ | 4 ☐ | 5 ☐ | 6 ☐ |
| 10. | ..... | 1 ☐ | 2 ☐ | 3 ☐ | 4 ☐ | 5 ☐ | 6 ☐ |
| 11. | ..... | 1 ☐ | 2 ☐ | 3 ☐ | 4 ☐ | 5 ☐ | 6 ☐ |
| 12. | ..... | 1 ☐ | 2 ☐ | 3 ☐ | 4 ☐ | 5 ☐ | 6 ☐ |
| 13. | ..... | 1 ☐ | 2 ☐ | 3 ☐ | 4 ☐ | 5 ☐ | 6 ☐ |
| 14. | ..... | 1 ☐ | 2 ☐ | 3 ☐ | 4 ☐ | 5 ☐ | 6 ☐ |
| 15. | ..... | 1 ☐ | 2 ☐ | 3 ☐ | 4 ☐ | 5 ☐ | 6 ☐ |
| 16. | ..... | 1 ☐ | 2 ☐ | 3 ☐ | 4 ☐ | 5 ☐ | 6 ☐ |
| 17. | ..... | 1 ☐ | 2 ☐ | 3 ☐ | 4 ☐ | 5 ☐ | 6 ☐ |
| 18. | ..... | 1 ☐ | 2 ☐ | 3 ☐ | 4 ☐ | 5 ☐ | 6 ☐ |
| 19. | ..... | 1 ☐ | 2 ☐ | 3 ☐ | 4 ☐ | 5 ☐ | 6 ☐ |
| 20. | ..... | 1 ☐ | 2 ☐ | 3 ☐ | 4 ☐ | 5 ☐ | 6 ☐ |

# List of Figures

# List of Tables

# Bibliography


[Aho 86] A. Aho, J. Hopcroft, and J. Ullman. *Compilers, Principles, Techniques and Tools*. Addison Wesley, 1986.

[Alla 07] C. Allauzen, M. Riley, J. Schalkwyk, W. Skut, and M. Mohri. "OpenFST: a General and efficient weighted finite-state transducer library". In: *Proc. Int'l Conference on Implementation and Application of Automata*, pp. 11–23, 2007.

[Baez 99] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1999.

[Bake 75] J. M. Baker. "The DRAGON system - an overview". *IEEE Trans. Acoustics, Speech and Signal Processing*, Vol. 23, No. 1, pp. 24–29, 1975.

[Barr 01] C. Barras, E. Geoffrois, Z. Wu, and M. Liberman. "Transcriber: Development and use of a tool for assisting speech corpora production". *Speech Communication*, Vol. 33, No. 1-2, pp. 5–22, 2001.

[Bene 08] J. Benesty, M. Sondhi, and Y. Huang, Eds. *Springer Handbook of Speech Processing*. Springer, 2008.

[Berg 13] C. Berger. "Controlling from a Distant Point the Operation of a Mechanism or Instrument". U.S. pat. 1,055,985 (filed Aug 26, 1909), 1913.

[Berg 16] C. Berger. "Sound Operated Circuit Controller". U.S. pat. 1,209,636 (filed May 19, 1916), 1916.

[Bisa 08] M. Bisani and H. Ney. "Joint-Sequence Models for Grapheme-to-Phoneme Conversion". *Speech Communication*, Vol. 50, No. 5, pp. 434–451, 2008.

[Bren 96] S. Brennan. "Lexical entrainment in spontaneous dialog". In: *Proc. Int'l Symposium on Spoken Dialogue (ISSD)*, pp. 41–44, 1996.

[Bril 92] E. Brill. "A Simple Rule-Based Part of Speech Tagger". In: *Proc. Conference on Applied Natural Language Processing (ANLP)*, pp. 152–155, 1992.

[Burg 98] C. Burges. "A Tutorial on Support Vector Machines for Pattern Recognition". *Data Mining and Knowledge Discovery*, Vol. 2, No. 2, pp. 121–167, 1998.

[Burn 07] L. Burnard, Ed. *Reference Guide for the British National Corpus*. Research Technologies Service at Oxford University Computing Services, 2007.

[Byro 08] L. Byron and M. Wattenberg. "Stacked Graphs — Geometry & Aesthetics". *IEEE Trans. Visualization and Computer Graphics (TVCG)*, Vol. 14, No. 6, pp. 1245–1252, 2008.

[Carb 98] J. Carbonell and J. Goldstein. "The use of MMR, diversity-based reranking for reordering documents and producing summaries". In: *Proc. ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 335–336, 1998.

[Chen 10] Y.-N. Chen, Y. Huang, S.-Y. Kong, and L.-S. Lee. "Automatic Key Term Extraction From Spoken Course Lectures Using Branching Entropty and Prosodic/Semantic Features". In: *Proc. IEEE Workshop on Spoken Language Technologies (SLT)*, pp. 265–270, 2010.

[Chen 96] S. F. Chen and J. Goodman. "An Empirical Study of Smoothing Techniques for Language Modeling". In: *Proc. Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 310–318, 1996.

[Chin 88] J. P. Chin, V. A. Diehl, and K. L. Norman. "Development of an Instrument Measuring User Satisfaction of the Human-Computer Interface". In: *Proc. ACM SIGCHI Conference on Human Factors in Computing Systems (CHI)*, pp. 213–218, 1988.

[Chri 04] H. Christensen, B. Kolluru, Y. Gotoh, and S. Renals. "From Text Summarisation to Style-Specific Summarisation for Broadcast News". *Lecture Notes in Computer Science*, Vol. 2997, pp. 223–237, 2004.

[Clim 61] W. D. Climenson, N. H. Hardwick, and S. N. Jacobson. "Automatic syntax analysis in machine indexing and abstracting". *American Documentation*, Vol. 12, No. 3, pp. 178–183, 1961.

[Cohe 60] J. Cohen. "A Coefficient of Agreement for Nominal Scales". *Educational and Psychological Measurement*, Vol. 20, No. 1, pp. 37–46, 1960.

[Davi 52] K. H. Davis, R. Biddulph, and S. Balashel. "Automatic Recognition of Spoken Digits". *J. Acoust. Soc. Am.*, Vol. 24, No. 6, pp. 637–642, 1952.

[Davi 62] E. E. J. David and O. G. Selfridge. "Eyes and Ears for Computers". *Proceedings of the Institute of Radio Engineers (IRE)*, Vol. 50, pp. 1093–1101, 1962.

[Davi 80] S. B. Davis and P. Mermelstein. "Comparison of Parametric Representations for Monosyllabic Word Recognition in Continuously Spoken Sentences". *IEEE Trans. Audio, Speech and Signal Processing*, Vol. 28, No. 4, pp. 357–366, 1980.

[Davi 89] F. D. Davis. "Perceived Usefulness, Perceived Ease of User, and User Acceptance of Information Technology". *MIS Quarterly*, Vol. 13, No. 3, pp. 319–340, 1989.

[Demp 77] A. P. Dempster, N. M. Laird, and D. B. Rubin. "Maximum Likelihood from Incomplete Data via the EM Algorithm". *Journal of the Royal Statistical Society*, Vol. 39, No. 1, pp. 1–38, 1977.

[Dudl 39a] H. W. Dudley. "The Vocoder". *Bell Labs Record*, Vol. 17, pp. 122–126, 1939.

[Dudl 39b] H. W. Dudley. "Voice Operated Mechanism". U.S. pat. 2,238,555 (filed Mar 15, 1939), 1939.

[Edmu 61] H. P. Edmundson and R. E. Wyllys. "Automatic Abstracting and Indexing – Survey and Recommendations". *Comm. ACM*, Vol. 4, No. 5, pp. 226–234, 1961.

[Edmu 69] H. P. Edmundson. "New Methods in Automatic Extracting". *Journal of the Association for Computing Machinery*, Vol. 16, pp. 264–285, 1969.

[Fell 98] C. Fellbaum, Ed. *WordNet: An Electronic Lexical Database*. The MIT Press, 1998.

[Fila 04] E. Filatova and V. Hatzivassiloglou. "Event-Based Extractive Summarization". In: *Proc. ACL Workshop on Summarization*, pp. 104–111, 2004.

[Fish 36] R. Fisher. "The Use of Multiple Measurements in Taxonomic Problems". *Ann. Eugenics*, Vol. 7, pp. 179–188, 1936.

[Furu 04] S. Furui, T. Kikuchi, Y. Shinnaka, and C. Hori. "Speech-to-text and speech-to-speech summarization of spontaneous speech". *IEEE Trans. Speech and Audio Processing*, Vol. 12, No. 4, pp. 401–408, 2004.

[Furu 86] S. Furui. "Speaker-Independent Isolated Word Recognition Using Dynamic Features of Speech Spectrum". *IEEE Trans. Audio, Speech and Signal Processing*, Vol. 34, No. 1, pp. 52–59, 1986.

[Gale 96] M. Gales, D. Pye, and P. Woodland. "Variance Compensation within the MLLR Framework for Robust Speech Recognition and Speaker Adaptation". In: *Proc. Int'l Conference on Spoken Language Processing (ICSLP)*, pp. 1832–1835, 1996.

[Gale 98] M. Gales. "Maximum Likelihood Linear Transformations for HMM-Based Speech Recognition". *Computer Speech and Language*, Vol. 12, pp. 75–98, 1998.

[Gall 06] M. Galley. "A Skip-Chain Conditional Random Field for Ranking Meeting Utterances by Importance". In: *Proc. ACL Conference on Empirical Methods in Natural Language Processing*, pp. 364–372, 2006.

[Garg 09] N. Garg, B. Favre, K. Riedhammer, and D. Hakkani-Tür. "ClusterRank: A Graph Based Method for Meeting Summarization". In: *Proc. Annual Conference of the Int'l Speech Communication Association (INTERSPEECH)*, pp. 1499–1502, 2009.

[Garo 07] J. Garofalo, D. Graff, D. Paul, and D. Pallett. "CSR-I,II (WSJ0,1) Complete". Linguistic Data Consortium, Philadelphia, USA, 2007.

[Gill 08] D. Gillick, B. Favre, and D. Hakkani-Tür. "The ICSI Summarization System at TAC'08". In: *Proc. of the Text Analysis Conference workshop*, pp. 227–234, 2008.

[Gill 09] D. Gillick, K. Riedhammer, B. Favre, and D. Hakkani-Tür. "A Global Optimization Framework for Meeting Summarization". In: *Proc. IEEE Int'l Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 4769–4772, 2009.

[Gori 97] A. Gorin, G. Riccardi, and J. Wright. "How may I help you?". *Speech Communication*, Vol. 23, pp. 113–127, 1997.

[Grev 98] J. Greven, Ed. *Das Funkkolleg 1966-1998. Ein Modell wissenschaftlicher Weiterbildung im Medienverbund*. Deutscher Studien Verlag, Weinheim, Germany, 1998.

[Grop 10] M. Gropp. "Key Phrases for the Textual and Visual Summarization of Academic Spoken Language". Studienarbeit Informatik, Dept. Informatik 5, Univ. Erlangen-Nuremberg, 2010.

[Grop 11] M. Gropp, E. Nöth, and K. Riedhammer. "A Novel Lecture Browsing System Using Ranked Key Phrases and StreamGraphs". In: *Proc. Int'l Conference on Text, Speech and Dialogue (TSD)*, pp. 17–24, 2011.

[Ha 02] L. Ha, E. Sicilia-Garcia, J. Ming, and F. Smith. "Extension of Zipf's law to words and phrases". In: *Proc. Int'l Conference on Computational Linguistics*, pp. 1–6, 2002.

[Haeb 92] R. Haeb-Umbach and H. Ney. "Linear discriminant analysis for improved large vocabulary continuous speech recognition". In: *Proc. IEEE Int'l Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 13–16, 1992.

[Havr 02] S. Havre, E. Hetzler, P. Whitney, and L. Nowell. "ThemeRiver: Visualizing Thematic Changes in Large Document Collections". *IEEE Trans. Visualization and Computer Graphics (TVCG)*, Vol. 8, No. 1, pp. 9–20, 2002.

[He 00] L. He, E. Sanocki, A. Gupta, and J. Grudin. "Comparing Presentation Summaries: Slides vs. Reading vs. Listening". In: *Proc. ACM SIGCHI Conference on Human Factors in Computing Systems (CHI)*, pp. 177–184, 2000.

[He 99] L. He, E. Sanocki, A. Gupta, and J. Grudin. "Auto-Summarization of Audio-Video Presentations". In: *Proc. ACM International Conference on Multimedia*, pp. 489–498, 1999.

[Hona 05] M. Honal and T. Schultz. "Automatic Disfluency Removal on Recognized Spontaneous Speech – Rapid Adaptation to Speaker-Dependent Disfluencies". In: *Proc. IEEE Int'l Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 969–972, 2005.

[Hori 02] C. Hori, S. Furui, R. Malkin, H. Yu, and A. Waibel. "Automatic Speech Summarization Applied to English Broadcast News Speech". In: *Proc. IEEE Int'l Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 9–12, 2002.

[Horn 07] K. Hornbæk and E. L. C. Law. "Meta-analysis of correlations among usability measures". In: *Proc. ACM SIGCHI Conference on Human Factors in Computing Systems (CHI)*, pp. 617–626, 2007.

[Hovy 06] E. Hovy, C. Lin, L. Zhou, and J. Fukumoto. "Automated Summarization Evaluation with Basic Elements". In: *Proc. Language Resources and Evaluation (LREC)*, pp. 899–902, 2006.

[Huan 01] X. Huang, A. Acero, and H.-W. Hon. *Spoken Language Processing – A Guide to Theory, Algorithm, and System Development.* Prentice Hall, 2001.

[Huan 07] Z. Huang, M. Harper, and W. Wang. "Mandarin Part-of-Speech Tagging and Discriminative Reranking". In: *Proc. Conference on Empirical Methods in Natural Language Processing (EMNLP) and Computational Natural Language Learning (CoNLL)*, pp. 1093–1102, 2007.

[Huan 89] X. D. Huang and M. A. Jack. "Semi-continuous hidden Markov models for speech signals". *Computer Speech and Language*, Vol. 3, No. 3, pp. 239–251, 1989.

[Hube 10] J. Huber, J. Steimle, S. Olberding, R. Lissermann, and M. Mühlhäuser. "Browsing E-Lecture Libraries on Mobile Devices: A Spacial Interaction Concept". In: *Proc. IEEE Int'l Conference on Advanced Learning Technologies*, pp. 151–155, 2010.

[Hunt 91] J. Hunt, S. Richardson, D. Bateman, and A. Piau. "An Investigation of PLP and IMELDA Acoustic Representations and of their Potential for Combination". In: *Proc. IEEE Int'l Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 881–884, 1991.

[Inou 04] A. Inoue, T. Mikami, and Y. Yamashita. "Improvement of Speech Summarization Using Prosodic Information". In: *Proc. Int'l Conference on Speech Prosody*, pp. 599–602, 2004.

[Jani 03] A. Janin, D. Baron, J. Edwards, D. Ellis, D. Gelbart, N. Morgan, B. Peskin, T. Pfau, E. Shriberg, and A. Stolcke. "The ICSI Meeting Corpus". In: *Proc. IEEE Int'l Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 364–367, 2003.

[Jarv 00] K. Järvelin and J. Kekäläinen. "IR Evaluation Methods for Retrieving Highly Relevant Documents". In: *Proc. ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 41–48, 2000.

[Jarv 02] K. Järvelin and J. Kekäläinen. "Cumulated Gain-Based Evaluation of IR Techniques". *ACM Trans. Information Systems*, Vol. 20, No. 4, pp. 422–446, 2002.

[Jeff 48] H. Jeffreys. *Theory of Probability.* Claredon Press, 2 Ed., 1948.

[Jeli 76] F. Jelinek. "Continuous Speech Recognition by Statistical Methods". *IEEE Proceedings*, Vol. 64, No. 4, pp. 532–556, 1976.

[Jura 08] D. Jurafsky and J. H. Martin. *Speech and Language Processing.* Pearson Prentice Hall, 2008.

[Knes 95] R. Kneser and H. Ney. "Improved Backing-Off for m-gram Language Modeling". In: *Proc. IEEE Int'l Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 181–184, 1995.

[Knut 97] D. E. Knuth. *The Art of Computer Programming, Volume 2: Seminumerical Algorithms.* Addison-Wesley, 3rd Ed., 1997.

[Kraa 06] W. Kraaij and W. Post. "Task Based Evaluation of Exploratory Search Systems". In: *Proc. ACM SIGIR Workshop on Evaluation of Exploratory Search Systems*, pp. 24–27, 2006.

[Krip 03] K. Krippendorff. *Content Analysis, an Introduction to Its Methodology.* Sage Publications, Thousand Oaks, CA, 2nd Ed., 2003.

[Kull 51] S. Kullback and R. A. Leibler. "On Information and Sufficiency". *The Annals of Mathematical Statistics*, Vol. 22, No. 1, pp. 79–86, 1951.

[Leba 01] K. Lebart and J. M. Boucher. "A New Method Based on Spectral Subtraction for Speech Dereverberation". *ACOUSTICA*, Vol. 87, pp. 359–366, 2001.

[Leve 66] V. I. Levenshtein. "Binary codes capable of correcting deletions, insertions, and reversals". *Soviet Physics Doklady*, Vol. 10, No. 8, pp. 707–710, 1966.

[Levi 07] M. Levit. *Spoken Language Understanding without Transcriptions in a Call Center Scenario.* Logos, 2007.

[Lewi 95] J. R. Lewis. "IBM Computer Usability Satisfaction Questionnaires: Psychometric Evaluation and Instructions for Use". *International Journal of Human-Computer Interaction*, Vol. 7, No. 1, pp. 57–78, 1995.

[Like 32] R. Likert. "A Technique for the Measurement of Attitudes". *Archives of Psychology*, Vol. 140, pp. 1–55, 1932.

[Lin 04] C. Lin. "ROUGE: A Package for Automatic Evaluation of Summaries". In: *Proc. ACL Workshop on Text Summarization*, pp. 25–26, 2004.

[Lin 09a] H. Lin, J. Bilmes, and S. Xie. "Graph-based Submodular Selection for Extractive Summarization". In: *Proc. IEEE Workshop on Speech Recognition and Understanding (ASRU)*, pp. 381–386, 2009.

[Lin 09b] J. Lin. "Summarization". In: L. Liu and M. Özsu, Eds., *Encyclopedia of Database Systems*, pp. 2884–2889, Springer, 2009.

[Lin 10] J. Lin, N. Madnani, and B. Dorr. "Putting the user in the loop: Interactive Maximal marginal Relevance for Query-focused Summarization". In: *Proc. Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL): Human Language Technologies (HLT)*, pp. 305–308, 2010.

[Lin 97] H. X. Lin, Y.-Y. Choong, and G. Salvendy. "A Proposed Index of Usability: A Method for Comparing the Relative Usability of Different Software Systems". *Behaviour & Information Technology*, Vol. 16, No. 4, pp. 267–278, 1997.

[Liu 07] Y. Liu and E. Shriberg. "Comparing Evaluation Metrics for Sentence Boundary Detection". In: *Proc. IEEE Int'l Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 185–288, 2007.

[Liu 08] Y. Liu and S. Xie. "Impact of Automatic Sentence Segmentation on Meeting Summarization". In: *Proc. IEEE Int'l Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 5009–5012, 2008.

[Liu 09] F. Liu and Y. Liu. "From Extractive to Abstractive Meeting Summaries: Can It Be Done by Sentence Compression?". In: *Proc. Annual Meeting of the Association for Computational Linguistics (ACL) and the Asian Federation of Natural Language Processing (AFNLP)*, pp. 261–264, 2009.

[Liu 11] F. Liu, F. Liu, and Y. Liu. "A Supervised Framework for Keyword Extraction from Meeting Transcripts". *IEEE Trans. Audio, Speech and Language Processing*, Vol. 19, No. 3, pp. 538–548, 2011.

[Luhn 58] H. P. Luhn. "The Automatic Creation of Literature Abstracts". *IBM Journal of Research and Development*, Vol. 2, No. 2, pp. 159–165, 1958.

[Lund 01] A. M. Lund. "Measuring Usability with the USE Questionnaire". *STC Usability SIG Newsletter*, Vol. 8, No. 2, p. *no pagination*, 2001.

[Mand 09] M. I. Mandel, R. J. Weiss, and D. Ellis. "Model-Based Expectation-Maximization Source Separation and Localization". *IEEE Trans. Audio, Speech and Language Processing*, Vol. 18, No. 2, pp. 382–394, 2009.

[Mani 01] I. Mani. *Automatic Summarization.* Vol. 3 of *Natural Language Processing*, John Benjamins, Amsterdam, NL, 2001.

[Mask 05] S. Maskey and J. Hirschberg. "Comparing Lexical, Acoustic/Prosodic, Structural and Discourse Features for Speech Summarization". In: *Proc. European Conference on Speech Communication and Technology (EUROSPEECH)*, pp. 621–624, 2005.

[Mate 05a] P. Matejka, , L. Burget, P. Schwarz, and J. Cernocky. "Brno University of Technology System for NIST 2005 Language Recognition Evaliation". In: *Proc. 2005 NIST Language Recognition Evaluation*, pp. 1–7, 2005.

[Mate 05b] P. Matejka, P. Schwarz, J. Cernocky, and P. Chytil. "Phonotactic Language Identification using High Quality Phoneme Recognition". In: *Proc. Annual Conference of the Int'l Speech Communication Association (INTERSPEECH)*, pp. 2237–2240, 2005.

[McCo 05] I. McCowan, J. Carletta, W. Kraaij, S. Ashby, S. Bourban, M. Flynn, M. Guillemot, T. Hain, J. Kadlec, V. Karaiskos, G. Kronenthal, M. Lincoln, A. Lisowska, W. Post, D. Reidsma, and P. Wellner. "The AMI Meeting Corpus". In: *Proc. Int'l Conference on Methods and Techniques in Behavioral Research (Measuring Behavior)*, p. *no pagination*, 2005.

[McDo 07] R. McDonald. "A Study of Global Inference Algorithms in Multi-document Summarization". *Lecture Notes in Computer Science*, Vol. 4425, pp. 557–564, 2007.

[Merg 85] D. Mergel and H. Ney. "Phonetically Guided Clustering for Isolated Word Recognition". In: *Proc. IEEE Int'l Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 854–857, 1985.

[Mies 07] M. Mieskes, C. Müller, and M. Strube. "Improving Extractive Dialog Summarization by Utilizing Human Feedback". In: *Proc. IASTED Artificial Intelligence and Applications*, pp. 672–677, 2007.

[Mohr 02] M. Mohri, F. Pereira, and M. Riley. "Weighted Finite-State Transducers in Speech Recognition". *Computer Speech and Language*, Vol. 16, pp. 69–88, 2002.

[Mohr 97] M. Mohri. "Finite-State Transducers in Language and Speech Processing". *Computational Linguistics*, Vol. 23, No. 2, pp. 269–311, 1997.

[Mroz 05] J. Mrozinski, E. Whittaker, P. Chatain, and S. Furui. "Automatic sentence segmentation of speech for automatic summarization". In: *Proc. IEEE Int'l Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 981–984, 2005.

[Murr 05a] G. Murray, S. Renals, and J. Carletta. "Extractive Summarization of Meeting Recordings". In: *Proc. European Conference on Speech Communication and Technology (EUROSPEECH)*, pp. 593–596, 2005.

[Murr 05b] G. Murray, S. Renals, J. Carletta, and J. Moore. "Evaluating automatic summaries of meeting recordings". In: *Proc. ACL Workshop on Machine Translation and Summarization Evaluation (MTSE)*, pp. 39–52, 2005.

[Murr 06] G. Murray, S. Renals, J. Carletta, and J. Moore. "Incorporating speaker and discourse features into speech summarization". In: *Proc. Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL): Human Language Technologies (HLT)*, pp. 367–374, 2006.

[Murr 09] G. Murray, T. Kleinbauer, P. Poller, T. Becker, S. Renals, and J. Kilgour. "Extrinsic summarization evaluation: A decision audit task". *ACM Trans. Speech Lang. Process.*, Vol. 6, No. 2, pp. 1–29, 2009.

[Nenk 04] A. Nenkova and R. Passonneau. "Evaluating content selection in summarization: The Pyramid Method". In: *Proc. Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL): Human Language Technologies (HLT)*, pp. 145–152, 2004.

[Nenk 06] A. Nenkova. "Summarization Evaluation for Text and Speech: Issues and Approaches". In: *Proc. Annual Conference of the Int'l Speech Communication Association (INTERSPEECH)*, pp. 1527–1530, 2006.

[Newm 68] E. A. Newman. "Speech Recognition Apparatus". U.S. pat. 3,400,216 (filed Feb 1, 1965), 1968.

[Pear 96] K. Pearson. "Mathematical Contributions to the Theory of Evolution. III. Regression, Heredity, and Panmixia". *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, Vol. 187, pp. 253–318, 1896.

[Penn 08] G. Penn and X. Zhu. "A Critical Reassessment of Evaluation Baselines for Speech Summarization". In: *Proc. Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL): Human Language Technologies (HLT)*, pp. 470–478, 2008.

[Phan 06] X.-H. Phan. "CRFTagger: CRF English POS Tagger". 2006. `http://crftagger.sourceforge.net`, retrieved Feb. 20, 2012.

[Pohl 05] K. C. Pohlmann. *Principles of Digital Audio*. McGraw-Hill Professional, 2005.

[Port 80] M. F. Porter. "An algorithm for suffix stripping". *Program: Electronic Library & Information Systems*, Vol. 40, No. 3, pp. 211–218, 1980.

[Pove 11a] D. Povey, L. Burget, M. Agarwal, P. Akyazi, F. Kai, A. Ghoshal, G. O., N. Goel, M. Karafiát, A. Rastrow, R. Rose, P. Schwarz, and S. Thomas. "The subspace Gaussian mixture model - A structured model for speech recognition". *Computer Speech and Language*, Vol. 25, No. 2, pp. 404–439, 2011.

[Pove 11b] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, G. Nagendra, M. Hannemann, P. Motlíček, Y. Qian, P. Schwarz, J. Silovský, G. Stemmer, and K. Veselý. "The Kaldi Speech Recognition Toolkit". In: *Proc. IEEE Workshop on Speech Recognition and Understanding (ASRU)*, p. *no pagination*, 2011.

[Pove 12] D. Povey, M. Hannemann, G. Boulianne, L. Burget, A. Ghoshal, M. Janda, M. Karaffiat, S. Kombrink, P. Motlicek, Y. Quian, N. Thang Vu, K. Riedhammer, and K. Vesely. "Generating Exact Lattices in the WFST Framework". In: *Proc. IEEE Int'l Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 4213–4216, 2012.

[Pras 05] R. Prasad, S. Matsoukas, C. Kao, J. Ma, D. Xu, T. Colthurst, O. Kimball, R. Schwartz, J. Gauvain, L. Lamel, *et al.* "The 2004 BBN/LIMSI 20xRT English conversational telephone speech recognition system". In: *Proc. Annual Conference of the Int'l Speech Communication Association (INTERSPEECH)*, pp. 1645–1648, 2005.

[Reis 87] R. A. Reiser. *Instructional Technology: A History*, Chap. 2, pp. 11–48. Lawrence Erlbaum Associates, Inc., Hillsdale, NJ, USA, 1987.

[Reyn 00] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn. "Speaker Verification Using Adapted Gaussian Mixture Models". *Digital Signal Processing*, Vol. 10, pp. 19–41, 2000.

[Ried 07] K. Riedhammer. "User Modeling in Emotion Recognition". Diplomarbeit Informatik, Dept. Informatik 5 (Pattern Recognition), University of Erlangen-Nuremberg, 2007.

[Ried 08a] K. Riedhammer, B. Favre, and D. Hakkani-Tür. "A Keyphrase Based Approach to Interactive Meeting Summarization". In: *Proc. IEEE Workshop on Spoken Language Technologies (SLT)*, pp. 153–156, 2008.

[Ried 08b] K. Riedhammer, D. Gillick, B. Favre, and D. Hakkani-Tür. "Packing the Meeting Summarization Knapsack". In: *Proc. Annual Conference of the Int'l Speech Communication Association (INTERSPEECH)*, pp. 2434–2437, 2008.

[Ried 10] K. Riedhammer, B. Favre, and D. Hakkani-Tür. "Long Story Short – Global Unsupervised Models for Keyphrase Based Meeting Summarization". *Speech Communication*, Vol. 52, No. 10, pp. 801–815, 2010.

[Ried 11] K. Riedhammer, M. Gropp, and E. Nöth. "A Novel Lecture Browser Using Key Phrases and StreamGraphs". In: *Proc. IEEE Workshop on Speech Recognition and Understanding (ASRU)*, p. *no pagination*, 2011.

[Ried 12] K. Riedhammer, T. Bocklet, A. Ghoshal, and D. Povey. "Revisiting Semi-Continuous Hidden Markov Models". In: *Proc. IEEE Int'l Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 4721–4724, 2012.

[Roy 09] B. Roy and D. Roy. "Fast transcription of unstructured audio recordings". In: *Proc. Annual Conference of the Int'l Speech Communication Association (INTERSPEECH)*, pp. 1647–1650, 2009.

[Ryba 09] D. Rybach, C. Gollan, G. Heigold, B. Hoffmeister, J. Lööf, R. Schlüter, and H. Ney. "The RWTH Aachen University Open Source Speech Recognition System". In: *Proc. Annual Conference of the Int'l Speech Communication Association (INTERSPEECH)*, pp. 2111–2114, 2009.

[Saet 68] P. Saettler. *History of Instructional Technology.* McGraw-Hill, NY, USA, 1968.

[Sako 78] H. Sakoe and S. Chiba. "Dynamic programming algorithm optimization for spoken word recognition". *IEEE Trans. Audio, Speech and Signal Processing*, Vol. 26, No. 1, pp. 43–49, 1978.

[Salo 78] A. Salomaa and M. Soittola. *Automata-Theoretic Aspects of Formal Power Series.* Springer-Verlag, 1978.

[Sant 90] B. Santorini. "Part-of-Speech Tagging Guidelines for the Penn Treebank Project (3rd Revision)". Tech. Rep. MS-CIS-90-47, University of Pennsylvania Department of Computer and Information Science, 1990.

[Schu 94] E. Schukat-Talamazzini, T. Kuhn, and H. Niemann. "Speech Recognition for Spoken Dialog Systems". In: H. Niemann, R. De Mori, and G. Hanrieder, Eds., *Progress and Prospects of Speech Research and Technology in Proc. Artificial Intelligence*, pp. 110–120, 1994.

[Schu 95] E.-G. Schukat-Talamazzini. *Automatische Spracherkennung – Grundlagen, statistische Modelle und effiziente Algorithmen.* Vieweg, Braunschweig, Germany, 1995.

[Shne 04] B. Shneiderman and C. Plaisant. *Designing the User Interface: Strategies for Effective Human-ComputerInteraction.* Addison Wesley, 4th Ed., 2004.

[Simp 02] R. Simpson, S. Briggs, J. Ovens, and J. Swales. "The Michigan Corpus of Academic Spoken English". Tech. Rep., University of Ann Arbor, MI, USA, 2002.

[Sjob 07] J. Sjöbergh. "Older versions of the ROUGEeval summarization evaluation system were easier to fool". *Information Processing & Management*, Vol. 43, No. 6, pp. 1500–1505, 2007.

[Spar 72] K. Spärck Jones. "A Statistical Interpretation of Term Specificity and its Application in Retrieval". *Journal of Documentation*, Vol. 28, No. 1, pp. 11–21, 1972.

[Spar 96] K. Spärck Jones and J. Galliers. *Evaluating Natural Language Processing Systems.* Vol. 1083 of *Lecture Notes in Artificial Intelligence*, Springer-Verlag Berlin, 1996.

[Spar 98] K. Spärck Jones. "Automatic Summarising: Factors and Directions". In: *Advances in Automatic Text Summarization*, pp. 1–12, MIT Press, 1998.

[Spea 04] C. Spearman. "The Proof and Measurement of Association between Two Things". *The American Journal of Psychology*, Vol. 15, No. 1, pp. 72–101, 1904.

[Stei 11] S. Steidl, K. Riedhammer, T. Bocklet, F. Hönig, and E. Nöth. "Java Visual Speech Components for Rapid Application Development of GUI based Speech Processing Applications". In: *Proc. Annual Conference of the Int'l Speech Communication Association (INTERSPEECH)*, pp. 3257–3260, 2011.

[Stol 08] A. Stolcke, X. Anguera, K. Boakye, O. Cetin, A. Janin, M. Magimai-Doss, C. Wooters, and J. Zheng. "The SRI-ICSI Spring 2007 Meeting and Lecture Recognition System". In: *Multimodal Technologies for Perception of Humans. International Evaluation Workshops CLEAR 2007 and RT 2007, Springer Lecture Notes in Computer Science 4625*, pp. 450–463, 2008.

[Stol 11] A. Stolcke, J. Zheng, W. Wang, and V. Abrash. "SRILM at Sixteen: Update and Outlook". In: *Proc. IEEE Workshop on Speech Recognition and Understanding (ASRU)*, p. *no pagination*, 2011.

[Stol 98] A. Stolcke. "Entropy-based Pruning of Backoff Language Models". In: *DARPA Broadcast News Transcription and Understanding Workshop*, pp. 8–11, 1998.

[Sutt 06] C. Sutton and A. McCallum. "An Introduction to Conditional Random Fields for Relational Learning". In: L. Getoor and B. Taskar, Eds., *Introduction to Statistical Relational Learning*, pp. 93–127, MIT Press, 2006.

[Taka 09] H. Takamura and M. Okumura. "Text summarization model based on maximum coverage problem and its variant". In: *Proc. Conference of the European Chapter of the ACL*, pp. 781–789, 2009.

[Thed 99] S. Thede and M. Harper. "A Second-Order Hidden Markov Model for Part-of-Speech Tagging". In: *Proc. Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 175–182, 1999.

[Tomo 03] T. Tomokiyo and M. Hurst. "A Language Model Approach to Keyphrase Extraction". In: *Proc. ACL Workshop on Multiword Expressions*, pp. 33–40, 2003.

[Tur 08] G. Tur, A. Stolcke, L. Voss, J. Dowding, B. Favre, R. Fernandez, M. Frampton, M. Frandsen, C. Frederickson, M. Graciarena, D. Hakkani-Tür, D. Kintzing, K. Leveque, S. Mason, J. Niekrasz, S. Peters, M. Purver, K. Riedhammer, E. Shriberg, J. Tien, D. Vergyri, and F. Yang. "The CALO Meeting Speech Recognition and Understanding System". In: *Proc. IEEE Workshop on Spoken Language Technologies (SLT)*, pp. 69–72, 2008.

[Tur 10] G. Tur, A. Stolcke, L. Voss, S. Peters, D. Hakkani-Tür, J. Dowding, B. Favre, R. Fernández, M. Frampton, M. Frandsen, C. Frederickson, M. Graciarena, D. Kintzing, K. Leveque, S. Mason, J. Niekrasz, M. Purver, K. Riedhammer, E. Shriberg, J. Tien, D. Vergyri, and F. Yang. "The CALO Meeting Assistant System". *IEEE Trans. Audio, Speech and Language Processing*, Vol. 18, No. 6, pp. 1601–1611, 2010.

[Tur 11] G. Tur and R. De Mori, Eds. *Spoken Language Understanding: Systems for Extracting Semantic Information from Speech.* Wiley, Chichester, UK, 2011.

[Wahb 90] G. Wahba. *Spline Models for Observational Data.* Vol. 59 of *CBMS-NSF Regional Conference Series in Applied Mathematics*, SIAM, 1990.

[Wang 10] W. Wang, G. Tur, J. Zheng, and N. F. Ayan. "Automatic Disfluency Removal For Improving Spoken Language Translation". In: *Proc. IEEE Int'l Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 5214–5217, 2010.

[Well 00] J. C. Wells. *Longman Pronunciation Dictionary.* Longman, 2 Ed., 2000.

[Well 05] P. Wellner, M. Flynn, S. Tucker, and S. Whittaker. "A Meeting Browser Evaluation Test". In: *Proc. ACM SIGCHI Conference on Human Factors in Computing Systems (CHI)*, pp. 2021–2024, 2005.

[Wilk 62] S. Wilks. *Mathematical Statistics.* John Wiley, 1962.

[Witt 99] I. H. Witten, G. Paynter, E. Frank, C. Gutwin, and C. G. Nevill-Manning. "KEA: Practical Automatic Keyphrase Extraction". In: *Proc. ACM Conference on Digital Libraries*, pp. 254–255, 1999.

[Xie 09] S. Xie, D. Hakkani-Tür, B. Favre, and Y. Liu. "Integrating Prosodic Features in Extractive Meeting Summarization". In: *Proc. IEEE Workshop on Speech Recognition and Understanding (ASRU)*, pp. 387–391, 2009.

[Yang 07] S. Yang, H. Zhu, A. Apostoli, and P. Cao. "N-gram Statistics in English and Chinese: Similarities and Differences". In: *Proc. Int'l Conference on Semantic Computing*, pp. 454–460, 2007.

[Zah 10] U. Zäh, K. Riedhammer, T. Bocklet, and E. Nöth. "Clap Your Hands! Calibrating Spectral Subtraction for Dereverberation". In: *Proc. IEEE Int'l Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 4226–4229, 2010.

[Zech 02] K. Zechner. "Automatic Summarization of Open-Domain Multiparty Dialogues in Diverse Genres". *Computational Linguistics*, Vol. 28, No. 4, pp. 447–485, 2002.

[Zhan 04] L. Zhang, Y. Pan, and T. Zhang. "Focused Named Entity Recognition Using Machine Learning". In: *Proc. ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 281–288, 2004.

[Zhan 07a] J. Zhang, H. Chan, and P. Fung. "Improving Lecture Speech Summarization using Rhetorical Information". In: *Proc. IEEE Workshop on Speech Recognition and Understanding (ASRU)*, pp. 195–200, 2007.

[Zhan 07b] J. Zhang and P. Fung. "Speech Summarization Without Lexical Features for Mandarin Broadcast News". In: *Proc. Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL): Human Language Technologies (HLT)*, pp. 213–216, 2007.

[Zhan 09] X. Zhang, H. Huang, and Z. Liang. "The Application of CRFs in Part-of-Speech Tagging". In: *Proc. Int'l Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC)*, pp. 347–350, 2009.

[Zhao 91] Y. Zhao, H. Wakita, and X. Zhuang. "An HMM Based Speaker-Independent Continuous Speech Recognition System with Experiments on the TIMIT Database". In: *Proc. IEEE Int'l Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 333–336, 1991.

[Zhu 05] Q. Zhu, A. Stolcke, B. Chen, and N. Morgan. "Using MLP features in SRI's conversational speech recognition system". In: *Proc. European Conference on Speech Communication and Technology (EUROSPEECH)*, pp. 2141–2144, 2005.

[Zhu 06] X. Zhu and G. Penn. "Utterance-Level Extractive Summarization of Open-Domain Spontaneous Conversations with Rich Features". In: *IEEE Int'l Conference on Multimedia and Expo*, pp. 793–796, 2006.