Diss. ETH No. 17039

Improving Speech Recognition through Linguistic Knowledge

A dissertation submitted to the
SWISS FEDERAL INSTITUTE OF TECHNOLOGY
ZURICH

for the degree of
DOCTOR OF TECHNICAL SCIENCES

presented by
RENÉ BEUTLER
Dipl. Informatik-Ing. ETH
born February 3, 1975
citizen of Buchholterberg (BE), Switzerland

accepted on the recommendation of
Prof. Dr. L. Thiele, examiner
Prof. Dr. H. Strik, co-examiner
Prof. Dr. J. Nouza, co-examiner

2007

Seite Leer /
Blank leaf

Dedicated to Consti,
Käthi, Urs and Jörg

Seite Leer /
Blank leaf

Acknowledgments

This thesis strongly relies on the state-of-the-art linguistic system developed by Tobias Kaufmann. I want to thank him for the fruitful collaboration.

In particular I would like to thank Prof. Dr. Lothar Thiele and Dr. Beat Pfister for supervising and guiding my research, and Prof. Dr. Helmer Strik and Prof. Dr. Jan Nouza for their willingness to be the co-examiners of my thesis.

I would like to express gratitude to my parents for their support at every moment in life. Consti, thanks for all you have done, it really means a lot to me.

This work has been the supported by the Swiss authorities in the framework of COST 278. This support is gracefully acknowledged.

Seite Leer /
Blank leaf

Contents

List of Abbreviations

ASR	Automatic speech recognition
CFG	Context-free grammar
DCG	Definite clause grammar
GMM	Gaussian mixture model
HMM	Hidden Markov model
HPSG	Head driven phrase structure grammar
LM	Language model
LVCSR	Large vocabulary continuous speech recognition
NL	Natural language
NLP	Natural language processing
NLU	Natural language understanding
MFCC	Mel frequency cepstral coefficients
OOV	Out-of-vocabulary
SCFG	Stochastic context-free grammar
WER	Word error rate

Seite Leer /
Blank leaf

Notation

$\mathbb{E}$	Set of edges of a lattice or word graph, $\mathbb{E} = \{e_1, e_2, \ldots, e_{\|\mathbb{E}\|}\}$
$\mathbb{N}$	Set of nodes of a lattice or word graph, $\mathbb{N} = \{n_1, n_2, \ldots, n_{\|\mathbb{N}\|}\}$
G	Lattice or word graph
K	Length of word sequence
$P(\mathbf{X}\|W)$	Likelihood of observation sequence $\mathbf{X}$ given word sequence W (acoustic model)
$P(W)$	A-priori probability of word sequence W (language model)
$P_N(W)$	N-gram probability of word sequence W
S_i	State i of an HMM
T	Length of a sequence of feature vectors or observations
V	Vocabulary of a speech recognizer: $V = \{v_1, v_2, \ldots, v_{\|V\|}\}$
$\|V\|$	Number of words in vocabulary V
V^*	Set of all word sequences over the vocabulary V
$W, \hat{W}$	Word sequence $w_1 w_2 \ldots w_K$, most likely word sequence
$\hat{W}, \hat{W}^+$	Most likely word sequence without parser, most likely word sequence with parser
$\mathbf{X}$	Sequence of feature vectors or observations $\mathbf{x}_1 \mathbf{x}_2 \ldots \mathbf{x}_T$
a_{ij}	HMM state transition probability from state i to state j
$a_{ij}(d)$	HMM state transition probability from state i to state j given the duration d of the state i
$b_j(\mathbf{x})$	HMM observation probability of $\mathbf{x}$ given state j
c_α	Syntactical score of a full parse

c_β	Syntactical score of a phrase
c_γ	Syntactical score of a single word
e_i	i^{th} edge of a graph or of an edge sequence
ip	Word insertion penalty
n_i	i^{th} node of a graph or a sequence of nodes
t	Index of observation vector
w_i	i^{th} word of a word sequence
v_i	i^{th} word of vocabulary V
$\mathbf{x}_t$	Observation vector at time t
$\delta_t(j)$	Best score along a single path at time t which ends in state S_j and accounts for the first t observations
ϵ	Empty word, $\|\epsilon\| = 0$
γ	Posterior probability
λ	Language model weight
μ	Mean value
σ	Standard deviation
σ^2	Variance

Abstract

Todays speech recognizers use very little knowledge of what language really is. They treat a sentence as if it would be generated by a random process and pay little or no attention to its linguistic structure. If recognizers knew about the rules of grammar, they would potentially make less recognition errors.

Highly linguistically motivated grammars that are able to capture the deeper structure of language have evolved from the natural language processing community during the last few years. However, the speech recognition community mainly applies models which disregard that structure or applies very coarse probabilistic grammars.

This thesis aims at bridging the gap between statistical language models and elaborate linguistic grammars. The first goal is to introduce precise linguistic knowledge into a medium vocabulary continuous speech recognizer. The second goal consists of investigating the capabilities and limitations of qualitative language models to improve medium vocabulary continuous speech recognizers.

Two architectures are studied in-depth. The first is a novel architecture which integrates a non-probabilistic grammar into speech recognition based on a word spotter, an island chart parser for definite clause grammars and a strategy component. It uses morphological knowledge to improve performance and is able to properly treat German noun compound words.

The second architecture extends a speech recognizer by a rule-based component in a way such that any improvement can be clearly attributed to that component and therefore to the linguistic knowledge. A speech recognizer creates word lattices and at the same time provides

the baseline word error rate. The word lattices are subsequently processed by a natural language processing module. A score derived from the syntactic structures found by a parser is used to rescore the word lattice such that grammatical phrases are slightly favoured. However, we do not require that the utterances to be recognized are grammatical. By comparing the word error rate of the enhanced system with the baseline word error rate we can directly quantify the benefit of our approach. The lattice parsing system is based on a linguistically motivated HPSG grammar which was developed by Tobias Kaufmann in a separate project.

We provide evidence for the first time that a statistically significant improvement of recognition accuracy on a medium vocabulary continuous speech recognition dictation task due to a non-stochastic hand-written grammar is possible over a competitive baseline recognizer. The baseline recognizer uses cross-word triphone hidden Markov models and a 4-gram language model. The relative reduction of the word error rate due to the parser is 27.0% which is statistically significant at a level of 0.001.

Our results suggest that a sophisticated qualitative language model is complementary to an N-gram model. A grammar is best at modeling long-distance dependencies and hierarchical structures, while an N-gram model captures local and lexical dependencies.

The main limitation of the lattice parsing approach is the out-of-vocabulary (OOV) rate. For increasing OOV rates the relative improvement due to parsing decreases. Missing words break up the syntactic structure of a sentence and less constraints can be imposed.

Kurzfassung

Heutige Spracherkennungssysteme verfügen über relativ wenig Sprachwissen. Sie behandeln Äusserungen, als ob diese von einem Zufallsprozess erzeugt worden wären und ignorieren weitgehend die linguistischen Strukturen, die der Sprache zu Grunde liegen. Wenn ein Spracherkenner die grammatikalischen Regeln einer Sprache kennen würde, würde er womöglich weniger Fehler machen.

Auf dem Gebiet der Computerlinguistik wurden Fortschritte erzielt, so dass heute Grammatiken zur Verfügung stehen, welche die Regeln von natürlicher Sprache und deren Strukturen recht präzise abbilden. In der Spracherkennung werden diese aber entweder gar nicht oder nur in stark vereinfachter Form angewendet.

Diese Doktorarbeit verfolgt das Ziel, regelbasiertes grammatikalisches Wissen in den Spracherkennungsprozess einzubringen, sowie die Möglichkeiten und Grenzen aufzuzeigen, dadurch die Erkennungsleistung der kontinuierlichen Spracherkennung zu verbessern.

Zwei Architekturen werden genauer betrachtet. Die erste, neuartige Architektur integriert regelbasiertes Wissen, indem ein Wordspotter mit einem Insel-Chart-Parser für Definite-Klausel-Grammatiken kombiniert wird, welche von einer Strategiekomponente gesteuert werden. Dank der Verwendung morphologischen Wissens kann die Effizienz des Verfahrens gesteigert werden, und deutsche Komposita können korrekt behandelt werden.

Die zweite Architektur erweitert einen Spracherkenner mit einer regelbasierten linguistischen Komponente so, dass jede Änderung der Erkennungsleistung eindeutig dieser Komponente und damit dem linguistischen Wissen zugeordnet werden kann. Ein Spracherkenner er-

zeugt Worthypothesengraphen und liefert gleichzeitig die Referenzerkennungsrate. Die Worthypothesengraphen werden aufgrund der von einem Parser gefundenen syntaktischen Strukturen neu bewertet, so dass grammatikalisch korrekte Phrasen leicht bevorzugt werden. Die linguistische Komponente basiert auf einer HPSG-Grammatik und einem Parser, der in einem separaten Projekt von Tobias Kaufmann entwickelt wurde.

Wir erbringen zum ersten Mal den Nachweis, dass eine signifikante Verbesserung der Erkennungsgenauigkeit aufgrund einer handgeschriebenen Grammatik für kontinuierliche Spracherkennung mit mittelgrossem Vokabular möglich ist. Das Referenzsystem verwendet wortübergreifende Triphon-Hidden-Markov-Modelle und ein 4-Gram-Sprachmodel. Die gemessene relative Verbesserung der Wortfehlerrate beträgt 27% und ist statistisch signifikant (Signifikanzniveau 0.001).

Unsere Resultate legen nahe, dass eine fortgeschrittene Grammatik komplementäres Wissen zum N-Gram-Model enthält. Die Stärke der Grammatik liegt bei der Modellierung von nichtlokalen Abhängigkeiten und hierarchischen Strukturen, während das N-Gram-Modell lokale und lexikale Abhängigkeiten beschreibt.

Der wichtigste Einfluss auf die Verbesserungsmöglichkeiten einer Grammatik auf die Erkennungsleistung ist die out-of-vocabulary (OOV) Rate. Mit grösser werdender OOV-Rate nimmt die relative Verbesserung aufgrund des Parsings ab. Die fehlenden Wörter zerstören die syntaktische Struktur des Satzes, wodurch weniger syntaktische Einschränkungen gemacht werden können.

Chapter 1

Introduction

1.1 Problem Statement and Aim

Today's speech recognizers use very little knowledge of what language really is. They treat a sentence as if it would be generated by a random process and pay little or no attention to its linguistic structure. If recognizers knew about the rules of grammar, they would potentially make less recognition errors.

The predominantly used N-gram language model assumes that a word is only influenced by a few preceding words, typically one or two. However, natural language is more precisely described in terms of hierarchical structures and dependencies between constituents in order to account for longer-distance constraints [Moo99].

There is clearly a trend to extend the language models to make more use of the structure of a language [LST92, Cha97, Ros00, Roa01, XCJ02, HJ03, WH03, CCP04]. However, these models are still based on very course grammars which are extracted automatically from syntactically annotated corpora such as the *Penn Treebank* [MSM94].

While the speech community mainly applies models which disregard the structure of language or applies very coarse probabilistic grammars, highly linguistically motivated grammars that are able to capture the deeper structure of language have evolved from the natural language

processing community during the last few years [Mül99, Net96, Hau00, Gör88].

This thesis aims at bridging the gap between statistical speech recognition and elaborate linguistic grammars. The first goal is to introduce precise linguistic knowledge into a medium vocabulary continuous speech recognizer. The second goal consists of investigating the capability of qualitative language models to improve medium vocabulary continuous speech recognition systems for general texts.

The idea to use grammars for speech recognition is not new. In fact, it was already present in the early recognizers [Low76, EHRLR80]. However, these systems are either restricted to simple language and small vocabulary (e.g. commands or simple questions [MAD+95]), are very domain specific (like scheduling meetings [Wah00]), or they aimed at improving natural language understanding rather than word accuracy [NBKvN97, ZGG+91, CR89].

This thesis goes beyond previous work in that the task is more general. We allow general language which is not restricted to a certain domain so that domain specific knowledge cannot be used. So far, no significant improvement of recognition accuracy was reported due to using a linguistically motivated grammar for a task comparable to ours.

1.2 Hypothesis

Based on the experience that a good language model is important for high recognition accuracy, and the fact that current LMs do not use all information about language available, we expect that linguistic sophistication leads to an improved recognition accuracy. The working hypothesis is thus as follows:

> Adding a rule-based linguistic sub-system to a speech recognizer which takes the structure of language into consideration improves its accuracy.

The central question of this thesis is how rule-based knowledge can improve recognition accuracy. This question can be broken down into three basic issues.

1. How can the rule-based knowledge be incorporated into the statistical framework of a speech recognizer?
2. How much can a knowledge based language model improve the word error rate, and by which factors is it influenced?
3. What are the limitations of the approach?

1.3 Scientific Contribution

Two architectures are investigated. The first is a novel architecture which integrates a non-probabilistic grammar based on word spotting and island chart parsing. It uses morphological knowledge to improve performance and properly treat German noun compound words. It is different from [Nak89], which also uses word spotting and context-free parsing, in that it allows bi-directional interaction, bi-directional parsing and the application of morphological knowledge.

The second architecture is based on lattice parsing. We provide evidence for the first time that a statistically significant improvement of recognition accuracy on a medium vocabulary CSR dictation task due to a non-stochastic hand-written grammar and lattice parsing is possible over a competitive baseline recognizer.

1.4 Structure of the Thesis

Chapter 2 identifies the weaknesses of the prominent N-gram language model and motivates the incorporation of more linguistic knowledge. Chapter 3 surveys different architectures that allow to use rule-based knowledge in ASR. We propose our own architecture based on word-spotting and parsing in Chapter 4. The lattice parsing architecture applied in Chapter 5 is especially suited to provide evidence that rule-based knowledge can improve LVCSR accuracy. We measure the gain of the lattice parsing language model in the experiments in Chapter 6. Chapter 7 concludes the thesis by giving answers to our three main questions. The appendix contains the test sentences that were used in the experiments and a short overview about the HPSG grammar formalism, which was used in the lattice parsing approach.

Seite Leer /
Blank leaf

Chapter 2

Language Modeling

2.1 Speech Recognition

The aim of automatic speech recognition is to enable a machine to recognize what a human speaker said. A machine that can "hear" can be helpful in many ways. The user can control the machine by voice, which keeps his hands and eyes free for other tasks, it can save the user from typing vast amounts of text by simply dictacting it, the recognized speech can be used to index speech such as broadcast news which allows efficient document retrieval, or the system may even understand what the user intends to do or answer his questions. These examples illustrate that speech recognition is an important aspect of improving human-machine interfaces and thus making machines more usable and user friendly.

Speech recognition research has started more than five decades ago. In the late 40's the invention of the sound spectrograph made it possible to visualize acoustic signals [Pot45]. It was believed, that as soon as the spectrum of a speech signal could be computed fast enough, the speech recognition problem could be easily solved. Although thousands of researchers around the world worked on the problem for more than half a century, the task must be still considered to be unsolved. In difficult acoustical environments machines perform orders of magnitude worse than humans [Lip97].

How was such a misinterpretation possible? On one hand the speech recognition problem is often largely underestimated because it is so natural for human beings to listen to others and understand them. We are not aware of the tremendous amount of variability present in a speech signal. We can understand people we never met before, we are able to recognize a huge amount of different words in continuous speech, and we are even able to understand ungrammatical utterances or expressions we have never heard before. We are able to perform so well because we include a wide variety of knowledge sources: we have prior knowledge about the syntax and semantics of a language, we can derive the meaning of new words by analogy, we use situational clues like the course of a dialogue and we have access to all experiences we made in our live and all knowledge about the world we have. Machines can not keep up with that.

There are a lots of other reasons why speech recognition is inherently difficult. The next section discusses a selection of them. They point up why speech recognition must make use of knowledge about a language, and they motivate the approach that was taken in this thesis.

2.1.1 The Ambiguity of Speech

Written language consists of a sequence of discrete symbols, the letters of the alphabet. These symbols are uniquely identifiable and do not interact. The boundaries of a word are well defined as words are separated by spaces. This is still true for the smallest linguistic elements of speech, the phonemes. In written form, these are discrete symbols as well. However, the situation changes dramatically when we are going from written form to spoken form, or more specifically if we look at a speech signal.

A speech signal contains a tremendous amount of variability from several sources. There is no one-to-one relationship between letters or phonemes and their physical realisation in a speech signal:

- The acoustic realisation of a phone largely depends on the individual speaker properties such as sex, vocal tract shape, origin, dialect tone coloration, speaking rate, speaking style (normal, whispering, shouting), mood and health.

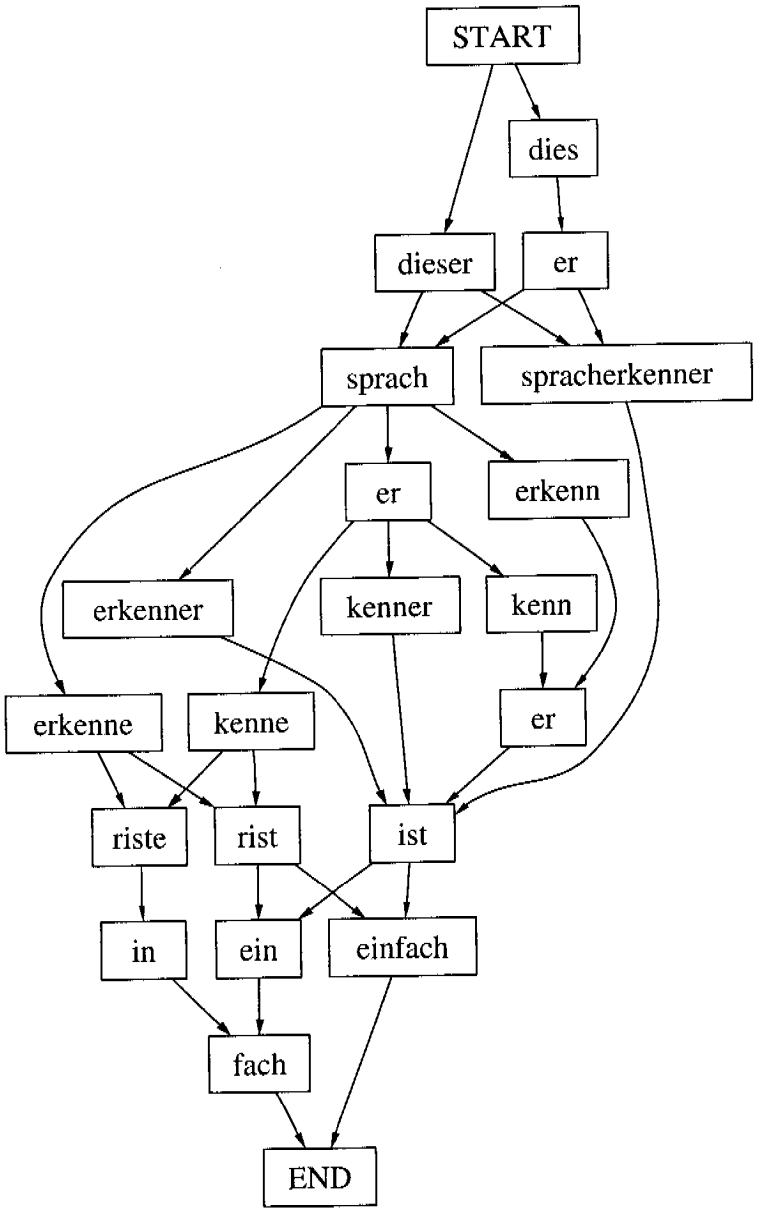


Figure 2.1: *The effect of missing word boundaries. If the spaces in the German sentence "dieser spracherkenner ist einfach" (engl. "this speech recognizer is simple") are removed, the resulting letter sequence "diesterspracherkenneristeinfach" can be split in 32 ways into a sequence of German words. The original sequence can only be reconstructed by means of syntactic and semantic knowledge.*

- The pronunciation of a particular phone is influenced by its phonetic context (coarticulation). This influence may span several phones and even syllable and word boundaries.
- Allophonic variants and phoneme variations. The phones [r] and [ɐ̯], e.g., are allophonic variants of the phoneme /r/. A phoneme variation that often occurs in Swiss standard German is the replacement of /z/ with /s/, e.g., /sɜks/ instead of /zɜks/ for the digit "6".
- The signal is altered by the room characteristics like reverberation, the microphone characteristics, signal coding and compression, as well as background noise.

Given these facts it is clear that the feature space regions of speech units like phones do largely overlap and that there is always uncertainty about the presence of such units in a speech signal. Even if the phone sequence could be recognized perfectly, its conversion to the corresponding textual representation has to deal with a high amount of ambiguities that arise from unknown word boundaries in continuous speech (cf. Fig. 2.1) and the existence of words that are pronounced the same way but are differently spelled (homophones, e.g. /baɪ/ is the pronunciation of *buy*, *by*, and *bye*).

In order to convert speech to text, a description of the acoustic events of speech alone is not sufficient. To resolve ambiguities, knowledge about the language at hand is indispensable and plays a very important role in speech recognition. The claim to use linguistic knowledge in speech recognition is justified from a mathematical point of view as well, as explained in the next section.

2.1.2 MAP Recognizer

From the perspective of information theory speech recognition can be considered as decoding problem (cf. Fig. 2.2). A speaker has a message W in his mind. He pronounces the message and transforms it by means of his vocal chord and vocal tract into a signal s. The signal is transmitted to the recognizer. The digitized speech signal is converted by the feature extraction into a sequence of feature vectors $\mathbf{X}$. The speech production and the feature extraction together form the

acoustic channel. The acoustic channel is noisy due to influences of the speaker, the transmission channel and the feature extraction. The task of the linguistic decoder is to make an optimal estimate $\hat{W}$ of the uttered message W given the feature sequence $\mathbf{X}$.

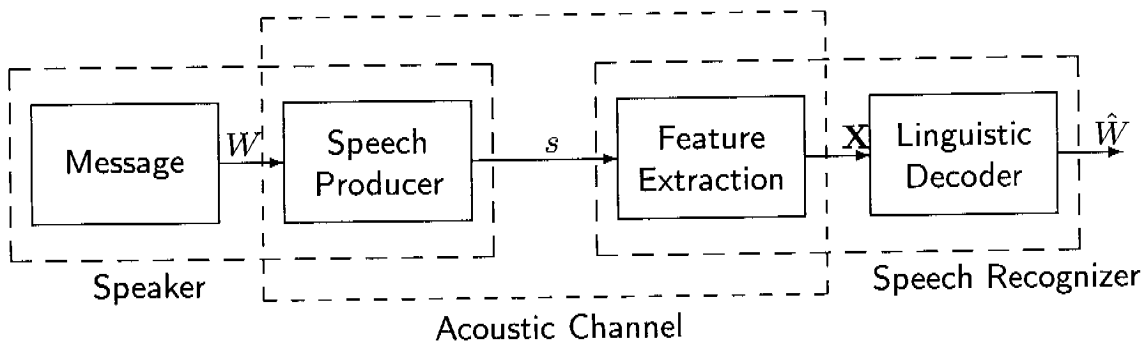


Figure 2.2: *Jelinek's source-channel model of speech recognition, according to [Jel98].*

How can the optimal word sequence $\hat{W}$ be determined? According to the Bayes decision theory the following decision rule must be applied:

> To minimize the error probability choose the word sequence $\hat{W}$ among all possible word sequences V^* of a given vocabulary V which has the highest posterior probability given the feature sequence $\mathbf{X}$:

$$\hat{W} = \underset{W \in V^*}{\operatorname{argmax}} P(W|\mathbf{X}) \; . \tag{2.1}$$

This decision rule is known as maximum-a-posterior rule or short MAP rule.[1] The proof of optimality of this decision strategy is given in [DH73, p. 13]. Using the Bayes rule, the posterior probability can be written as

$$P(W|\mathbf{X}) = \frac{P(\mathbf{X}|W) \cdot P(W)}{P(\mathbf{X})} \; . \tag{2.2}$$

[1] The MAP-rule minimizes the probability that the recognized utterance is wrong. An utterance is not correctly recognized if a single word or several words are incorrect. Hence, the rule does not minimize the number of incorrect words but rather the number of not fully correct utterances.

Since the probability of the feature sequence $P(\mathbf{X})$ is fixed for a given utterance it can be dropped from the decision process. Thus, the MAP-rule can be written as

$$\hat{W} = \underset{W \in V^*}{\operatorname{argmax}} P(\mathbf{X}|W) \cdot P(W) \ . \tag{2.3}$$

Eq. (2.3) reveals two important parts of the speech recognition process:

1. $P(\mathbf{X}|W)$ is the likelihood that we will observe the feature sequence $\mathbf{X}$ given that W was uttered. The likelihood is estimated by an *acoustic model*, e.g. a hidden Markov model (HMM).

2. The prior probability $P(W)$ is independent of the observed feature sequence. It is provided by a *language model*.

2.2 Language Models in General

A language model (LM) is a collection of prior knowledge about a language. This knowledge is independent of an utterance to be recognized. It therefore represents *previous knowledge* about language and the expectations at utterances. Knowledge about a language can be expressed in terms of which words or word sequences are possible or how frequently they occur.

As explained in the last sections, the need for a language model in a speech recognizer arises from the variability of the speech signal, missing word boundaries and homographs. It is needed to resolve the ambiguities which the acoustic model is not able to handle. It is also mathematically justified by the MAP decision rule.

Language models can be divided into two groups. The criterion is whether the model is data driven or expert-driven:

- **Statistical language models**: If the model is based on counting events in a large text corpus, for example how frequent a certain word or word sequence occurs, the model is called to be a *statistical language model*. Such a model describes language as if utterances were generated by a random process. It is therefore also known as *stochastic language model*.

- **Knowledge based models**: If the knowledge comes from a human expert the model is called *knowledge-based language model.* Such linguistic knowledge could for example include syntax, the conjugation of verbs or the declension of adjectives. The basis of such a model does not rely on counting observable events, but rather the understanding of the mechanisms, coherences and regularities of a language. If this knowledge is defined by rules, such models are also called *rule-based models.*

Since statistical language models are the most commonly used models, they will be discussed first. The next section describes the key idea, the advantages and the limitations of statistical LMs. The limitations will motivate the use of knowledge based models and the approach that was taken in this thesis.

2.3 Statistical Language Models

A statistical LM aims at providing an estimate of the probability distribution $P(W)$ over all word sequences W. It must be able to assign a probability to each possible utterance. By applying the multiplication law, the probability of a word sequence can be decomposed into a product of probabilities:

$$P(w_1 w_2 \dots w_K) = \prod_{k=1}^{K} P(w_k | w_1 w_2 \dots w_{k-1}) \tag{2.4}$$

$P(w_k|w_1 w_2 \dots w_{k-1})$ is the probability that w_k occurs given that the words $w_1 w_2 \dots w_{k-1}$ were observed previously. The preceding words of w_k are also referred to as *history* and denoted as $h_k = w_1 w_2 \dots w_{k-1}$. Eq. (2.4) states that the probability that W was uttered is the product of the probability that w_1 was spoken, times the probability that w_2 follows w_1, etc., times the probability that w_K was uttered given that all its preceding words were uttered.

The conditional probabilities $P(w_k|h_k)$ must be estimated on large amounts of texts related to the recognition task at hand. The maximum likelihood estimate can be computed by the relative frequency

approach:

$$\tilde{P}(w_k|w_1 w_2 \ldots w_{k-1}) = \frac{\text{freq}(w_1 w_2 \ldots w_k)}{\text{freq}(w_1 w_2 \ldots w_{k-1})} , \tag{2.5}$$

where freq(e) denotes the frequency of event e in the training corpus. In order for the estimate to be appropriate, freq($w_1 w_2 \ldots w_k$) must be sufficiently large.

The number of frequencies that must be counted and stored for this model is prohibitive. The number of different strings of length K composed of words of a vocabulary of size $|V|$ is $|V|^K$. Even for a vocabulary size of $|V| = 1000$ words and a string length of $K = 10$ the number of different strings and thus the number of frequencies to determine is $1000^{10} = 10^{30}$. Besides, the longer the conditioning history h_k gets, more and more strings $w_1 w_2 \ldots w_k$ will never occur in the training data.

An obvious solution is to limit the length of the histories by assuming that the probability of each word does not depend on all previous words, but only on the last $N-1$ words:

$$P(w_k|w_1 w_2 \ldots w_{k-1}) \approx P(w_k|w_{k-N+1} w_{k-N+2} \ldots w_{k-1}) , \tag{2.6}$$

which leads to the so called *N-gram language model.*

2.3.1 N-Gram Language Models

An N-gram language model assumes that the probability of a word is not influenced by words too far in the past. It considers two histories to be equivalent, if they have their last $N-1$ words in common. The $N-gram$ probability of a word sequence is thus defined by

$$P_N(W) = \prod_{k=1}^{K} P(w_k|w_{k-N+1} w_{k-N+2} \ldots w_{k-1}) \tag{2.7}$$

The most frequent choices are $N = 1$ (unigram), $N = 2$ (bigram) and $N = 3$ (trigram):

$$P_3(W) = P(w_1) \cdot P(w_2|w_1) \cdot \prod_{k=3}^{K} P(w_k|w_{k-2} w_{k-1}) . \tag{2.8}$$

With decreasing N the approximation gets coarser and the space requirements decrease.

The N-gram is currently the most widely used language model in speech recognition. Virtually all state-of-the-art large vocabulary continuous speech recognizers use purely statistical language models, most commonly N-gram models [SBB+00, BAHU+02, GLA02, HWEP00]. The simplicity of the model, its easy integration into the decoding process and its ability, at least to some extent, to take semantics into account, contribute to its success. It is also attractive because it is completely data driven, which allows engineers to apply it without requiring detailed knowledge about the language at hand.

However, despite of its success, the word N-gram language model has several flaws:

- **False conditional independence assumption.** The N-gram model assumes that a word is only influenced by its $N-1$ preceding words and that it is independent from other words farther in the past. It assumes that language is generated by a Markov process of order $N-1$, which is obviously not true [Ros00, Cho56].

- **Saturation.** The quality of N-gram models increased with larger amounts of data becoming available online. However the improvement is limited due to saturation [DUHW05]. Bigram models saturate within several hundred million words, and trigrams are expected to saturate within a few billion words [Ros00].

- **Lack of generalization across domains.** N-grams are sensitive to differences in style, topic or genre between training and test data. The quality of an N-gram model trained on one text source can degrade considerably when applied to an other text source, even if the two sources are very similar. Rosenfeld computed a trigram model on Dow-Jones (DJ) newswire texts. He compared the model's perplexity[2] on Associated Press (AP) newswire texts to the perplexity of DJ texts from the same time period. The perplexity of the AP data was twice that of the DJ data [Ros96, p. 220].

[2]The perplexity denotes the average number of words a speech recognizer has to choose from. Lower perplexities often correlate with lower recognition error rates.

This loss of modeling quality is even more pronounced if the mismatch between two domains is large. [WMH00] reports a perplexity of about 100 for the standard DARPA NAB word trigram LM for Wall Street Journal texts compared to a perplexity of more than 1000 for a personal information management domain.

- **Lack of extensibility.** Given an N-gram model it is difficult or even impossible to derive a new model which has additional words. The information contained in the model is not helpful to derive N-grams containing new words. Grammars, on the other hand, are able to generalize better because they are based on the underlying linguistic regularities. Adding a new word basically requires to know its syntactic class. It should be noted that the same is true for class-based N-grams [BDPd+92], if the classes are linguistically motivated and defined manually.

- **Language specific properties.** German has a number of properties which pose additional difficulties to N-grams. German is a compounding language and has a highly inflective nature. A compound is a complex expression in a single word, such as the concatenation of two or more nouns. Compounding and the high number of morphological endings result in very large vocabularies, which in turn aggravates the problem of data sparseness [HRB02].

- **Data Sparseness.** The severity of the data sparseness problem can be illustrated by the following findings in the context of Czech broadcast news transcription. Due to the high amount of inflectional word forms the vocabulary had a size of 300k words. A bigram language model was estimated on a 400 million words corpus. When the test sentences were added to the language model training corpus, the word error rate decreased 3% absolute.[3] This is remarkable, since the amount of added words (a few hundred words only) compared to the original corpus size of 400 million words is rather small. If data sparsity was not an issue, we would not expect such an impact by such a small change.

Grammaticality is not a strong constraint in the N-gram language models as the next section explains.

[3] Personal communication with Jan Nouza (Technical University of Liberec), April 2006.

2.3.2 Structured Language Models

N-grams fail on constructions like in the the following example sentence:

The dogs chasing the cat bark.

The trigram probability $P(\text{bark}|\text{the cat})$ will be very low because on one hand cats seldom bark, and on the other hand because a plural verb (bark) is unexpected after a singular noun (cat). Nevertheless this sentence is completely sound. The verb (bark) must agree in number with the noun (dogs) which is the head of the preceding noun phrase, and not with the noun that linearly precedes it.

One might be tempted to believe that increasing the length of the window on which the N-gram probabilities are conditioned might increase the model's ability to represent grammaticalness. However, [Cho56] argues the converse: With increasing N the model will exclude an increasing number of grammatical sentences, since longer histories are more and more unprobable due to data sparsity. Therefore, we must take the hierarchical structure of the sentence into account (cf. Fig. 2.3) to properly solve the problem.

Doing so has proven to be helpful. [BFHM98] used humans as post-processor of an ordered list of the ten most likely hypotheses of a speech recognizer. Either the test persons were restricted to choose one out of ten hypotheses, or they were allowed to freely edit them. The participants were able to improve the accuracy in both cases and on different corpora. Many of the gains involved linguistic proficiencies such as number agreement and tense agreement. The inclusion of agreement constraints into a CFG-based language model has been found to improve performance in therms of both word error rate and semantic error rate on different command and control tasks [RGH$^+$01].

Language models which take the hierarchical structure of a sentence into account are called *structured language models*. They rely on some sort of parsing. Most often a statistical parser is used which is trained on syntactically annotated text material like the English Penn Treebank [MSM94]. Indeed, there is clearly a trend to extend the N-gram models to make more use of the structure of a language [LST92, Cha97, Ros00, Roa01, XCJ02, HJ03, WH03, CCP04]. Some of

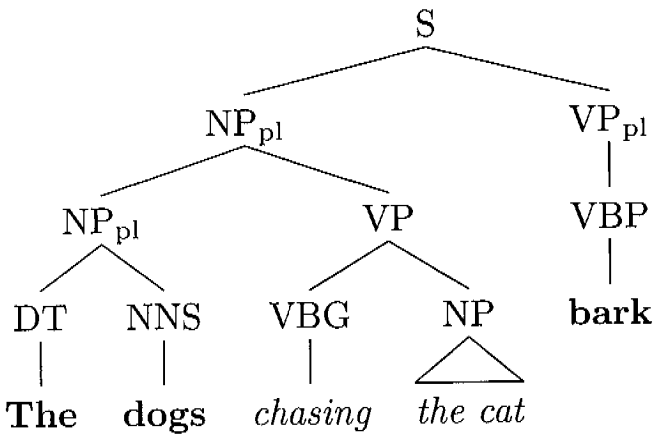


Figure 2.3: *Syntax tree of the sentence "The dogs chasing the cat bark".*

these models were applied to speech recognition and showed superior performance compared to an N-gram model.

2.4 Knowledge-Based LM

Undoubtedly, written and spoken language follow certain rules such as spelling and grammar. In a knowledge-based approach these rules are collected by experts (linguists) and are represented as a hand-crafted formal system. This system allows to decide if a sentence belongs to the language defined by the rules, and if that is the case, to derive its syntactic structure. The knowledge is explicitly available.

2.4.1 Terminology

The term *grammar* has a general and a technical meaning, referring to a natural language or a formal language, respectively. For clarity the following terms are defined and used consistently throughout this thesis:

- **Grammatical vs. ungrammatical.** These terms are used in conjunction with the grammar of a real world natural language. The grammar of a human language is elusive since it evolves over time and may be regionally different. Although the syntax of a language may be standardized by reference books, there may be sentences which cannot clearly be attributed to be grammatical or ungrammatical.

- **Intra-grammatical vs. extra-grammatical.** These terms are used to denote whether a sentence belongs to a formal language defined by an artificial grammar which is only the model of the grammar underlying a natural language. In contrast to real natural languages it is always well defined whether a sentence is intra-grammatical or extra-grammatical since the language is uniquely defined by a set of rules.

- **Non-grammatical and out-of-grammar.** These terms are used for sentences that are ungrammatical, extra-grammatical or both. They are used whenever it is not important to distinguish between ungrammaticality and extra-grammaticality.

- **Qualitative LM vs. quantitative LM.** A *qualitative language model* decides whether a sentence is intra-grammatical or extra-grammatical. This decision is binary, the model cannot assign a value or probability to a sentence. A *quantitative LM* assigns each conceivable sentence a value, it is therefore also denoted as *numerical LM*. If the assigned value satisfies the properties of a probability, the model is called *probabilistic LM* or *stochastic LM*.

2.4.2 Structured vs. Knowledge-Based LMs

The statistical parsers mentioned in the last section are somewhere in between a pure data-driven and a pure knowledge based approach. They are data driven because they "learn" the rules that are implicitly present in a large amount of training data. However, the successful approaches require the training data to be syntactically annotated. The syntactical annotation of the training corpus requires a significant amount of expert knowledge. The annotation is done by either human annotation or using a parser with semi-automatic disambiguation

[BHK+97, BDH+02]. Although the method itself is data-driven, expert knowledge and human interaction is still required to prepare the training data. From a linguistic point of view, the statistical grammars are still quite coarse.

In contrast to a statistical LM, no training data is needed for a knowledge-based system. This can be advantageous if no or only a small amount of (annotated) data is available from a specific domain. At the same time this means that the lexicon can be easily extended.

The successful applications of statistical parsing as language model [CCP04, HJ03, XCJ02, Roa01] did not have to deal with domain mismatch since they are trained on the Penn Treebank and tested on the DARPA '93 HUB-1 corpus, which both consist of sentences taken from the Wall Street Journal. It is not clear, however, how good the models generalize to other domains. At least for trigrams a mismatch was reported even on the same domain (newspapers text) for different sources (newswires) [Ros96, p. 220].

2.4.3 Challenges

The knowledge based approach faces several problems. One is of course the difficulty to build a formal system which appropriately reflects the phenomena of a natural language; however, this is not within the scope of this thesis.

The main problem that a speech recognizer has to deal with is the binary nature of a qualitative language model. If no appropriate measures are taken, the system is only capable of recognizing intra-grammatical utterances. This is quite a strong limitation, since a recognizer should be able to transcribe extra-grammatical utterances as well.

The lack of frequencies of a purely rule-based system is disadvantageous if the recognizer has several hypotheses to choose from which are all intra-grammatical. For example, the sentences "How to recognize speech?" and "How to wreck a nice beach?" are both syntactically correct, however the first is a-priori more likely and should be preferred.

Chapter 3

Integration of Linguistics

3.1 Architectures

This chapter reviews the architectures that were applied in the literature to introduce more linguistic knowledge into speech recognition. Two related projects which are similar to the work presented in this thesis are highlighted as well.

This review is part of the answer of the first question, how rule-based knowledge can be incorporated to the statistical framework of a speech recognizer. In the subsequent chapters, two architectures are further investigated. The first one is a novel approach based on word spotting and island chart parsing, the second one is known from natural language understanding systems and is based on lattice parsing.

The chapter starts with a literature review and closes with the motivation for the choice of the two architectures which are studied more in-depth.

3.1.1 N-grams Derived from a Statistical Grammar

Stochastic grammars have the advantage that they typically have much less parameters than an N-gram model. Stochastic grammars can thus be more reliably estimated from sparse data than N-grams. However, N-grams can be more easily integrated into the decoder without re-

quiring a parser. The idea is therefore to combine the advantage of reliably estimating the parameters of a stochastic grammar with the ease of integration of N-gram models. This is accomplished by estimating N-gram probabilities from a stochastic grammar instead of using the N-gram counts of sparse data.

[ZGG+91] builds a word-pair grammar by randomly generating sentences from a stochastic context-free grammar (SCFG). [SS94] describes an algorithm to compute the distribution of N-grams for a probabilistic language given by a SCFG in closed form. [WSH04] converts the SuperARV LM (a sort of class-based LM) to a word N-gram in ARPA LM format.

3.1.2 Parsing and Stack Decoding

[God92] computes word transition probabilities by conditioning the probability of the next word w_k on the top N grammar symbols on the stack of a shift-reduce parser after parsing the preceding words $w_1 w_2 \ldots w_{k-1}$:

$$P(w_k|w_1 w_2 \ldots w_{k-1}) \approx P(w_k|G_{M-N+1}, \ldots, G_M) \qquad (3.1)$$

where G_M is the grammar symbol on the top of the stack. The parser is integrated into an acoustic stack decoder. The decoder removes the top hypothesis from the priority queue. The parser analyses the word string of the current hypothesis and computes the transition probability for the next-word candidates given by Eq. (3.1). The hypothesis is extended by each candidate, combined with the acoustic score and added to the priority queue.

[ZGG+91] uses the prediction capability of the probabilistic natural language processing component TINA [Sen89] to propose words to extend partial paths of a stack decoder.

3.1.3 Dynamic Generation of Networks

Including natural-language constraints into the decoder can be desirable for two reasons: First, decoding can be more efficient due to the reduced search space, and second, it may improve recognition accuracy. The advantage is that undesired, extra-grammatical sentences can be

ruled-out early and that low scored intra-grammatical sentences can be saved from being pruned away. To include a grammar into a Viterbi decoder it must be possible to process the grammar left-to-right as the Viterbi-algorithm runs time-synchronously.

If the grammar is regular, it can be modeled by a finite state automaton and directly integrated into the recognition network of an HMM recognizer. Some natural language phenomena can not be described in terms of regular grammars or are more elegantly formulated by a context-free grammar [SW97, p. 27]. It is not feasible to compile CFGs into a static, finite state transition network because the number of states could be unmanageably large or infinite.

However, due to pruning only a part of the state transition network is active at each point in time, therefore a CFG can be realized as a network by dynamically extending the necessary part of the finite state network [MM90, Dup93].

[MPM89] incrementally extends the recognition network of a Viterbi decoder by a NL parser and a unification-based CFG. The recognition network is generated on the fly, by expanding the state transitions of an ending word into all words which can follow according to the grammar. It does so by predicting terminal symbols in a top-down manner; non-terminal symbols on the right-hand-side of context-free rules are expanded until a terminal is found. Therefore, the grammar must not be directly or indirectly left-recursive, which can be guaranteed if the grammar is given in the *Greibach normal form*[1].

The dynamic approach was extended by a probabilistic component in [JWS+95]. It uses a SCFG to compute a follow set and word transition probabilities for a given prefix string. If the prefix string is parseable the SCFG is used to compute the probability distribution of possible following words. If the string cannot be parsed, the system falls back to bigram probabilities instead.

3.1.4 Predict and Verify

The idea behind *predict and verify* is very similar to the dynamic generation of partial grammar networks. The main difference is that in

[1] A grammar is in Greibach normal form if it is ϵ-free and each production is of the form $A \rightarrow a\alpha$, where a is a terminal and α is a string of nonterminals, possibly empty.

the dynamic generation approach the parser is driven by the HMM decoder, while in the predict and verify approach the emphasis is put on the parser which drives the recognition process. It is based on predicting the next word or the next phone in a top down manner and is also called *analysis by synthesis*. A word or a phone is assumed to be present if it's maximal likelihood over all possible ending points is larger than a threshold.

In [KTS89] an LR parser and a context-free grammar were used to predict the next phone, which was verified by a phone HMM. The algorithm is not time synchronous. A time synchronous strategy which includes a parser and a unification grammar to top-down predict the next word is described in [HW94].

A far more complex system within this paradigm is Hearsay-II [EHRLR80], which applies various knowledge sources at different levels that communicate through a so called *blackboard*.

3.1.5 Word Spotting and Parsing

The system described in [KS89] uses a word spotter to detect keywords as islands in small vocabulary continuous speech. From these islands the system expands the islands by verifying neighboring words which are predicted by a word pair grammar.

[Nak89] spots words in a speech signal and creates a list of word hypotheses. A context-free grammar is then used to concatenate the word hypotheses to partial sentences. The sentences are related to electronic mail.

3.1.6 N-Best Filtering and N-Best Rescoring

The approaches described so far had in common that the linguistic knowledge was introduced into the decoder, which can be problematic when complex knowledge sources are applied. For example, a complex knowledge source might be computationally very expensive, or it may require that a whole sentence is available.

In the *N-best filtering* approach a speech recognizer enumerates the N-best recognition hypotheses, and the first hypothesis which is fully accepted by a grammar or a semantic module is chosen. An example for

syntactical filtering is [ZGG+91] while semantical filtering was applied in [CR89].

If a numerical language model is available *N-best rescoring* can be applied [OKA+91, God92, XCJ02]. The recognition score is combined with a natural language score and the hypothesis with the best overall score is selected.

The computational demand for the exact enumeration of the N-best recognition hypotheses increases linearly with N for the best-known method [CS89]. For an approximative enumeration the computational costs increase only by a small factor independent of N [SA90]. Some recognizer implementations require N to be known in advance [SA90], others can incrementally generate new hypotheses on demand [Pau89].

The advantage of the N-best approach is its simplicity. However, the computational demand to linguistically process the N-best list increases linearly with N which is a disadvantage for large values of N.

3.1.7 Word Lattice Parsing

Compared to N-best lists, word graphs are a more compact representation [JHJ98]. When comparing two consecutive utterances of an N-best list, they typically differ in only a few words, i.e. large parts are identical. While identical word sequences of two utterances occur in both entries of the N-best list, they share the same nodes in a word graph. The word graph representation is thus denser.

The advantage of using N-best lists or word lattices as interface is that natural language processing and speech recognition can be decoupled and developed independently of each other [HJM+94]. The information flow is strictly feed-forward which simplifies the design.

Analogous to N-best processing there are two approaches to process lattices: lattice filtering [BCH+89, CR89, HJJ+99, HJ03] which selects the acoustically highest scored lattice path which is accepted by the linguistic module, and lattice rescoring [vNBKN99, Kie00, CCP04] which combines the acoustic score with a syntactic score and extracts the best path with respect to the combined score.

A robust approach being able to combine partial analyses of a rule-based grammar was implemented as part of the Verbmobil project in [Kie00]. The system uses an HPSG parser to find passive edges over

a continuous sequence of words. The system then assigns each passive edge a score, at which edges with utterance status (e.g. NPs, PPs) receive a better score than lexical edges. A shortest-path algorithm finds the optimal sequence of partial analyses covering the whole input. The acoustic score and the probabilistic language model score are not considered in determining the best analysis. A very similar approach was taken in OVIS [NBKvN97, vNBKN99], which, however, also takes the acoustic score into account.

Most approaches first create a word lattice [ON97] and then apply the parser in a post-processing step. A heuristic time-synchronous generation of word-lattices based on local decisions without backtracking is described in [PR96]. The advantage is that the higher level processing can take place at the same time as the signal is decoded which allows to parallelize computation. The size of the lattice is controlled by a beam pruning parameter. More effective pruning methods such as forward-backward pruning [SO99], however, require the full lattice to be available and can thus not be applied.

3.2 Related Projects

In Chapter 5 we will use an architecture based on lattice parsing to give evidence that rule-based linguistic knowledge can significantly improve the recognition accuracy. This architecture was also used in two large projects: OVIS and Verbmobil.

3.2.1 OVIS

OVIS is a demonstrator of a Dutch spoken public transport information system over telephone lines [SRVDH+97]. The aim was to automate the processing of queries which are limited to travels between two train stations. The vocabulary consists of about 3200 words which are mostly names of stations and cities.

In its first version, an NLP module looked for sequences of semantic concepts rather than trying to find a full parse. Later on a grammar inspired by HPSG whose rules were compiled into a definite-clause grammar (DCG) was used for analyzing the speech input. To achieve robustness, the system searched not only for sentences, but also grammatical

categories such as noun phrases or prepositional phrases, which correspond for example to temporal expressions or locative phrases from that domain.

Given some optimization criteria the system finds an optimal path through a word hypothesis graph, for example, minimizing the number of grammatically correct phrases and skipped words. In [NBKvN97] the optimization criteria first minimizes the number of skips, then the number of phrases, and then the acoustic scores. In [vNBKN99] the optimization criterion was replaced by a function which is a weighted sum of log acoustic likelihood, log language model probability and number of edges.

3.2.2 Verbmobil

The Verbmobil system recognizes spontaneous continuous speech, translates the utterance to another language and converts the translated text to speech [Wah00].

In the first phase the domain was appointment negotiation. The research prototype translated German to English (vocabulary of 2500 words) and Japanese to English (vocabulary of 400 words). The objective of the second phase was the bidirectional translation of spontaneously spoken dialogues for German/English and German/Japanese with a vocabulary size of about 10,000 words. The domain was appointment scheduling, travel planning and making hotel reservations.

Vermobil makes use of deep syntactic analysis with HPSG (word lattice parsing) and is able to select partial analyses by a shortest-paths algorithm. From the algorithmic point of view the approach taken in Chapter 5 is very similar to the approach described in [KKK+99]. However, the limited domain of Verbmobil reduces lexical ambiguity and excludes certain types of constructions [Usz02].

3.3 Further Investigations

The thesis subsequently analyses two architectures. In Chapter 4 we propose our own architecture which is based on word spotting and island chart parsing. We have chosen this approach because of its flexibility.

The second approach in Chapter 5 uses a parser to rescore a word lattice. The motivation for the choice of the second approach lies in its ability to clearly attribute any improvement of the recognition accuracy to the linguistic component. This property makes it a good choice to give an answer to question two, how much linguistical knowledge can improve recognition accuracy.

Chapter 4

Word Spotting Approach

4.1 Introduction

Since precise linguistic knowledge is rule-based, the recognition system gets inhomogeneous: it includes a statistical and a rule-based part that have to co-operate in some way.

A very simple type of co-operation is sequencing the two subsystems, as can be seen in many so-called speech understanding systems: The statistical subsystem provides some hypotheses, e.g. an N-best word lattice which is subsequently processed by the knowledge-based subsystem. In particular, there is no feedback from the knowledge-based subsystem to the statistical one.

This sequencing can be considered to be inappropriate because it is impossible to determine the number of best hypotheses that have to be provided by the statistical subsystem in advance. It can always happen that a necessary hypothesis is missing and therefore the knowledge-based subsystem cannot find the correct sentence. Increasing the number of hypotheses does not eliminate but only decrease this problem at the costs of a new one, namely the workload of the parser of the knowledge-based subsystem (combinatorial explosion).

A further issue arises from the fact that not all parts of an utterance are equally intelligible. Emphasized syllables are more precisely articulated, whereas others might be pronounced rather carelessly. Addition-

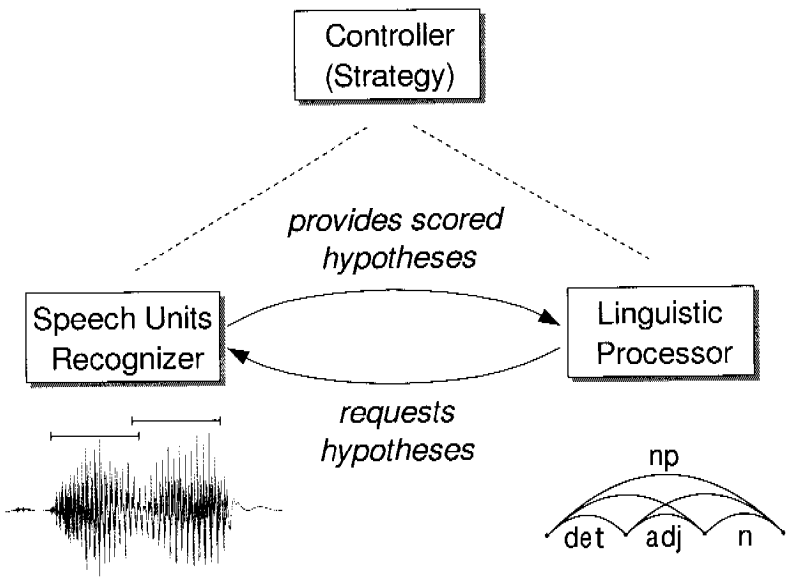


Figure 4.1: *Fundamental architecture of a speech recognition system that combines statistical speech units recognition and linguistic knowledge processing (bidirectional interaction)*

ally, there are coarticulation, pronunciation variations and noise. This motivates to start the recognition at those points where we consider the hypotheses to be reliable using some sort of confidence measure.

4.2 System Architecture

Based on the above considerations, we propose a speech recognition system architecture which includes a statistical speech units recognizer and a rule-based linguistic processor [BP03]. It has to be emphasized that these two subsystems work tightly together. The speech unit recognizer provides an initial set of hypotheses to the linguistic processor, that in turn can request additional hypotheses. Such a request can be very specific, e.g. with respect to location or syntactic information. This requires an incremental processing and a bidirectional interaction between these two components, as illustrated in Fig. 4.1.

On top of these two subsystems, an operational system needs some control module. Consequently, the proposed system is composed of the following three sub-systems:

- A statistical speech units recognizer which *i)* provides initial hypotheses, *ii)* is ready to produce additional hypotheses on demand at designated locations and *iii)* is able to score hypotheses proposed by the linguistic processor.
- A rule-based linguistic processor that concatenates small hypotheses to larger ones according to a grammar without any restriction of the parsing direction.
- A control module which includes among other things a strategy component that decides which of the currently possible actions the parser has to execute in order to optimally make progress towards the final solution.

This architecture specifies a broad class of recognizers that can be instantiated in different ways. For our prototype we have chosen some sort of word spotter as speech units recognizer and an island chart parser as linguistic processor.

This prototype has been implemented primarily as a proof of concept. Therefore, the main focus was on exploring principles rather than achieving efficiency.

4.3 Speech Units Recognizer

As outlined in Sec. 4.2, the speech units recognizer has to provide either hypotheses or has to compute the score of a suggested hypothesis. In both cases some sort of word spotting is used. The speech units recognizer is based on phonemes and is able to provide hypotheses for phoneme sequences of arbitrary length, ranging from single phones to morphemes, words, phrases and sentences.

4.3.1 Benefits of Word Spotting

A word spotter is able to find the best match of a keyword in a signal. For a given keyword, it computes both the location in the signal and a

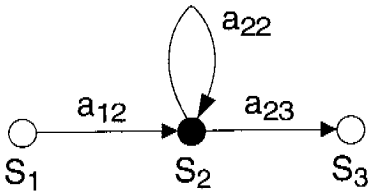


Figure 4.2: *Single state HMM with non-emitting entry and exit states. The stay state transition probability is denoted as a_{22}, the state change probability is $a_{23} = 1 - a_{22}$, and $a_{12} = 1$.*

corresponding acoustic score. Note that the spotting algorithm is not restricted to entire words, but can also be used for sub-word units (e.g. morphemes), multi-word expressions or even whole sentences. This has been exploited threefold:

1. By spotting every word of the recognizer lexicon and sending the best scored words as initial hypotheses to the linguistic processor.

2. The linguistic processor can request additional hypotheses at a specific location in the signal.

3. When the linguistic processor concatenates two hypotheses into a single one, the joined hypothesis can be scored by spotting it in the vicinity of the original ones. Scoring the joint hypothesis is important as its constituting hypotheses often overlap or have a gap in between, as will be explained in more detail in Sec. 4.4.2.

4.3.2 Word Spotter

The word spotter is based on the Viterbi decoding algorithm and operates on phoneme models. Phonemes are represented by 40 context-independent monophone HMM/ANN models with a single state, as depicted in Fig. 4.2.

As filler model a variant of the online garbage model in [BDB94] is used. It computes the average of the N best local phoneme scores,

whereby the top-value itself is not considered. Thus, the garbage model is never the best one, but is always one of the top candidates.

Duration Model

The geometric duration model of standard HMMs based on static state transition probabilities has been replaced by a more adequate parametric model following a Gamma distribution. Explicit modeling of the state distribution was shown to improve recognition in [Rab89] and Gamma distributions are reported to fit empirical distributions sufficiently well in [Bur95].

Our duration model replaces the static state transition probabilities by dynamic ones which depend on the duration d spent so far in the current state. The state change transition probability $a_{23}(d)$ in function of this duration d is computed such that the distribution of the phone duration follows a Gamma distribution [MP02]. The free parameters of the Gamma distributions have been estimated from the mean μ and variance σ^2 of the duration of HMM states from a standard Viterbi segmentation of the training set.[1]

In order to apply the duration model in the word spotter, the Viterbi recursion has been changed slightly by replacing the constant a_{ij} by the function $a_{ij}(d)$:

$$\delta_t(j) = \left[\max_i \delta_{t-1}(i)\, a_{ij}(d)\right] \cdot b_j(\mathbf{x}_t) \tag{4.1}$$

where $\delta_t(j)$ is the score for observing the feature vectors $\mathbf{x}_1$ to $\mathbf{x}_t$ and being in state S_j at time t; and $b_j(\mathbf{x}_t)$ is the probability to observe the feature vector $\mathbf{x}_t$ in state S_j.

Double Normalized Scores

The speech units hypotheses are scored with a double normalization technique which takes into account the number of frames in each phone and the number of phones in each word (phone-based normalized posterior confidence measure, [BB98]). According to our experience this measure significantly improves the accuracy of the word spotter score.

[1] Shape parameter $\alpha = \mu^2/\sigma^2$ and inverse scale parameter $\beta = \mu/\sigma^2$.

4.4 Linguistic Processor

The linguistic processor is realized as an active island chart parser. This section outlines our motivation for this choice and explains further details.

4.4.1 Island Chart Parsing

The chart parsing framework [Kap73, Kay82] is very powerful for parsing natural language. Hypotheses are represented by *edges* which are stored in a data structure called *chart*. The chart represents all syntactic structures that have been found or tried so far. Thus, it shows also the parsing state and it prevents any work from being repeated. Active chart parsing uses an *agenda* to keep track of the grammar rules still to be applied. Depending on the implementation of the agenda different rule invocation strategies (top-down, bottom-up or mixed) and different search strategies (depth-first, breadth-first, best-first etc.) can be adopted. This is attractive as it provides a flexible framework that can be controlled by a strategy component.

An extension of standard chart parsing is island parsing, which enables the recognizer to start at "trustily" recognized constituents and proceed bidirectionally [SDR87, SFI88]. There are two extremes in choosing the number of islands, either only a single one is selected or every word hypothesis is regarded as an island. Our implementation does the latter. Only hypotheses which belong to an island can be expanded. Since we do not know which words belong to the correct solution, and since we want to be able to find grammatically correct phrases anywhere in the utterance, our implementation regards every word hypothesis as an island.

The properties of island chart parsing conform to the requirements discussed in Sec. 4.1 and Sec. 4.2.

4.4.2 Parsing Text vs. Speech

Chart parsing is commonly used for parsing written text. In this case there is always a unique unambiguous boundary between two consecutive words. These boundaries are the vertices to which the edges in the chart are anchored. Only adjoined edges can be combined.

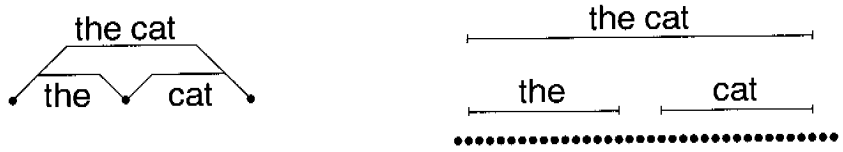


Figure 4.3: *When parsing text, the vertices of the chart are unique word boundaries (left) and it is thus always clear which words are neighbours and consequently can be concatenated to a larger hypothesis. This contrasts to acoustic hypotheses from a word spotter: generally such hypotheses do not fit; there is often an overlap or a gap (right). Therefore, when parsing speech, the vertices are acoustic frames and words can be connected when they meet some adjacency criterion.*

In contrast to that, the word boundaries of hypotheses created by a word spotter are not constrained to be adjoined (typically they overlap or have gaps in between), so chart parsing is not directly applicable. This difference is illustrated in Fig. 4.3.

The simplest solution is to redefine the term of adjacency of two edges. Instead of saying two edges e_1 and e_2 are adjacent if $e_1.end = e_2.start$ we define them to be still adjacent as long as the mismatch of the concerned boundaries is less than τ:

$$\text{adjacent}(e_1, e_2) = \begin{cases} true & \text{if } |e_1.end - e_2.start| < \tau \\ false & \text{else} \end{cases}$$

4.4.3 Morphological Knowledge

Most German words are composed of a stem, an ending and optional prefixes. From one stem typically dozens of correct word forms can be derived and thus the number of full word forms is much higher than the number of stems. Using morphological knowledge has a number of advantages. The lexicon can be more compactly represented[2], the use

[2]The system was extended after the experiments and was able to derive 30'000 word forms from 5'000 morpheme stems and 750 endings.

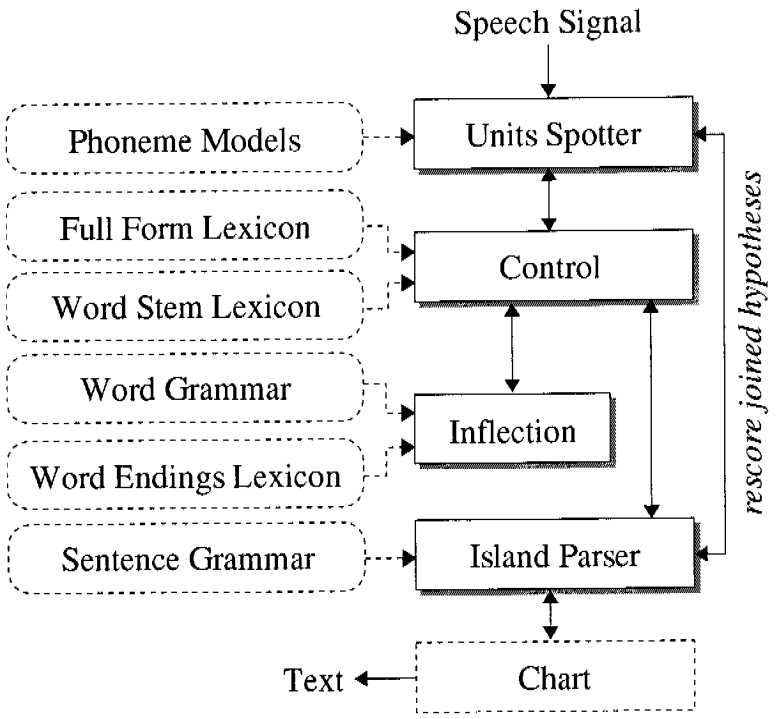


Figure 4.4: *Block-diagram of the recognizer.*

of rules improves consistency and in our architecture it can be used to improve performance.

In order to reduce the workload of the word spotter, we spot the stems instead of the full word forms. Only if the control module decides that a stem hypothesis has to be considered for further processing, this stem is expanded to full word forms which are again supplied to the spotter for scoring.

Some words are not subject to this optimization, namely uninflectable words and irregular forms. They are treated as full word forms and are spotted directly.

Consequently, the recognizer shown in Fig. 4.4 has got three lexica that contain *full word forms*, *stems* and *endings*, resp. The full word forms lexicon mainly contains grammatical words, but also some irregular forms. The stem lexicon also informs for each stem which prefixes it can take. Additionally, there is a word grammar that tells the inflec-

tion module which stems can be combined with which endings to full word forms. The implementation represents the morphological knowledge in terms of *definite-clause-grammar (DCG)* rules and *finite state transducers (FSTs)* and is largely based on [Tra95].

4.5 Control Module

This section demonstrates the recognition process by showing how the statistical subsystem and the linguistic processor operate under the coordination of the control module.

4.5.1 The Recognition Process

The recognizer depicted in Fig. 4.4 processes an input speech signal as follows: First and independently of the subsequent processing, the utterance boundaries are detected by computing the best alignment of the model sequence *silence - filler model - silence* using dynamic programming.

Afterwards, all entries of the full form lexicon and the stem lexicon are spotted and the resulting hypotheses are stored in a list. Then the following steps are repeated:

1. Select the best scored hypothesis and remove it from the list.

2. Since a word or a stem can appear more than once in one and the same utterance, it has to be spotted again in the areas it has not already been found. The resulting hypotheses are added to the list. If the hypothesis selected in step 1 is a word, skip step 3.

3. For the *stem* selected in step 1, all possible full word forms are generated and spotted in the signal (near the found stem). The resulting hypotheses are added to the list. Go back to step 1.

4. The *full word form* is passed to the island chart parser that adds an edge representing the word hypothesis to the agenda. Then a full parse is executed, i.e., the rule invocation is repeated until the agenda is empty. In other words, all syntactic structures derivable from the word hypotheses known so far (i.e. that have been passed to the parser) are computed. Note that whenever

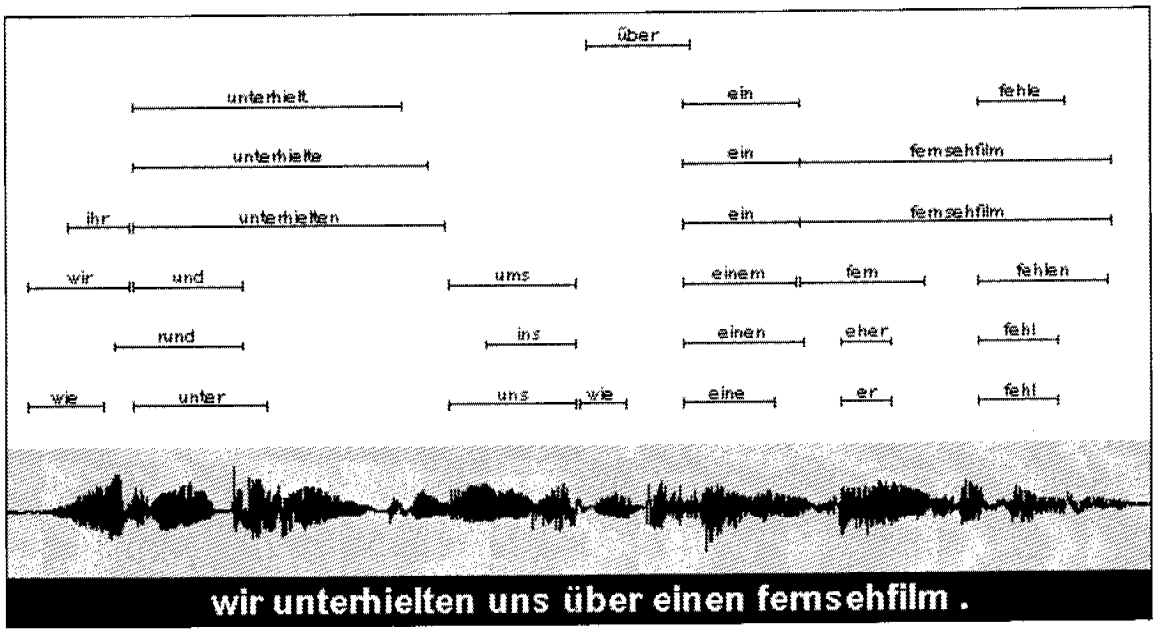


Figure 4.5: *Screenshot of the continuous speech recognizer based on word spotting. The upper part shows the word hypotheses provided by the word spotter, which are concatenated to larger ones according to a grammar. The recognized utterance is shown at the bottom ("we were talking about a TV movie.")*

two edges are combined to a single one, the combined edge has to be rescored by spotting the corresponding word sequence (cf. Sec. 4.4.2).

5. If no stop criterion is met, continue with step 1.

6. The best scored sentence hypothesis is printed out. Sentence hypotheses that span the utterance boundaries are always preferred from shorter ones and are compared using the double normalization technique from Sec. 4.3.2. If no sentence hypothesis spans the utterance boundaries, the sum of the log phoneme probabilities is used for comparison instead of the normalized score.

The stop criterion currently used is a timeout proportional to the utterance length.

Pruning

Unpromising edges are conservatively pruned. Before an edge is added to the agenda its score is compared to all edges spanning approximately the same part of the utterance. If the score of the edge in question is not among the N best, it is pruned. N depends on the number of words spanned by the edge (cf. Table 4.1 for exact values). The values were chosen ad hoc.

k	1	2	3	4	5	6	>6
N	100	10	5	4	3	3	2

Table 4.1: *An edge spanning k words is added to the agenda if its score is within the N best competing edge scores found in the chart, otherwise it is pruned.*

4.6 Experiments

The aim of the experiment was in a first attempt to explore the proposed concept and test the prototype. Since the entire system was designed and implemented from scratch, the task had to meet some constraints. The utterances should be of a manageable complexity, which restricts the size of the vocabulary and the range of grammatical phenomena in the test sentences. Another simplification was achieved by allowing the system to be speaker dependent.

In order to train acoustic models and test the system, a corpus was required which has enough data of a single speaker to train a speaker dependent model and contains suitable sentences for testing. There was no German corpus available which satisfied these constraints. Therefore the recording of the test and training data was done by ourselves. To prevent an oversimplification of the task, the test sentences were not defined by ourselves, but taken from a dictation book for pupils (cf. Appendix A). The sentences used to train the acoustic models are taken from newspaper text. Thus, the training and test data can be assumed to be disjoint.

The utterances of both the training and test set were spoken by a single male speaker in an office environment with low background noise using a head-set microphone sampled at 16 kHz. The feature vector extracted at each frame consisted of 14 standard MFCCs plus the log-energy [Eur00]. These vectors were extracted from 25ms windows at a frame rate of 100 Hz.

4.6.1 Training

40 context-independent monophone HMM/ANN models with a single state were trained. The neural network is a three layer perceptron with 120 and 80 neurons in the first and second hidden layer, respectively. The input is composed by the features of the current frame plus 8 frames context. Thus, the input layer has $15 \cdot 9 = 135$ neurons. The input of each input neuron is transformed to have zero mean and standard deviation on the training data. There are 40 neurons in the output layer, one for each phoneme. The network was trained for 10 epochs on 887 newspaper sentences containing 51937 phonemes, corresponding to 101 minutes speech. The samples were chosen randomly such that the phonemes were trained uniformly. The weights were updated after each sample (stochastic learning) using standard back-propagation.

4.6.2 Testing

The test data consists of 72 sentences taken from a dictation book for pupils.[3] The sentences used for training the HMM/ANN models and the test sentences are disjoint. The utterances contain 3, 6.8 and 11 words in minimum, mean and maximum respectively. The total number of words in the test sentences is 489. Out of the 4067 different word forms that can be recognized, 232 word forms appear in the test sentences. The full form lexicon contains 1418 entries and the morpheme stem lexicon consists of 492 stems. The pronunciation of a lexicon entry follows the citation form given in [Dud90], pronunciation variations

[3]Of the set of test sentences (cf. Appendix A) the first 72 parseable sentences were used. These are: 0, 1, 2, 3, 4, 5, 7, 8, 11, 12, 13, 14, 15, 16, 17, 19, 20, 21, 22, 23, 24, 26, 29, 30, 32, 33, 35, 36, 38, 39, 40, 41, 42, 46, 47, 48, 52, 55, 58, 59, 61, 68, 70, 75, 79, 83, 84, 86, 90, 92, 93, 96, 99, 100, 102, 103, 104, 106, 107, 110, 111, 113, 115, 116, 117, 118, 119, 120, 133, 134, 135, 136

were not taken into account. There are neither out-of-vocabulary words nor out-of-grammar sentences.

The 130 *Definite Clause Grammar* (DCG, [PW80]) rules of the sentence grammar cover the following aspects of the German language. Verb tenses: present tense (Präsens), simple past tense (Präteritum), present perfect tense (Perfekt), past perfect tense (Plusquamperfekt), moods: indicative and subjunctive, main sentences with free word order (affirmative and negative statements) and questions. The morphological knowledge was expressed in terms of 27 word grammar rules.

The experiments have been conducted for different neural networks (size of hidden layers, number of epochs trained etc.), garbage model parameter values N, and gap tolerances. Only the best result is reported here. These parameters were thus optimized to the test data. The parameters were chosen as follows. The online garbage model was computed by the top 15 phone probabilities and two parsing edges were considered to be adjacent if the gap was smaller than $\tau = 10$ frames (100ms). The recognition process was stopped when it ran 10 times as long as the duration of the signal.

The best scored hypothesis is compared against the reference in terms of substitutions, insertions and deletions.

4.6.3 Results

A word error rate (WER) of 6.5% was achieved. The recognition results on the word and sentence levels are given in Table 4.2. The detailed results are: 461 correct words (C), 4 insertions (I), 7 deletions (D) and 21 substitutions (S) on a total number of 489 words (N).

4.7 Compound Words

Several advantages of using morphological knowledge have already been pointed out in Sec. 4.4.3. Another advantage is the efficient treatment of compound words. This section describes an extension to the system which allows to handle compound words and prefixing of verbs. This extension was not part of the experiment described in the last section.

words correct	94.3%
word accuracy	93.5%
sentences correct	66.7%
word error rate (WER)	6.54%

Table 4.2: *Recognition results in terms of words correct (C/N), word accuracy $(C-I)/N$, word error rate $(I+D+S)/N$ and number of sentences without any error on a speaker dependent recognition task with a 4k words dictionary.*

4.7.1 The Importance of Compound Words

Compound words are an important aspect of German's rich morphology: it means that new words can be created by concatenating words and/or morphemes. This kind of word formation can produce correct words which are easily understandable, but cannot be found in any dictionary. A famous example is the word *Donaudampfschifffahrtsgesellschaftskapitän* (Danube steamboat navigation company captain), which is a single word formed by six nouns (Donau, Dampf, Schiff, Fahrt, Gesellschaft and Kapitän) and two filler morphemes (s).

Compounds are widely used both in written and spoken language. The vocabulary of the German language has therefore to be considered of virtually infinite size. Although this problem can be mitigated by using a large recognizer vocabulary with several hundred thousand words [MAD03], a large vocabulary does not really solve the problem.

A speech recognizer for German definitely needs mechanisms to cope with this phenomenon in order to reduce the problem of out-of-vocabulary errors. Since the proposed recognizer is morpheme-based anyway, it can be extended easily to handle compounds as well by adding word compound rules. These rules define how morphemes can be combined to words and allow to derive the syntactic properties of the new word. In German, a noun compound word inherits the syntactic properties of the last noun. The semantic properties however cannot be derived.

Rule	Example
noun → noun + noun	Baumstamm
noun → noun + 's' + noun	Staatstreffen
noun → verb stem + noun	Gehstock
noun → verb stem + 'e' + noun	Haltestelle
noun → adjective stem + noun	Schnellstrasse

Table 4.3: *Most often used compound rules for German noun compounds.*

4.7.2 Regularities

Although compounds can be created with virtually no restriction, there are a few rules that cover a fair amount of compounds. The rules described in Table 4.3 cover the most frequent German noun compounds (approximately 90%). There are other possibilities as well, e.g. noun → particle + noun, but they occur much less frequent and are not considered for the sake of performance. As these rules are recursively defined, compound words of arbitrary length can be handled. In addition to the rules above, adjective compounds and the prefixing of verbs ('be', 'ver', 'um', 'weiter', 'wieder', etc.) were also implemented. The number of rules of the compound grammar is 9.

4.7.3 Search Strategy

The number of hypotheses which a speech recognizer has to process increases tremendously when compound rules are added. Words then not only interact on the level of the sentence grammar, but also on the level of the compound grammar. Each additional word hypothesis originating from compounding results in an additional work load for the sentence parser. Restrictive pruning must be applied in order to deal with the exceedingly productive compound rules.

The recognizer described in Sec. 4.5.1 spots word stems, predicts possible word endings and creates the corresponding full form words,

which are spotted again. The top-down prediction of endings like 'e', 'es' and 't' is appropriate because endings are very short. Short keywords are difficult to spot since they *i)* match in many places because they are not so specific, and *ii)* can be easily missed if there is a slight mismatch. For the same reason, the two compound filler morphemes 's' and 'e' (cf. Table 4.3) are predicted and tested for their presence in a top-down manner. For example, when two nouns are close to each other, a compound word is hypothesized bottom-up and the existence of the morpheme 's' is verified top-down, which results in two compound nouns hypotheses, one with filler and one without. The existence of a compound filler morpheme cannot be predicted by rules since it is arbitrary to some extent and must therefore be detected in the signal.

A similar bottom-up / top-down strategy was used to process verb prefixes. Prefixes which are too short to be detected reliably in the speech signal bottom-up are predicted top-down (e.g. 'ge', 'be', 'um', 'ab'). All the others are processed bottom-up (e.g. 'fort', 'heraus', 'unter').

4.7.4 Results

The extended system is able to handle noun and verb compounds in roughly the same amount of processing time as before, due to the optimization described above. Allowing compounds is equivalent to a very large vocabulary. As a result, the word error rate increased to 27%. To improve, better acoustic models and a statistical language model or a statistical compound model would be required. Due to the limitations discussed in the next section, further work on the architecture was discontinued.

4.8 Discussion

The aim of the work described in this chapter was to provide an architecture which allows to apply rule-based linguistic knowledge within the statistical framework of a speech recognizer. As outlined in the previous chapter, a variety of approaches can be taken. In the following, the advantages and drawbacks of the proposed architecture are discussed.

4.8.1 Advantages

Flexibility is the architecture's main advantage. The use of a word spotter allows to use any kind of search strategy (depth-first, breadth-first, best-first, left-to-right etc.) and any kind of rule invocation strategy (top-down, bottom-up, mixed).

The flexibility was not exhaustively used in the experiments. When two hypotheses are combined, the joint hypothesis is rescored by spotting it in the vicinity of the original ones. The rescoring could be done more accurate, e.g. by taking cross-word pronunciation variations at the joint into account.

Since the system is not restricted to process the utterance from left to right it can skip unintelligible parts. Skipping is not done intentionally but is a side-effect of word spotting and island parsing. Depending on the application, it may be preferable to mark the parts of the signal where recognition was not possible rather than to count on a transcription which is very unreliable.

4.8.2 Problems

The control module is a complex part. This became clear when the system was extended to handle compound words. The problem is to decide when to take which action. For example, one has to trade exploration versus exploitation, i.e. when does is make sense to provide additional hypotheses by spotting additional stems and thus broaden the search, when should the compound grammar be used, and when and how long should the sentence parser run? In the current system all derivable syntactic structures are computed. For longer sentences or under more difficult acoustic conditions this approach could render to be computationally prohibitive due to combinatorial explosion.

The detection of short stems or short words (like grammar words) is difficult for a word spotter. If these words are carelessly pronounced, they receive a low score. It may happen, that they are either pruned away or the timeout occurs before they are processed. This problem was not pronounced so much in the experiments described here since the word spotter was speaker dependent, however, it may aggravate for a speaker independent system.

The effect of allowing compounds is comparable to having a very large dictionary. With increasing dictionary size the confusability increases, which leads to lower recognition accuracy. Therefore, adding the possibility of recognizing almost arbitrary compound words comes at the cost of a lower recognition rate. In order to compensate the increased confusability a statistical compound model would have to be used.

4.8.3 Further Work

Apart from the acoustic score, the described system treats each word sequence to be equally likely. The performance could be improved by adding a stochastic language model, which would allow the control module to use prior knowledge to guide the system. The acoustic score of each hypothesis (consisting of at least a full form word) could be multiplied by a language model score. This would allow to reorder the agenda and reflect the fact that some utterances occur more frequently than others. A stochastic compound language model would be required as well when the compound grammar is enabled.

The search space is restricted by the grammar to the language it defines. Thus, the current system can only recognize intra-grammatical sentences. This limitation could be overcome by taking advantage of the island parser. The fragments (islands) found by the parser could be combined if no solution is found which spans the whole signal. For example, a minimum number of adjacent fragments could be chosen by means of dynamic programming. In order to be robust, several decompositions of the signal into fragments would have to be considered. Each would have to be scored by the word spotter and the highest scored solution would be chosen.

4.9 Conclusions

A novel speech recognizer architecture which integrates both statistical acoustic knowledge and rule-based linguistic knowledge was proposed. It applies morphological knowledge for efficient processing and can handle compound words of arbitrary length in a straightforward way. As a proof of concept a prototype was implemented and a recognition ex-

periment was carried out. This prototype achieves a word recognition accuracy of 93.5% on a speaker dependent continuous speech recognition task consisting of grammatical sentences with a vocabulary of 4k words. To apply the system on a more complex task, a statistical language model and an extension to recognize non-grammatical utterances as well would have to be added. From the experiences made it can be concluded that rule-based knowledge should not be seen as a replacement for an N-gram, but rather as complementary information.

Seite Leer /
Blank leaf

Chapter 5

Lattice Parsing Approach

The architecture introduced in the previous chapter was mainly an answer to the first question of this thesis, how non-probabilistic knowledge can be integrated into a stochastic speech recognizer. The architecture described in this chapter is less versatile, however, it is better suited to compare a linguistically enhanced system with a standard system.

The architecture is frequently used in natural language understanding systems: a word lattice serves as an interface between an acoustic recognizer and a natural language processing module. In our approach a score derived from the syntactic structures found by the parser is used to rescore the word lattice such that grammatical phrases are slightly favoured.

5.1 Architecture

The aim is to provide evidence that rule-based knowledge can improve LVCSR accuracy. To do so, a speech recognizer must be extended by a rule-based component in a way such that any improvement of recognition accuracy can be clearly attributed to that component and therefore to the linguistic knowledge. The architecture shown in Fig. 5.1 fulfills this requirement: a speech recognizer creates word lattices and at the same time provides a baseline word error rate. The word lattices are subsequently rescored by a natural language processing module.

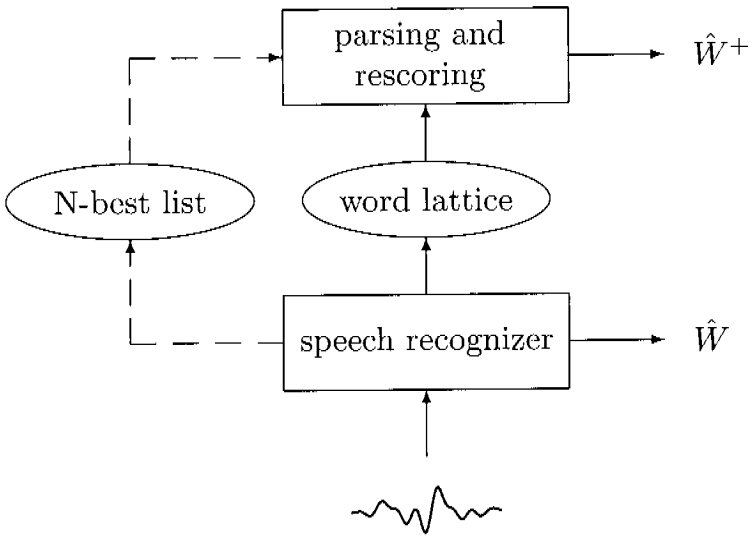


Figure 5.1: *A speech recognizer is used to measure the baseline recognition accuracy for a given task. We aim at outperforming this baseline system by rescoring the word lattice created by the recognizer by means of linguistic knowledge. The N-best list is used to drive the parser to work on the most promising hypotheses first.*

By comparing the word error rate of the enhanced system with the baseline word error rate we can directly quantify the benefit of the linguistic component.

Initially, a word lattice is produced by the baseline speech recognizer. Due to the uncertainty of word boundaries in continuous speech, the same word sequence may be represented by several lattice paths with different acoustic scores. The parser only considers the word sequence of a path but not its score. Consequently, there is no use in parsing several paths that differ in score only. By ignoring acoustic scores and timing information we create a word graph which represents all word sequences of the lattice in compact form and thus can be processed more efficiently by the parser.

Ideally, the parser processes each path in the word graph, producing all phrases which can be derived from the corresponding word sequences. As the number of paths in a word graph may be huge, one

has to focus on the most promising hypotheses. For this reason only the paths in the baseline system's N-best list are parsed, starting with the best hypothesis. If the parsing time exceeds some predefined limit, the parsing procedure is terminated. Parsing can potentially lead to a combinatorial explosion of hypotheses resulting in large processing times. The parsing time limit guarantees a worst-case running time of the experiments.

The phrases derived by the parser are used to rescore the recognizer lattices. However, the final solution is not restricted to the N-best paths but can rather be any path in the lattice. The rescoring step will be described in the next section.

5.2 Scoring Syntactic Structures

A speech recognizer's aim is to find the word sequence which was most likely uttered given an acoustic observation $\mathbf{X}$. The *maximum a posteriori* (MAP) criterion chooses the word sequence W such that the product of the acoustic likelihood $P(\mathbf{X}|W)$ and the language model probability $P(W)$ is maximized (cf. Eq. (2.3)).

In practical applications the acoustic likelihood and the language model probability have to be balanced to optimize performance. Also, adding a word insertion penalty has proven to be advisable to get optimal results:

$$\hat{W} = \arg\max_{W \in V^*} P(\mathbf{X}|W) \cdot P(W)^{\lambda} \cdot ip^{|W|} \quad . \tag{5.1}$$

λ denotes the language model weight, the norm $|\cdot|$ measures the length of a sequence and ip is the word insertion penalty.

5.2.1 Extending the MAP Criterion

We extend the MAP criterion with an additional parsing score $f(W)$ which allows us to favour grammatical utterances:

$$\hat{W}^{+} = \arg\max_{W \in V^*} P(\mathbf{X}|W) \cdot P(W)^{\lambda} \cdot ip^{|W|} \cdot f(W) \tag{5.2}$$

As we use a non-probabilistic rule-based grammar the parser does not provide a score but only syntactic structures. In the remainder of this section we explain how the parsing score $f(W)$ can be computed from syntactic structures.

Let W be a word sequence in the lattice spanning the whole utterance. W can be decomposed into a sequence $U = \langle u_1, u_2, \ldots, u_n \rangle$ of so-called parsing units u_i. A parsing unit u_i represents a word sequence $w(u_i)$ which the parser identified as being grammatically correct. The decomposition is such that the concatenation $w(u_1) \circ w(u_2) \circ \cdots \circ w(u_n) = W$. Note that for a given W there may exist several different decompositions.

Three types of parsing units are distinguished. The smallest unit represents a single word. Units which are larger than one word but do not span a whole utterance are called fragment units. A unit spanning the whole utterance is called an utterance unit. We first define the parsing score $s(\cdot)$ for a single parsing unit to depend on its unit type:

$$s(u) = \begin{cases} c_\alpha & \text{if } w(u) = W, \\ c_\beta & \text{if } 1 < |w(u)| < |W|, \\ c_\gamma & \text{else} \end{cases} \tag{5.3}$$

where c_α, c_β, and c_γ denote the scores for utterance units, fragment units and single word units, respectively. The score of a decomposition of W is

$$g\left(\langle u_1, \ldots, u_n \rangle\right) = \prod_{i=1}^{n} s(u_i) \ . \tag{5.4}$$

The score of word sequence W is the maximal score of all its valid decompositions:

$$f(W) = \max_U g(U) \ . \tag{5.5}$$

Note that $f(W)$ is always defined, even if the utterance is not fully parsable, because W can always be decomposed into single word units. Therefore a fall-back mechanism for unparsable sentences is superfluous. Table 5.1 illustrates how parsing scores are computed.

sentence	parsing score
$\underbrace{\text{(Anna and Bob go to school)}}_{c_\alpha}$	$f(W) = c_\alpha$
$\underbrace{\underbrace{\text{(Anna and Bob)}}_{c_\beta}}_{c_\alpha}\ \underbrace{\text{(so)}}_{c_\gamma}\ \underbrace{\underbrace{\text{(to school)}}_{c_\beta}}_{c_\alpha}$	$f(W) = c_\beta^2 \cdot c_\gamma$
$\underbrace{\text{(Anna)}}_{c_\gamma}\ \underbrace{\text{(and)}}_{c_\gamma}\ \underbrace{\text{(bobbed)}}_{c_\gamma}\ \underbrace{\text{(go)}}_{c_\gamma}\ \underbrace{\text{(two)}}_{c_\gamma}\ \underbrace{\text{(school)}}_{c_\gamma}$	$f(W) = c_\gamma^6$

Table 5.1: *Three examples illustrating how the parsing score is computed.*

5.2.2 Parameter Optimization

The parameters λ, ip, c_α, c_β, and c_γ are optimized on development data to minimize the empirical word error rate.

Experience suggests that the error surface is not smooth and has a large number of local minima [KOR92]. Because the word error rate is not a continuous objective function gradient descent methods can not be applied. One possibility would be to optimize the weights by Powell's Method [PTVF02], as described in [OKA+91].

In this work, however, the downhill simplex method known as amoeba search (a multidimensional unconstrained nonlinear minimization algorithm, [NM65]) is applied, since it was already successfully used for different weighting problems in speech recognition [VTB00, Ver00, GVN+01].

5.3 Algorithms and Data Structures

The preceding section contains a high-level overview of the lattice parsing approach. This section explains in detail how the word sequence $\hat{W}^+$ can be extracted from a given lattice according to the extended MAP criterion of Eq. (5.2).

The main data structures on which the algorithms operate are word lattices and word graphs. Since in literature the terms word lattice and word graph are used differently, the section starts with a definition of these terms.

5.3.1 Definition of Word Lattice and Word Graph

All descriptions of algorithms are based on the definitions given here.

Word lattice. Words are represented by weighted edges, where the edge weight corresponds to the acoustic score. Nodes define the start and end of a word and stand for a point of time in the signal. An example of such a lattice is given in Fig. 5.2.

Word graph. Words are represented by nodes and the node weight corresponds to the acoustic score. The edges define the word sequences.

Formal Definition of Word Lattices

A *word lattice* is a *directed acyclic graph* (DAG) consisting of a tuple $G = (\mathbb{N}, \mathbb{E})$ of a set of nodes $\mathbb{N} = \{n_1, n_2, \ldots, n_{|\mathbb{N}|}\}$ and a set of directed edges $\mathbb{E} = \{e_1, e_2, \ldots, e_{|\mathbb{E}|}\}$.

Each node $n_i \in \mathbb{N}$ is associated with a floating point value $time(n_i)$ which denotes a point of time in the signal in seconds, and an integer identifier $id(n_i)$.

Each directed edge $e_i = (n_{start}, n_{end}) \in \mathbb{E}$ is defined by a start node $n_{start} = start(e_i)$, an end node $n_{end} = end(e_i)$, a string $word(e_i)$, an acoustic score $score_{ac}(e_i)$, a language model score $score_{lm}(e_i)$ and an optional parsing score $score_{parse}(e_i)$. Each node n_i has a set of predecessor edges $pred\_edges(n_i) = \{e_j \,|\, end(e_j) = n_i\}$ and a set of successor edges $succ\_edges(n_i) = \{e_j \,|\, start(e_j) = n_i\}$, as well as a set of predecessor nodes $pred\_nodes(n_i) = \{n_j \,|\, (n_j, n_i) \in \mathbb{E}\}$ and a set of successor nodes $succ\_nodes(n_i) = \{n_j \,|\, (n_i, n_j) \in \mathbb{E}\}$. All incoming edges of a node belong to the same word.

Each lattice has a unique start node $start(G)$ and a unique end node $end(G)$. These nodes can be assumed to be unique without loss of

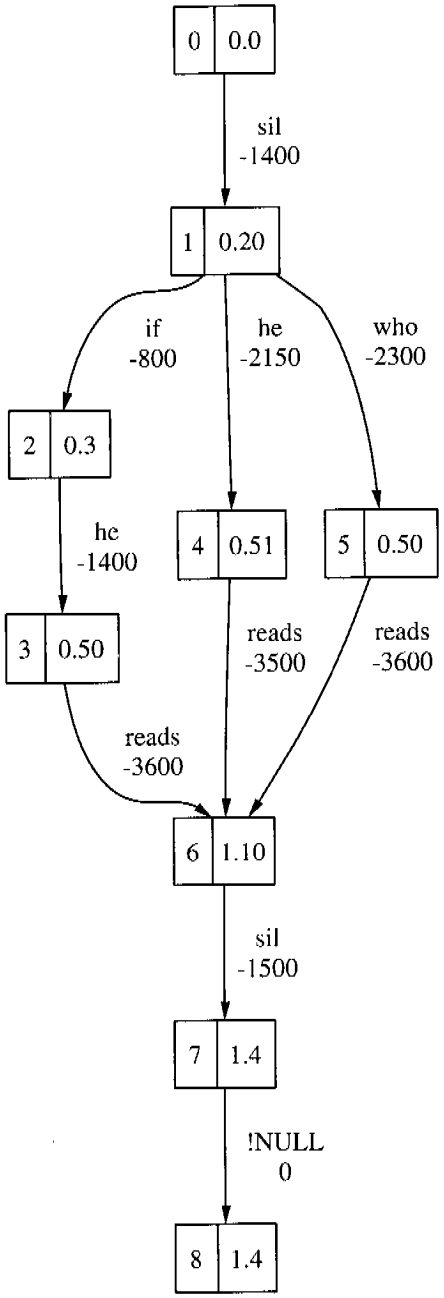


Figure 5.2: *Example of a word lattice. Edge weights correspond to the log-likelihood of the respective acoustic HMM word model. The language model score is not shown.*

generality because such a graph can always be constructed. For any pair of two nodes (n_i, n_j) there is exactly zero or one edge $e_k = (n_i, n_j) \in \mathbb{E}$.

A *partial path* is defined to be a sequence of edges $\langle e_1, e_2, \ldots, e_n \rangle$ where $end(e_i) = start(e_{i+1})$. Because there is at most one edge between two nodes it is also possible to equivalently define a path to be a sequence of adjacent nodes $\langle n_1, n_2, \ldots, n_n \rangle$ with $n_i \in pred\_nodes(n_{i+1})$. A *full path* is a path with the restriction that $start(e_1) = start(G)$ and $end(e_n) = end(G)$ respectively $n_1 = start(G)$ and $n_n = end(G)$. $paths(G)$ is the set of all full paths in graph G. The *score* of a path is the sum of scores along the path

$$score(e_1, \ldots, e_n) = \sum_{i=1}^{n} score(e_i)$$

assuming that the scores are log likelihoods or log probabilities.

5.3.2 Overview

There are basically two approaches to choose from. The simpler one uses the parsing score to re-rank the N-best list and can treat each N-best solution independently of the others. The final solution is restricted to an utterance present in the N-best list. The more complicated solution operates directly on the lattice. It takes advantage of the fact that the N-best solutions largely overlap which prevents it from repeatedly parsing common word sequences. The decision fell on the lattice approach. It was expected to be more powerful since the final solution is not restricted to the N-best paths but can rather be any path in the lattice.

The key idea is to add new edges to the lattice, one for each phrase found by the parser. The acoustic score of a new edge is computed by summing the acoustic scores of the edges subsumed by the parsing edge. The same applies to the language model score. The parsing score is set according to the parsing unit type (word unit, fragment unit or utterance unit). The best scored utterance can then be extracted in the usual way by dynamic programming.

As a lattice can be very large and parsing is an expensive operation, the lattice is not parsed directly. Instead, several measures are taken to keep the computational complexity within reasonable bounds. In short, the processing consists of the following steps:

1. *Pruning.* During decoding the size of the word lattice can be controlled by appropriately setting the beam search parameters. Once the decoding is complete the lattice size is reduced further by forward-backward posterior pruning.

2. *Non-word removal.* Word lattices can contain non-words which the parser might not be able to handle, e.g. denoting silence or speaker noise. The nodes and edges belonging to non-words are removed from the lattice. The scores of the removed edges are pushed to the remaining edges so that the total score remains constant.

3. *N-best extraction.* A stack decoder extracts the N-best paths from the word lattice. The N-best paths are used to guide the parser to process the most promising hypotheses first.

4. *Lossy compression.* In a lattice, the same word sequence can be represented by multiple paths with different acoustic scores due to the uncertainty of word boundaries in continuous speech. From a syntactical point of view timing is irrelevant. By ignoring acoustic scores and timing information a word graph which represents all word sequences of the lattice in compact form is created.

5. *Word graph parsing.* The N-best paths of the word lattice are mapped to the word graph and processed step by step. In each step a bottom-up chart parser produces all phrases which can be derived from the words of the current step.

6. *Creation of annotated lattices.* A new edge is created in the uncompressed word lattice for each phrase the parser has found. The word lattice enriched with the results of the parser is called *annotated lattice.* Annotated lattices are used to compute the acoustic and language model score of a phrase, since that information was removed during the lossy compression.

7. *First best extraction.* The best scored solution of the annotated lattice according to Eq. (5.2) is extracted using dynamic programming.

Although some of these or similar algorithms are already known from various works, they are restated here under a common notation

for the sake of clarity and reproducibility. The N-best extraction by the stack decoder and the first best extraction are excluded from the description since they are considered to be standard techniques.

5.3.3 Forward-Backward Posterior Pruning

Pruning applied during time-synchronous decoding (e.g. Viterbi beam search, [HAH01, p. 625]) is forced to come to a decision based on partial information, namely all observations up to the current point in time. Forward-backward pruning, however, operates on the entire utterance and is therefore more effective.

The pruning algorithm of the *SRI Language Modeling Toolkit* [Sto02] was applied. The algorithm is described below since it is not documented in the manual and no reference is given.

A *full path* was defined in Sec. 5.3.1 to be a contiguous sequence of edges or nodes from the designated graph start node to the designated graph end node, and $paths(G)$ stands for the set of all full paths in graph G. Now we additionally define $paths(n_i)$ to be the set of all full paths through node n_i. The score or log likelihood of a path was defined to be the sum of scores along the edges, assuming that all scores are log likelihoods or log probabilities. To explain how posterior pruning works let us assume in this section that the scores are not in log domain, but rather likelihoods or probabilities. This will simplify the formulas considerably, but the concept remains the same. The score of a path is then

$$score(\text{path}) = \prod_{e_i \in \text{path}} score(e_i)$$

The scaled posterior probability $\gamma(n_i)$ of node n_i is defined to be the sum of probabilities of all full paths going through that node divided by the sum of probabilities of all full paths in the graph:

$$\gamma(n_i) = \frac{\sum_{p \in paths(n_i)} score(p)}{\sum_{q \in paths(G)} score(q)} \quad . \tag{5.6}$$

Using the forward-backward algorithm the posterior probabilities can be efficiently computed [SO99]. Note that $\gamma(n_i) = 1$ if all full paths go through node n_i.

A node is pruned if its posterior probability lies below a given threshold. The threshold can be either absolute or relative. The disadvantage of absolute thresholds is that if the threshold is too high the pruned lattice will be empty. In order to have a more meaningful pruning parameter, we do not choose an absolute threshold directly but derive it from a pruning factor η instead. If η is chosen to be zero no pruning will occur. If it is set to one the lattice will be maximally pruned, but it is guaranteed that at a least one full path will remain.

This can be accomplished as follows: First the minimum posterior probability along the nodes of a path is determined for every path. The result is a list of minima, one minimum for each path. Then we find the maximum of these minima and use this value as absolute threshold corresponding to pruning factor one:

$$\theta(G) = \max_{p \in paths(G)} \left(\min_{n_i \in p} \gamma(n_i) \right) . \tag{5.7}$$

There is at least one path in the graph whose minimum node posterior equals $\theta(G)$. Thus if only nodes with a posterior strictly less than $\theta(G)$ are pruned at least one path will survive the pruning. Using a relative pruning factor η, a node n_i is pruned if

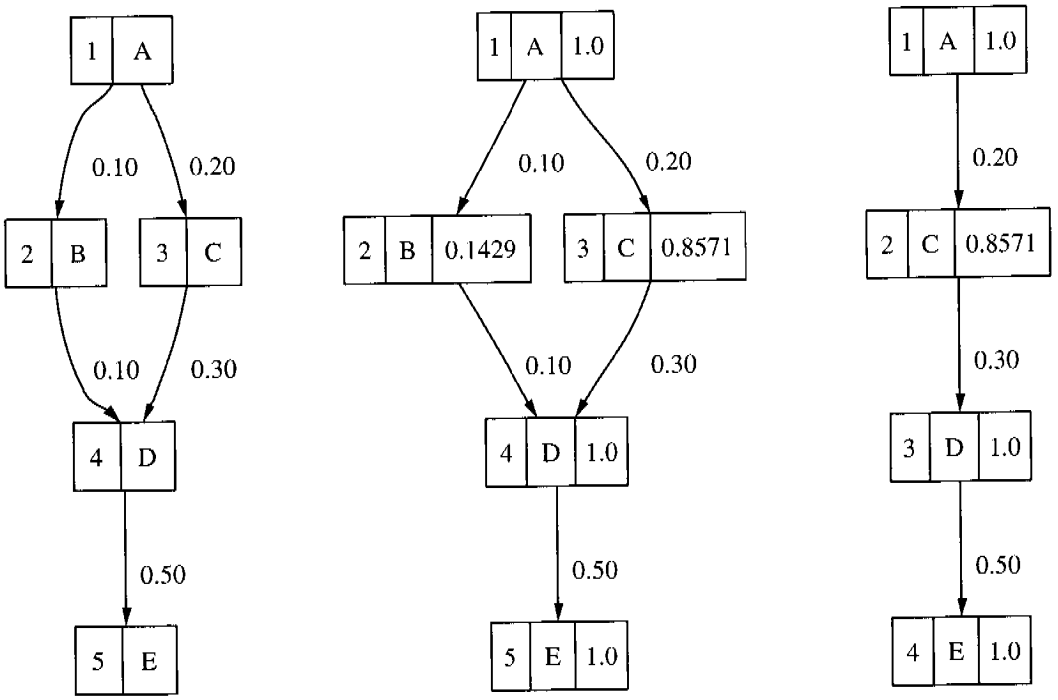
$$\gamma(n_i) < \eta \cdot \theta(G), \quad 0 \leq \eta \leq 1 . \tag{5.8}$$

An example of posterior pruning is given in Fig. 5.3.

5.3.4 Non-Word Removal

Lattices can contain non-word symbols denoting silence, noise or fillers, which not necessarily show up in the final transcription and which the parser cannot handle. The corresponding nodes and edges are removed from the lattice such that the total score of any word sequence remains the same by pushing the score of any removed edge into the remaining edges.

A node n_i is removed from the word lattice by first adding by-pass edges to the graph to preserve existing paths and then removing the node and all its edges from the graph. More formally, for any pair (e_p, e_s) of predecessor edges $e_p \in pred\_edges(n_i)$ and successor edges $e_s \in succ\_edges(n_i)$ a new edge $e_{\text{new}} = (start(e_p), end(e_s))$ is added



(a) *Lattice to be pruned. The weights on the edges are likelihoods (not log-likelihoods).*

(b) *Nodes annotated with scaled posterior probabilities* $\gamma(n_i)$. $\theta(G) = 0.8571$.

(c) *Lattice after posterior pruning with pruning factor* $\eta = 1$.

Figure 5.3: *Illustration of posterior pruning as described in Sec. 5.3.3*

to the graph with $score(e_{\text{new}}) = score(e_p) + score(e_s)$. Thus if a node has n incoming edges and m outgoing edges, $n \cdot m$ by-pass edges will be added to the graph. Node n_i and all its appendant edges are then removed. An example of non-word removal is given in Fig. 5.4 on the left.

5.3.5 Lossy Compression

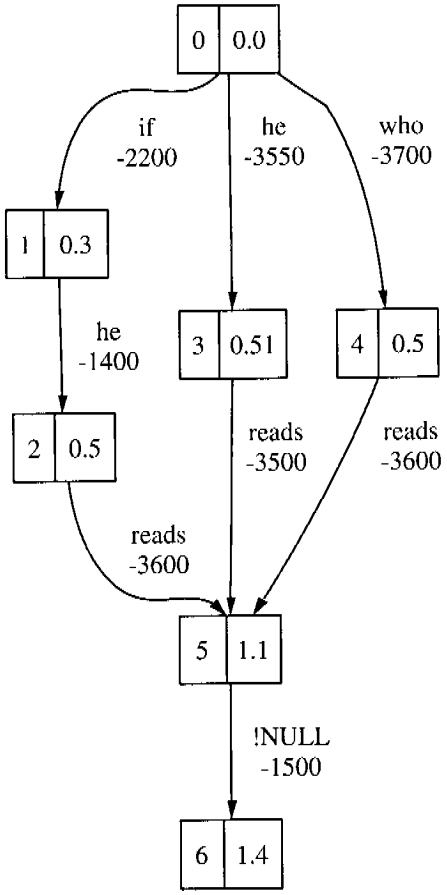
As parsing is a time consuming process, the lattice fed into the parser must be as small as possible. The lattice must thus be minimized with respect to the number of nodes and edges. The lattice size was already controlled during the decoding by the beam-search parameters and later by the posterior pruning factor. While these parameters affect the lattice word error rate, the compression described here does not. Because the parser does not know about scores and points of time in a signal we can remove these properties of the lattice and concentrate on word sequences. The transformation described in the next paragraph reduces the size of the lattice but preserves all word sequences. It is a simplified version of the algorithm described in [JH99].

Two nodes n_i and n_j can be merged into a single node n_{ij} if either $pred\_nodes(n_i) = pred\_nodes(n_j)$ or $succ\_nodes(n_i) = succ\_nodes(n_j)$. The merged node n_{ij} has the following properties: $pred\_edges(n_{ij}) = pred\_edges(n_i) \cup pred\_edges(n_j)$ and analogously $succ\_edges(n_{ij}) = succ\_edges(n_i) \cup succ\_edges(n_j)$.

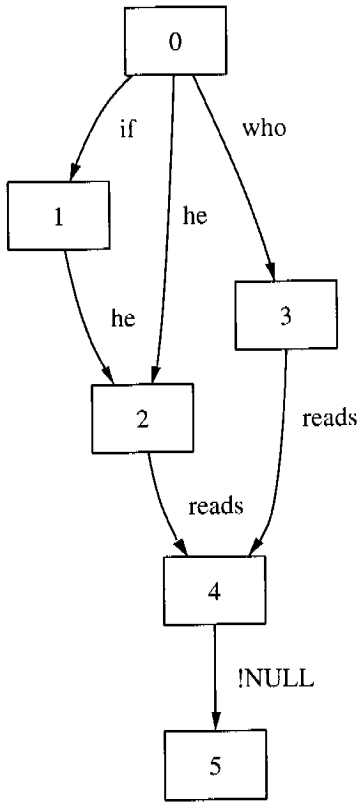
Lossy compression is illustrated in Fig. 5.4. The word "he" which occurred twice in the original lattice was merged into a single node while all word sequences are preserved. This compression is called lossy because scores are no longer available in the compressed lattice.

Expansion of Compressed Paths

At a later step (Sec. 5.3.7) it will become necessary to map the parser's paths given by a sequence of nodes of the compressed lattice $L_C = G(\mathbb{N}_C, \mathbb{E}_C)$ back to the corresponding paths in the uncompressed lattice $L_U = G(\mathbb{N}_U, \mathbb{E}_U)$ in order to compute their acoustic and language model scores. To do so we must be able to map each compressed node c to its set of uncompressed nodes $\mathbb{U}$. We define the bijective



(a) *The same lattice as in Fig. 5.2 but with silence (sil) removed as described in Sec. 5.3.4.*



(b) *The result of applying lossy compression to the lattice on the left as described in Sec. 5.3.5*

Figure 5.4: *Example of two lattice operations: (a) non-word removal and (b) lossy compression.*

function

$$\mathbb{U} = \text{map}(c) : c \rightarrow \mathbb{U} \quad c \in \mathbb{N}_C, \mathbb{U} \subseteq \mathbb{N}_U \ . \tag{5.9}$$

Such a mapping can be computed by adding a node equivalence set to each node of the uncompressed lattice. Before compression, this set is initialized to contain the node it is attached to. Whenever two nodes are merged their equivalence sets are unified. After compression the nodes are serially renumbered. A mapping table from nodes in the compressed lattice to nodes in the uncompressed lattice can be derived by listing the renumbered nodes and their set of equivalent nodes (which are kept fix during renumbering). The mapping table corresponding to the example given in Fig. 5.4 is shown in Table 5.2.

compressed node c	0	1	2	3	4	5
set of uncompressed nodes U	{0}	{1}	{2,3}	{4}	{5}	{6}

Table 5.2: *Mapping function corresponding to the example lattices in Fig. 5.4 .*

5.3.6 Lattice Parsing

As each word to be parsed will result in one or more lexical entries in the parser chart, it is desirable to keep the number of words small. In a lattice, the words are stored in the edges, so the parser has to create an entry for each edge. We defined that a word lattice has the property that all incoming edges of a node belong to the same word. Thus it is more efficient to convert the lattice to a word graph and parse the word graph nodes instead. The number of word graph nodes is usually smaller than the number of lattice edges, as illustrated in Fig. 5.5.

Parsing is the most complex and computationally demanding operation with regard to lattice processing. It is therefore important to guide the parser to work on promising hypotheses first, even more if we have to expect the parser not to be able to parse the full word graph due to time or memory constraints.

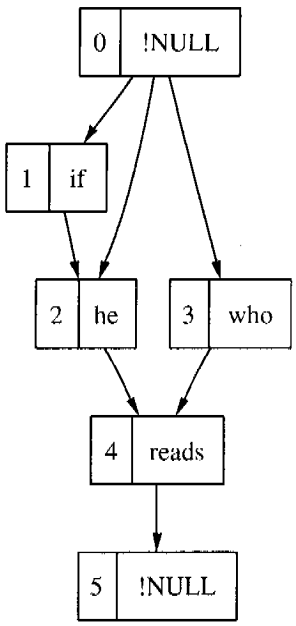


Figure 5.5: *The parser operates on word graphs. The word graph of this illustration corresponds to the the word lattice in Fig. 5.4(b). While the lattice representation would give rise to six lexical entries in the parsers chart (one for each non-null edge), the graph representation causes only four (one for each non-null node).*

Driving the Parser

In order to drive the parser to process promising hypotheses first, the N-best paths are incrementally extracted from the uncompressed word lattice using a stack decoder (A*-search). These uncompressed word lattice paths (Fig. 5.4(a)) are translated to compressed word graph paths (Fig. 5.5) by inverting the mapping function of Sec. 5.3.5. Null nodes are ignored. Only the paths in the baseline system's N-best list are processed, starting with the best hypothesis.

Processing means the parser builds all derivable syntactic structures. The phrases which have already been derived in previous steps are reused for efficiency reasons. This is repeated until all N paths are processed or a stop criterion, e.g. a time limit, is met. An example of N-best path extraction is given in Table 5.3. The parsing process can be stopped any time because the recognized utterance does not need to be fully parsed as it can be composed of partial parses which can even consist of a sequence of single words.

step	score	utterance	lattice edges	word graph nodes
1	-8550	he reads	$\langle 0, 3, 5, 6 \rangle$	$\langle 2, 4 \rangle$
2	-8700	if he reads	$\langle 0, 1, 2, 5, 6 \rangle$	$\langle 1, 2, 4 \rangle$
3	-8800	who reads	$\langle 0, 4, 5, 6 \rangle$	$\langle 3, 4 \rangle$

Table 5.3: *N-best paths extracted from the uncompressed word lattice given in Fig. 5.4(a). The word graph nodes refer to the word graph in Fig. 5.5.*

Parser Output

The parser returns a parsing path for any word sequence in the word graph which is part of the language described by a grammar. Such a parsing path is defined by a sequence of word graph nodes $e_{\text{parse}} = \langle n_1, \ldots, n_n \rangle$.

An example of parsing paths which continues the example from Table 5.3 is given in Table 5.4.

step	word sequence	path
1	he	$\langle 2 \rangle$
	reads	$\langle 4 \rangle$
	he reads	$\langle 2, 4 \rangle$
2	if	$\langle 1 \rangle$
	if he reads	$\langle 1, 2, 4 \rangle$
3	who	$\langle 3 \rangle$
	who reads	$\langle 3, 4 \rangle$
	who reads	$\langle 3, 4 \rangle$

Table 5.4: *Parsing paths. The step number in the leftmost column corresponds to the N-best path extraction step as given in Table 5.3. The node identifiers in the third column refer to the word graph shown in Fig. 5.5.*

5.3.7 Annotated Lattices

The aim of this section is to explain how the parsing paths can be used to introduce new edges into the uncompressed word lattice. The word lattice enriched with the results of the parser is called *annotated lattice* [vNBKN99]. As mentioned in Sec. 5.3.5, the parsing paths have no acoustic or language model scores. The lossy compression has removed them. To compute the scores of the parsing paths they must be mapped to their corresponding paths in the incompressed lattice. The conversion is performed in four steps.

1. The first step converts the word graph node sequence to a sequence of compressed lattice nodes. This step is trivial, since the node numbering is of the compressed word lattice and the word graph is the same.

2. The sequence of compressed lattice nodes is not yet complete. A parsing path representing a single word consist of a sin-

gle word graph node. In a word lattice however, a path needs at least two nodes to be properly defined, namely a node where it starts and a node where it ends. Therefore we have to prepend the node sequence $\langle n_1, \ldots, n_n \rangle$ of compressed lattice nodes with all possible predecessor nodes of the first node, which results in a new set of compressed lattice paths $\{p_i \mid p_i = \langle n_p, n_1, \ldots, n_n \rangle, n_p \in pred\_nodes(n_1)\}$.

3. Since a node c in the compressed lattice is related to a set of nodes $\mathbb{U} = \text{map}(c)$ in the uncompressed lattice, uncompressing a path results in a set of paths p_U, which is the cartesian product of the mapped node sets intersected with the uncompressed lattice:

$$p_U = (\text{map}(n_0) \times \text{map}(n_1) \times \ldots \times \text{map}(n_n)) \cap paths(L_C) \tag{5.10}$$

where map is the mapping function defined in Eq. (5.9). These paths can be efficiently computed by using a recursive depth first search without computing the full Cartesian product.

4. In the last step a new annotated edge e must be added to the lattice for every uncompressed path $p = (n_1, \ldots, n_n)$ which is longer than a single word with the following properties: $start(e) = n_1$, $end(e) = n_n$, $score_{ac}(e) = \sum_{i=1}^{n-1} score_{ac}(n_i, n_{i+1})$, the language model score is set analogous, and $word(e)$ is set to the word sequence along the parsing path. The parsing score $score_{parse}(e)$ of each edge is set to c_α, c_β or c_γ according to Eq. (5.3).

Example. Consider the utterance "he reads" which is found by the parser in step 1 (see Table 5.4). The parsing path $p_{\text{parse}} = \langle 2, 4 \rangle$ (Fig. 5.5) corresponds to the compressed lattice path $p_{LC} = \langle 2, 4 \rangle$ (Fig. 5.4(b)). By prepending the predecessor nodes $pred\_nodes(2) = \{0, 1\}$ to path p_{LC} we get two lattice paths $p_{LC_1} = \langle 0, 2, 4 \rangle$ and $p_{LC_2} = \langle 1, 2, 4 \rangle$. Now we compute the Cartesian product of the mapped nodes using the mapping from Table 5.2 for both paths:

$$p_{LC_1}: \quad \{0\} \times \{2, 3\} \times \{5\} = \{\langle 0, 2, 5 \rangle, \langle 0, 3, 5 \rangle\}$$
$$p_{LC_2}: \quad \{1\} \times \{2, 3\} \times \{5\} = \{\langle 1, 2, 5 \rangle, \langle 1, 3, 5 \rangle\}$$

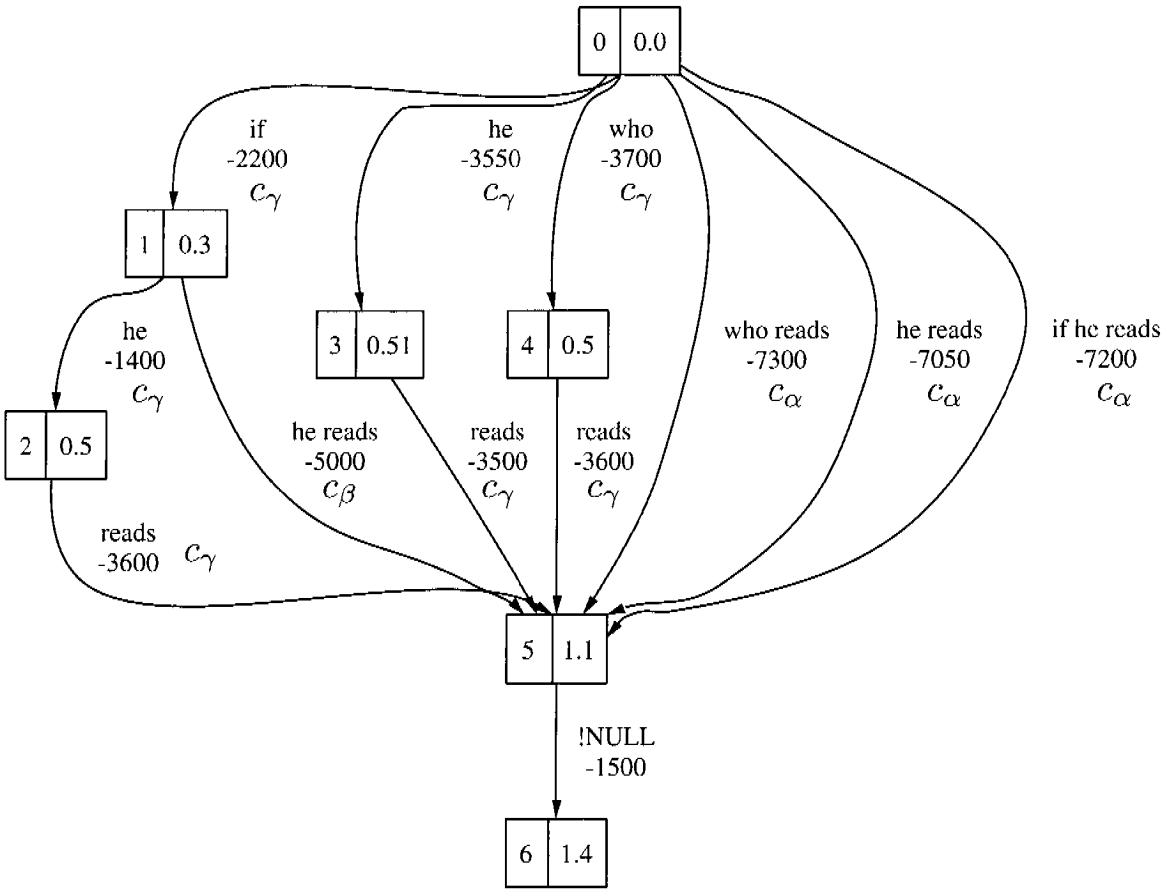


Figure 5.6: *Annotated word lattice.*

By intersecting the resulting four paths

$$\{\langle 0,2,5\rangle, \langle 0,3,5\rangle, \langle 1,2,5\rangle, \langle 1,3,5\rangle\}$$

with the lattice we find that two of them are valid, namely $\langle 0,3,5\rangle$ and $\langle 1,2,5\rangle$. So the uncompressed word lattice is annotated with two new edges: $(0,5,$"he reads"$,-7050)$ and $(1,5,$"he reads"$,-5000)$. The full annotated lattice is given in Fig. 5.6.

5.4 Grammar Formalism

So far, the grammar was assumed to be given. The architecture described in the last sections is independent of the grammar formalism used. However, for a real implementation, a few aspects deserve consideration.

A good grammar should accept as many grammatical word sequences as possible and at the same time reject as many ungrammatical word sequences as possible. Precision is the main requirement of a grammar to be used in our architecture: it only makes sense to favour the parsable word sequences if they are very likely to be correct. Note that since our approach can deal with unparsable word sequences, there is no need to artificially weaken the grammar rules, as is sometimes done to achieve a higher robustness.

However, it is also important that the grammar covers a wide range of syntactic constructions. It is necessary that the syntactically analyzable parts of the utterance are as large as possible, since only the words within an analyzable unit can be constrained. For instance, knowing a German verb's valency structure allows to constrain the inflectional endings of its objects (case, agreement of subject and finite verb). The disambiguation of inflectional endings is important since such endings are easily confused by the recognizer. In order to favour a given word sequence for obeying the valency constraint, the parser has to be able to derive a unit which contains the verb and all its objects. This in turn requires that each individual object is fully parsable.

The grammar formalism must not only be powerful enough to describe most of the linguistic phenomena, but must also allow the grammar writer to realize the phenomena in an elegant and outright way such that the system is clearly laid out [SW97, p. 33]. The most widely used formalisms in speech recognition, CFGs and DCGs, are not optimal choices [SW97, p. 45]. Different grammar formalisms were evaluated and discussed by Kaufmann in [Kau05b] in a project related to this thesis. As a grammar formalism, we have chosen *Head-Driven Phrase Structure Grammar* (HPSG) [PS87].

HPSG is a framework for linguistic theories. It uses linguistically motivated abstractions which substantially simplify the task of writing precise large-scale grammars. In addition to its linguistic adequacy, HPSG is well-suited for natural language processing applications. Ex-

isting systems like [Mül96] demonstrate that parsing efficiency can be reasonably high, even for large HPSG grammars which cover a substantial fragment of a natural language. A more detailed overview about this formalism and the reasons why we have chosen it is given in Appendix B.

5.5 Summary

This chapter describes an architecture which allows to compare a linguistically enhanced system with a standard speech recognizer. Any improvement of the word error rate can be clearly attributed to the linguistic component. The architecture is frequently used in spoken language understanding systems: a word lattice serves as an interface between an acoustic recognizer and a natural language processing module.

The word lattice is rescored such that grammatical phrases are slightly favoured by extending the MAP criterion by a parsing score. The parsing score is derived from the syntactic structures the parser found in the lattice. It relies on a few parameters which are optimized empirically on held-out data to minimize the word error rate.

This chapter describes the principles and implementation details of the post-processing approach. The next chapter concerns itself with the experiments and the data it was applied to.

Chapter 6

Lattice Parsing Experiments

The main goal of the experiments is to verify whether adding a general rule-based linguistic sub-system to a speech recognizer improves its accuracy.

This goal is divided into three sub-goals. First, we want to find evidence which supports the hypothesis that it is possible to significantly improve LVCSR accuracy by using a non-stochastic hand-written grammar and a parser in addition to a stochastic language model. Second, we want to find out which factors influence these improvements, and third, we are interested in identifying the limitations of the approach.

6.1 Introduction

The architecture described in the previous chapter was chosen because it allows to compare the output $\hat{W}$ of a standard recognizer with the output $\hat{W}^+$ of a linguistically enhanced recognizer (cf. Fig. 5.1). By comparing the word error rate WER_{parse} of the enhanced system with the baseline word error rate $\mathrm{WER}_{baseline}$ we can directly quantify the benefit of the linguistic component.

The measure used to evaluate the benefit of the parser throughout the experiments is the relative reduction of the baseline word error rate:

$$\Delta\text{WER} = \frac{\text{WER}_{parse} - \text{WER}_{baseline}}{\text{WER}_{baseline}} \,. \tag{6.1}$$

We also report the statistical significance for the Matched Pairs Sentence-Segment Word Error Test (MAPSSWE) and the McNemar Test on sentence level [GC89].

The first experiment compares the performance for speaker dependent monophone and triphone models with different number of mixtures and a trigram LM. The second experiment uses speaker adapted models and a 4-gram LM. The influence of the size of the N-best list is investigated in the third experiment, while the fourth experiment investigates the influence of out-of-vocabulary words.

6.1.1 Task

To initiate our work we looked for a task with a manageable complexity. We identified dictation texts for pupils as a suitable recognition task. We recorded the first 300 sentences (1892 words) from an exercise book for pupils in their third year of education (cf. Appendix A). The sentences were read aloud by a single male speaker and recorded with a headset microphone sampled at 16 kHz in an office environment with low background noise.

Although these sentences are rather simple, they comprise a wide variety of grammatical constructions, including verbal complexes with up to three verbs, prefix verbs, coordination of nominal and verbal projections, extraposition and genitive attributes. The sentence lengths range from a single word up to 14 words (cf. Fig. 6.1). Note that these sentences occur neither in the acoustic training corpus nor in the text corpus used for the estimation of the statistical language models.

6.1.2 The Linguistic Component

In a separate project an HPSG bottom-up chart parser and a German HPSG grammar were developed by Tobias Kaufmann [Kau05a].

The parser was especially designed for the application on word graphs. Typically, there is much overlap between the paths in a word

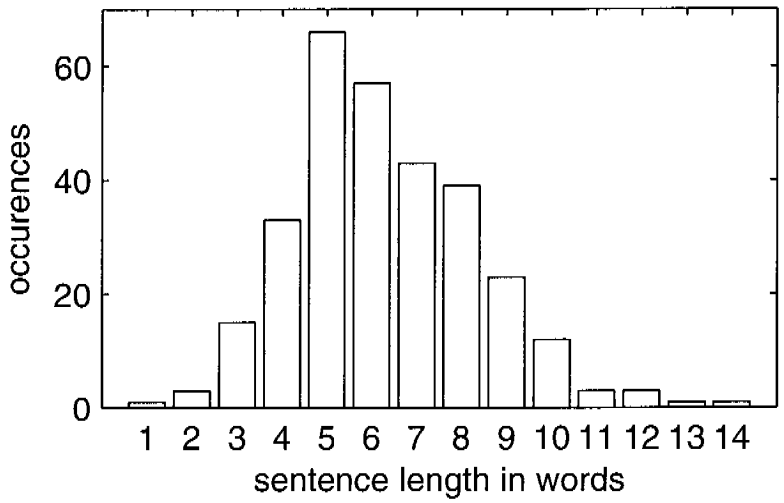


Figure 6.1: *Distribution of sentence lengths for dictation task.*

graph. To prevent identical word sequences from being analyzed several times, hypotheses from previously parsed paths are reused.

The grammar is largely based on the one proposed by [Mül99]. The semantic component of HPSG was not taken into account as we are only concerned with the grammaticality of utterances. We added some constructions which were observed in the experimental data and which occur frequently in language use. Among them are prenominal and postnominal genitives, expressions of quantity (2 dollars, 3 liters) and forms of address (Mr. and Mrs.). These constructions are very general and not specific to our task.

The current grammar covers many more phenomena than those which actually occur in the task. A list of test sentences which illustrates the abilities of the system is found in [Kau06].

Out of the 300 sentences in our task 278 (93%) are accepted by the grammar. In all experiments the parsing lexicon contains a basic set of closed-class words and those open-class words occurring the test sentences, which amounts to about 7'000 full word forms in total. The recognition dictionary is created from the parsing lexicon and contains exactly the same words. The pronunciation of a word follows the citation form given in [Dud90], pronunciation variations were not taken

into account. The open-class words include about 200 verbs, 340 nouns and 90 adjectives. For each verb, the possible valency structures (800 in total) have been determined using several sources independent of test set and development set [Dud99, ES76].

6.2 Speaker Dependent Dictation

The aim of the first experiment is to test our approach under optimal conditions. Optimal conditions means that the baseline word error rate should be low and the correct solution should be part of the lattice. These conditions allow the parser to choose the correct sentence among several alternatives. The best recognition results are achieved with speaker dependent models and no out-of-vocabulary (OOV) words. The influence of OOV words on the performance of our approach is scrutinized in a later experiment. We compare the performance for monophone and triphone models [BKP05b].

6.2.1 Data and Models

Continuous density HMMs have been trained by means of HTK [Cam02] with 7 hours of continuous German speech of a single male speaker sampled at 16 kHz in an office environment with low background noise using a headset microphone. The recording of the training data and the test data was performed by the same speaker as in Chapter 4. The 39-dimensional feature vector consists of 13 Mel-frequency cepstral coefficients (MFCCs) including the 0th coefficient, the delta and the delta-delta coefficients. The HMMs are three state left-to-right models with 8 or 32 Gaussian mixtures per state. For each of the 40 phonemes a context-independent monophone model was trained (called mono_8 and mono_32). Context-dependent cross-word triphone models were trained as well (tri_8 and tri_32). The states have been tied using a decision-tree based clustering according to yes/no questions regarding phonetic context, resulting in 3355 triphone models.

Bigrams and trigrams serve as statistical language models. The N-gram probabilities were estimated with the SRI language modeling toolkit [Sto02] on a 50 million words text corpus (German newspaper text and literature) using Good-Turing discounting for smoothing. The

N-grams were estimated for a recognizer vocabulary of 7k words. There are no out-of-vocabulary words.

On the test data, the task perplexity is 339.5 for the bigram LM and 274.9 for the trigram LM. The sentences in the test set do neither occur in the acoustic training corpus nor in the text corpus used for the estimation of the language models.

6.2.2 Experimental Setup

The test sentences were partitioned into a development set (200 sentences, 1255 words) and a test set (100 sentences, 637 words). The parameters λ, ip, c_α, c_β and c_γ introduced in Sec. 5.2.1 were optimized on the development set to minimize the empirical word error rate.

The development set was also used to manually choose the beam search and posterior pruning parameters. The parameters were set to values which substantially reduced the lattice sizes and at the same time yielded a reasonably high lattice accuracy. All optimizations were done for each HMM set (mono_8, mono_32, tri_8, tri_32) individually.

The HTK decoder performed a time synchronous Viterbi beam search storing the 5 best tokens per HMM state. The recognition network was a back-off bigram word-loop. The resulting lattices were rescored with the trigram language model and posterior pruning was applied ($\eta = 10^{-6}$). The 100 best scored recognizer lattice paths were processed by the incremental lattice parser. The parsing timeout was set to one minute on a 1 GHz UltraSPARC IIIi processor. Finally, the optimal word sequence was extracted by combining acoustic, language model and parsing scores.

6.2.3 Results

The results in Table 6.1 show that the linguistic component consistently decreased the word error rate for all acoustic models. The relative reduction of the word error rate using the parser in addition to the trigram language model was 48.6% in the best case and 28.9% in the worst case, as shown in Table 6.2.

The improvement using the parser is significant for the mono_32 model on a 0.001 level for the MAPSSWE test and at a level of 0.05 for the McNemar test. The other results are not significant.

WER	mono_8	mono_32	tri_8	tri_32
no LM	19.94	13.97	10.68	8.79
+ bigram	7.22	5.65	3.61	2.35
+ trigram	5.97	5.81	2.35	1.88
+ parsing	4.24	2.98	1.57	1.26

Table 6.1: *Word error rates in percent measured for different acoustic models and language models.*

model	ΔWER
mono_8	-28.9%
mono_32	-48.6%
tri_8	-33.3%
tri_32	-33.3%

Table 6.2: *Relative reduction of the word error rate on the test set due to extending the MAP criterion with a parsing score.*

6.2.4 Discussion

We have given evidence that rule-based knowledge capturing the structure of natural language can be a valuable information source complementary to N-grams. The additional score derived from the syntactic structures considerably decreased the word error for all acoustic models. To our knowledge, a comparable reduction of the word error rate due to applying a parser has not yet been reported for a similar task.

Our basic assumption was that the utterances to be recognized have to be grammatical to a sufficient degree. This assumption holds well for our experiment since most sentences in the test and development sets are covered by our grammar. Yet there is still room for improving recognition performance: the parser sometimes does not arrive at processing the correct utterance because parsing is stopped due to a

timeout. Timeouts are quite frequent, as parsing efficiency is still a major problem. We expect to decrease the word error rate further by improving the performance of the linguistic subsystem.

However, sometimes the criterion of grammatical correctness is not discriminative enough. Word lattices often contain several grammatically sound utterances. If a grammatically correct utterance is ranked before the correct utterance, the latter can not be chosen. For example, the acoustic recognizer sometimes confuses different verb tenses, which usually does not affect grammaticality.

6.3 Speaker Adapted Dictation

Most speech recognition systems today are either speaker independent or can be adapted to a given speaker. Therefore an experiment with a speaker adapted system was performed to evaluate the performance on a more realistic scenario [BKP05a]. Additionally, the baseline system was made more competitive by using an interpolated 4-gram LM instead of a 3-gram LM and by increasing the language model corpus size as well. Additionally, the LM was adapted to the task as well. The LM perplexities were thereby reduced by 27% on average.

6.3.1 Data and Models

Speaker-independent continuous density HMMs have been trained by means of HTK on the PhonDat 1 corpus [Pho90] which contains about 21 hours of clean continuous German speech of 200 speakers. The sampling rate was 16 kHz. The feature extraction was the same as in the speaker dependent experiment: the 39-dimensional feature vector consisted of 13 Mel-frequency cepstral coefficients (MFCCs) including the 0th coefficient, the delta and the delta-delta coefficients. In order to compensate linear channel distortions speaker-based cepstral mean subtraction was applied. The HMMs were three-state left-to-right models. For each of the 40 phones a context-independent monophone model with 32 Gaussian mixtures was trained. Context-dependent cross-word triphone models with 16 Gaussians were trained as well. The states have been tied using a decision-tree-based clustering according

to yes/no questions regarding phonetic context, resulting in 875 unique states and 2540 triphone models.

Additionally, 30 minutes of speech were recorded from the same speaker and the same recording conditions as in the speaker-dependent experiment. This data was used to adapt the speaker-independent acoustic models. We applied maximum likelihood linear regression (MLLR) and the maximum a posterior approach (MAP) in a supervised manner.

Interpolated back-off N-grams serve as statistical language models. Two text corpora were used: a 70 million words corpus (German newspaper text and literature) and a 30 thousand words corpus (texts from various dictation exercise books). The first one allows smoother estimates but the latter is closer to our task. The interpolation weights and the discounting strategy were optimized for low perplexity on the test data in order to get a competitive LM as our baseline.[1] The N-gram probabilities were estimated for a vocabulary of 7k words with the SRI language modeling toolkit [Sto02] using modified Kneser-Ney discounting for smoothing. The word classes of the class-based 2-gram were induced so as to minimize the perplexity of the model. There are no out-of-vocabulary words. The perplexities for the different LMs are given in Table 6.3.

The decoder of the baseline system performed two passes. In the first pass speaker-adapted monophone models and a bigram language model were used. The HTK [Cam02] decoder was used to create word lattices by time-synchronous Viterbi beam search. At each HMM state, the 5 best tokens were taken into account. In the second pass, the resulting lattices were rescored with cross-word triphones and a 4-gram language model and posterior pruning was applied ($\eta = 10^{-7}$).

The 20 best scored recognizer hypotheses were processed by the parser as described in Sect. 5.3. The parsing timeout was set to one minute on a 1 GHz UltraSPARC IIIi processor. Finally, the optimal word sequence was extracted by combining acoustic, language model and parsing scores as defined in Eq. (5.1) and Eq. (5.2).

[1] This is valid because the parameters optimized on the test data are part of the baseline system and not of our extension.

basic models	data	PPL
class-based 2-gram (500 classes)	70M	566
2-gram$_{30k}$	30k	389
2-gram$_{70M}$	70M	291
4-gram$_{70M}$	70M	222

interpolated models	PPL
decoding:	
2-gram$_{70M}$ + 2-gram$_{30k}$ + class-2-gram	251
rescoring:	
4-gram$_{70M}$ + 2-gram$_{30k}$ + class-2-gram	198

Table 6.3: *Perplexities (PPL) measured on the test data.*

6.3.2 Results

The relative reduction of the word error rate due to the parser is 27.0%. We tested the statistical significance of our results with the Matched Pairs Sentence-Segment Word Error Test (MAPSSWE) and the McNemar Test on sentence level [GC89]. The improvement over the baseline is significant on a 0.001 level for both tests. The detailed results are given in Table 6.4.

We have carried out further experiments (not shown in Table 6.4) to explore the influence of the baseline word error rate on the relative improvement. The results indicate that the relative improvement increases with the quality of the input word lattices: The lower the baseline word error rate, the higher the relative improvement. For a setup with only 30 seconds of adaptation data the baseline word error rate of 17.3% could be reduced by 20.2% relative.

	WER	ΔWER
baseline without 4-gram	7.24	
baseline	5.87	−19.0%
baseline	5.87	
baseline + parsing	4.28	−27.0%

Table 6.4: *Word error rates in percent for different language models: baseline system (2-gram during decoding and 4-gram lattice rescoring), baseline without 4-gram rescoring and enhanced system.*

6.4 Influence of Size of N-Best List

In the experiment of Sec. 6.3, the size of the N-best list was set arbitrarily to 20. The aim of this experiment is to investigate how the choice of N influences the word error rate of the enhanced system. To measure the influence of N on the word error rate, recognition experiments were carried out for different values of N, while the lattice remained the same. The parameters λ, ip, c_α, c_β and c_γ introduced in Sec. 5.2.1 were optimized for each N separately.

Results and Discussion

The results are presented in Fig. 6.2. It is observed that the word error rate does not change much for values beyond $N = 20$. The curve exhibits some peculiarities that deserve a more detailed analysis.

The curve is quite noisy, and for N=1 and N=3 the parsing system performs worse than the baseline. The main reason for this behaviour is a mismatch between the development set and the test set. To verify this statement, the experiment was run again, but that time all parameters were optimized on the test data. As can be seen in Fig. 6.3, without mismatch the parsing system performed always better than the baseline system and the curve is smoother.

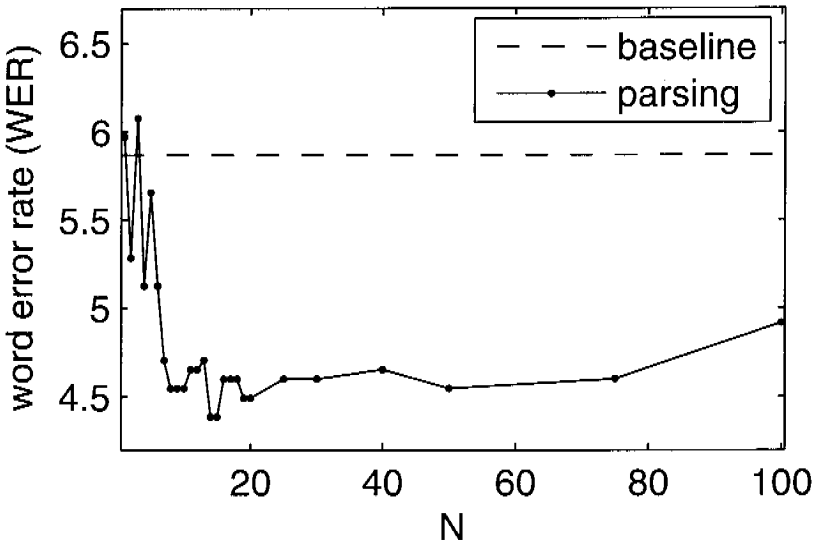


Figure 6.2: *Word error rate of the enhanced system vs. the size of the N-best list processed by the parser. The parameters were optimized on development data.*

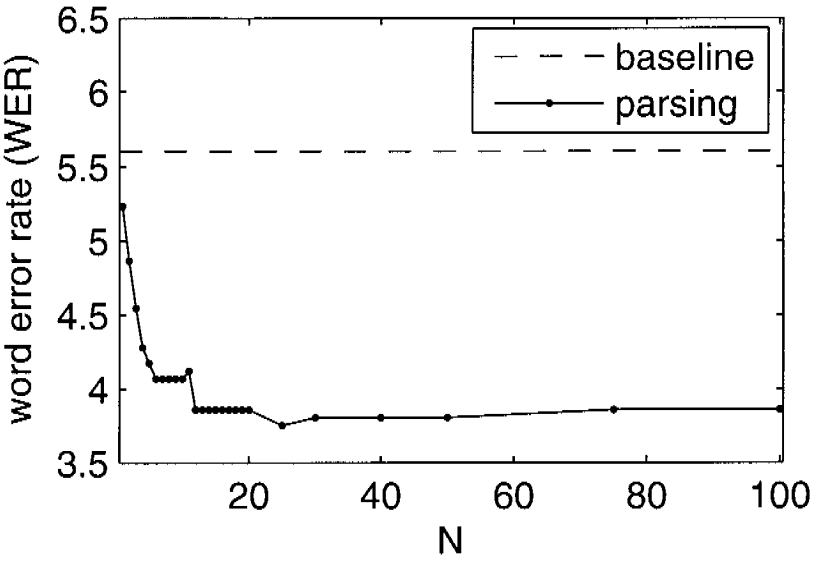


Figure 6.3: *Word error rate of the enhanced system vs. the size of the N-best list processed by the parser as in Fig. 6.2, but with the parameters optimized on the test data.*

The word error rate does not strictly decrease for increasing values of N. One reason for this behaviour is the stochastic nature of the optimization process (the parameters were optimized for each N independently). Another reason is the following. If N is increased, the parser is presented an additional word sequence. This word sequence may be wrong but nonetheless grammatical to a high degree. In order to prevent this wrong word sequence from being strongly favoured, the parsing scores must be lowered, which in turn may result in a decreased overall word error rate.

The word error rate for $N = 20$ in Fig. 6.2 is not the same as in Table 6.4, although both experiments were conducted on the same data and in the same manner. The difference is explained by the fact that the parser runs for at most one minute, which means that it can produce a different amount of parsing edges depending on the load of the machine. The lesson is that the timeout should have been replaced by a limit which is not based on time directly.

At a first glance it might be surprising that the parsing system performs better than the baseline system for $N = 1$ in Fig. 6.3, which means that best path in the annotated lattice is different from the baseline solution. The reason is that the word sequences of N-best lists overlap, and the syntactic structures found for the first best solution are valid for the overlapping part of all other N-best solutions as well. The parsing score allows to stick to the "correct" parts of the first best solution and allows to increase the language model weight, which in turn corrects some errors. Indeed, the optimal language model weight for the baseline is 10.7 and 22.3 for the parsing system with $N = 1$.

6.5 Influence of OOV

This experiment aims at investigating the influence of the out-of-vocabulary (OOV) rate to the relative improvement due to the linguistic component.

In contrast to the experiments in Sec. 6.2 through Sec. 6.4, the OOV rate in a real dictation scenario is greater than zero. If an OOV word occurs, the recognizer will make at least one error. Due to the influence of the N-gram one OOV word often results in more than one error. The correct solution cannot be found by the parser in the lattice, and the

syntactic structure of the utterance breaks apart, which weakens the capabilities of the linguistic component. The question is how much the improvement deteriorates with an increasing OOV rate.

The experimental setup is the same as in the speaker adapted dictation experiment from Sec. 6.3 with the exception that the vocabulary size was decreased to different sizes to simulate different OOV rates. The original vocabulary was reduced by removing the least frequent words. The word frequencies were measured on the same 70 million words corpus that was used to estimate the N-gram models.

6.5.1 Results

The relationship between the OOV rate and the relative improvement is shown in Table 6.5. As expected, the number of errors that can be corrected by the linguistic module decreases with increasing OOV rate.

OOV rate	– 4-gram	baseline	+parsing	oracle	ΔWER
0.00%	7.24	5.87	4.28	1.00	-27.0%
0.32%	7.29	6.77	5.23	1.64	-22.7%
0.69%	7.77	6.98	6.13	2.01	-12.1%
1.16%	10.10	8.93	8.46	4.18	-5.33%
1.64%	10.89	9.78	9.25	5.02	-5.41%
2.11%	11.42	10.52	9.83	5.60	-6.53%
2.54%	11.58	10.62	10.04	5.76	-5.47%
3.07%	12.63	11.26	10.62	6.29	-5.63%
4.70%	15.33	14.32	13.53	8.77	-5.54%
8.35%	21.35	20.56	20.56	14.69	0.0%

Table 6.5: *Influence of OOV rate to different word error rates: the baseline without 4-gram rescoring, the baseline, the parsing system and oracle. ΔWER denotes the relative improvement of the parser to the baseline.*

6.5.2 Discussion

The results indicate that the OOV rate is a limiting factor for the given approach. To achieve a statistically significant improvement the OOV rate must be below 1% for the given task. If this is also true for other tasks, then a German broadcast news transcription system would require a vocabulary of about half a million to one million words to reach that rate [GLA02].

When the OOV rate increases, both the baseline WER and the oracle WER increase. The difference between the baseline WER and the oracle WER is the maximum possible improvement for any post-processing method. At an OOV rate of 8.35% the baseline WER of 20.56% could be theoretically reduced by 28.6% relative to the oracle WER of 14.69%. Although there would be room for improvement, the linguistic system cannot take advantage of it.

The interpretation of the results is that missing words break up the syntactic structure of a sentence. The long distance dependencies are lost and less constraints can be imposed on the individual phrases. In the extreme case, the phrases consist of a single word or a few words only, where an N-gram is more competitive.

Chapter 7

Discussion

7.1 Integration of Linguistic Knowledge

The first question of this thesis was how rule-based knowledge can be incorporated into the statistical framework of a speech recognizer. Two architectures were used in this thesis to integrate linguistic knowledge. The first one integrated the linguistic knowledge into the speech recognizer, while the second one post-processed the recognizer output.

The advantage of the word spotting approach is its flexibility. It allows to apply morphological and syntactical knowledge at an early stage of speech recognition and allows to properly handle compound words. Since island processing is used, any kind of knowledge and strategy can be applied. This flexibility, however, comes at the cost of an increased complexity for the control module. The control module has to decide when to provide additional hypotheses by spotting new words and when to apply which knowledge to combine smaller fragments to larger ones. It is questionable whether the system would scale well for a more complex task than the one at hand.

In the second architecture, the use of lattices as intermediate data structure between a speech recognizer and a parsing component allows to treat the speech recognizer and the parsing component independently of each other. This modularization greatly simplifies the development of the whole system. The parsing component is no longer integrated

into the decoder. The lattice can be pruned and compressed effectively which leads to less computational load for the parser, which in turn allows to use a complex and linguistically sophisticated grammar formalism, such as HPSG.

The implementation of the word spotting approach was only capable of recognizing intra-grammatical utterances since it completely relied on a qualitative (strictly rule-based) language model. Robustness was achieved in the lattice parsing approach by scoring syntactic structures and allowing the recognized utterance to be composed of parseable fragments. The lattice parsing system favours intra-grammatical utterances but can recognize extra-grammatical utterances as well.

7.2 Influencing Factors

This section discusses the various aspects which influence the performance improvements gained by a linguistic processor, which was the second main question of this thesis. As a preliminary remark, it must be stated that it is difficult or even impossible to compare the results of different works that are based on different data and different conditions. Nevertheless, doing so might give us insight into why applying linguistic knowledge has been successful for some applications and why it failed in other cases. Some parts of the discussion are therefore of speculative character.

7.2.1 Influence of Baseline

In the OOV experiment of the lattice parsing approach (Sec. 6.5) we observed that with an increasing OOV rate not only the baseline WER got worse. The relative improvement due to parsing decreased as well. This would suggest that we can expect a larger gain by using a parser when the baseline performance increases. Our interpretation was that missing words break up the syntactic structure of a sentence and less constraints can be imposed.

The same was observed in the experiments conducted by Brill, who investigated whether humans are able to improve the output of a speech recognizer [BFHM98]. There were two scenarios: either they were restricted to select one of the 10-best recognizer hypotheses or they were

allowed to freely edit the output. The test was performed for three different corpora (Switchboard, Broadcast News and Wall Street Journal). Humans were able to consistently improve the output across all corpora in both scenarios. They used linguistic knowledge such as tense agreement, preposition choice, determiner choice and complete sentence vs. not complete sentence, as well as semantic and pragmatic knowledge. When the accuracy of the recognizer increased, the relative human improvement increased as well. Brill concluded that the better the transcription of the recognizer is, the more reliable contextual cues exist to determine the correct solution.

It has been repeatedly observed in the past that natural language knowledge can improve speech recognition, however, the improvement often disappeared when the baseline performance increased [MAD+95].

[NBKvN97] concluded that grammatical analysis does not give a clear advantage to improve recognition accuracy. The word accuracy was increased by parsing by 30.3% relative to a system which used no language model as baseline, but only 1.45% when the baseline used a bigram language model. Their findings are in contrast to ours, where a relative reduction of the word error rate of 27% was achieved relative to a 4-gram baseline.

Adding a linguistic component can only be successful if the knowledge it has is somehow complementary to the statistical knowledge of the N-gram language model.

7.2.2 Complementary Information

Our results suggest that a sophisticated qualitative language model is complementary to an N-gram model. The qualitative language model provides additional information not present in the traditional N-gram model. The same finding is reported in [WLH02] where interpolating a word N-gram with a parser LM led to a consistent improvement of word and sentence error rates. The intuitive reason is that a grammar is best at modeling long-distance dependencies and hierarchical structures, while an N-gram captures local and lexical dependencies.

7.2.3 Scoring

In the 1993 DARPA benchmark speech recognition evaluation on the ATIS task, SRI used a scoring scheme similar to ours [MAD+95]. A natural language score was computed as a combination of the number of phrases found needed to cover a hypothesis, a bonus for a full sentence and penalties for grammar rules which were dispreferred. The natural language score was scaled and added to the recognition score, and the combined score was used to rescore an N-best list. Even though their scoring was slightly more sophisticated than ours, the word error rate was only reduced by about 5%. Only when a kind of multi-level N-gram was used, which takes the fragment types found by the parser and semi-automatically chosen word classes into account, an improvement of 14.6% was achieved.

It seems that the number of fragments found by the parser is a more reliable indicator for the correctness of a hypothesis for our task than for ATIS. A consistent finding is that a fragment insertion penalty is advantageous [MAD+95, KKK+99].

Optimizing the parameters of our parsing score computation (Sec. 5.2.2) on development data was found to be very important. Without adjusting the parsing score on some held-out data, no improvement was observed on the test data.

7.3 Limitations

The third main question of the thesis was what the limitations of the linguistic approach are.

The wordspotting approach requires reasonable acoustic conditions in order to reliably detect short words, and it requires the utterances to be recognized to be intra-grammatical. Using a statistical language model additionally to the rule-based grammar would be beneficial.

The post-processing approach is more robust and can deal with extra-grammatical utterances as well. However, since it assumes that grammatical utterances should be favoured, the domain at hand should reflect that fact. A domain where this approach can be expected to work well is for example the dictation of business letters, reports or

manuals, where the text preferably consists of grammatically sound sentences.

It remains an open question how well the approach would work on conversational speech, since conversational speech often contains non-grammatical sentences and abounds with disfluencies like repetitions, restarts and partial words. It is not clear how well a rule-based system can cope with the phenomena of spontaneous speech. One has to note, however, that spontaneous speech is also challenging for N-grams due to the lack of large amounts of stylistically matching training data [DLW04].

The main limitation of the post-processing approach is the OOV rate. With increasing OOV rate the relative improvement of the parser degraded quickly, so that the recognizer lattice should be of a reasonable quality to make the approach work. While this requirement may be a limitation, it is at the same an advantage. It is often difficult to further improve an already well performing system, but such a system seems to be the best prerequisite for the parsing approach to perform well. Thus, the lattice parsing approach may become more interesting the better the recognizers get.

7.4 Outlook

Our results were produced for a task which is rather forthcoming in that it consists of grammatical sentences most of which are accepted by our grammar. As the presented results are convincing, a more difficult real-world task should be addressed, such as broadcast news transcription or the dictation of more complicated texts.

Also, wrong hypotheses are sometimes grammatical. Some sentences are grammatically completely sound, although a human would never analyse the sentence in the same way, so some sort of statistical information for disambiguation will be needed in future.

Seite Leer /
Blank leaf

Appendix A

Test Sentences

The test set consists of the first 300 sentences taken from a dictation book for pupils [Mül01]. Punctuation marks were omitted and all words were translated to lowercase.

```
000   ich bin neu in dieser schule
001   die klasse kenne ich auch noch nicht
002   mein nachbar heisst kevin
003   unsere lehrerin ist frau klein
004   sie sagt zu mir steffi
005   aber ich heisse stephanie
006   ich kann schon mehr schreiben als die anderen
007   in der pause bleibt anna bei mir
008   sie zeigt mir alles
009   bald kann ich das allein
010   heute kam der bus wieder zu spät
011   alle waren schon in der klasse
012   ich habe frau klein alles erzählt
013   sie hat mich getröstet
014   kevin hat mir geholfen
015   dann habe ich die richtige seite in meinem buch gefunden
016   morgen geht er mit mir zur haltestelle
017   dann bekomme ich auch einen sitzplatz
018   das hat kevin mir versprochen
019   ich freue mich schon darauf
020   unser schulbus ist oft ganz voll
```

021 alle sind dann sehr laut
022 die grossen kinder drängeln immer
023 sie helfen uns nicht
024 der fahrer sagt nie etwas
025 manchmal nimmt er nicht alle mit
026 die müssen dann eine stunde warten
027 im sommer ist das nicht so schlimm
028 aber im winter wird es kalt
029 bald sind ferien
030 ich freue mich schon sehr
031 dann kann ich den schulbus für einige zeit vergessen
032 heute ist der platz neben anna leer
033 dennis fehlt
034 ob er krank ist
035 keiner weiss es
036 nach der pause ist dennis da
037 er geht an den lehrertisch und spricht mit herrn bär
038 dann kommt er auf den platz neben mir
039 da wird er ganz rot
040 nach der stunde erzählt dennis mir alles
041 er hat verschlafen
042 im bett war es noch so schön
043 aber dann musste er sich beeilen
044 das kann jedem mal passieren oder
045 am freitag war ich zu früh in der schule
046 ich habe auf dem flur gewartet
047 dort traf ich anna
048 wir unterhielten uns über einen fernsehfilm
049 das war wohl zu laut
050 denn plötzlich ging eine klassentür auf
051 frau braun kam heraus und schimpfte
052 dann hat sie uns in einen gruppenraum gesteckt
053 aber jede in einen anderen
054 wir sollten lesen üben
055 meine grosse schwester hat ein mobiltelefon
056 anna und ich haben leider noch keins
057 sonst hätten wir uns unterhalten können
058 einmal lief ein hund durch einen bach
059 er trug in seinem maul ein stück fleisch
060 das wasser war wie ein spiegel
061 da sah der hund plötzlich das fleisch darin

062 gierig schnappte er danach
063 aber es war ja nur das spiegelbild
064 so verlor er durch seine dummheit seinen guten braten
065 es war einmal mitten im winter
066 die schneeflocken fielen wie federn vom himmel
067 da sass eine königin am fenster und nähte
068 das fenster hatte einen rahmen von schwarzem ebenholz
069 wie sie so nähte stach sie sich in den finger
070 da fielen drei tropfen blut in den schnee
071 das rot sah im schnee so schön aus
072 da wünschte sich die königin ein töchterchen
073 es sollte so schön sein wie blut schnee und ebenholz
074 das töchterchen wurde geboren und hiess schneewittchen
075 doch bald starb die königin
076 und der vater war mit schneewittchen allein
077 schneewittchen lebte mit dem vater allein im schloss
078 aber nach einem jahr heiratete der König wieder
079 die neue frau konnte schneewittchen nicht leiden
080 das mädchen war nämlich viel schöner als sie
081 die königin fragte jeden tag ihren spiegel
082 mit der antwort aber war sie nie zufrieden
083 darum sollte der jäger schneewittchen in den wald bringen
084 dort fand es bei den sieben zwergen ein zuhause
085 eines tages aber tauchte die böse stiefmutter auf
086 sie war verkleidet
087 schneewittchen kaufte von ihr einen vergifteten apfel
088 da konnten ihr auch die zwerge nicht mehr helfen
089 der vergiftete apfel hatte schneewittchen getötet
090 darum waren die sieben zwerge sehr traurig
091 sie legten das schöne mädchen in einen sarg aus glas
092 plötzlich kam ein königssohn in den wald
093 er fand schneewittchen in dem gläsernen sarg
094 als er es fort trug stolperte er
095 dadurch fiel dem mädchen das giftige apfelstück aus dem mund
096 da erwachte schneewittchen
097 der königssohn war von herzen froh und nahm es mit auf sein schloss
098 dort heirateten sie und lebten glücklich bis an ihr ende
099 eines tages kam besuch
100 es war onkel bert aus amerika

101 keins der kinder kannte ihn
102 darum waren alle besonders neugierig
103 onkel bert ist mein bruder
104 er ist vor zwölf jahren ausgewandert
105 ich habe ihn seitdem auch nicht mehr gesehen
106 eigentlich heisst er hubertus
107 onkel bert war lustig angezogen
108 er trug ein buntes hemd und einen breiten hut
109 papas bruder kann spannend erzählen
110 und er hat uns tolle sachen mitgebracht
111 onkel bert ist für drei wochen zu besuch gekommen
112 er sieht papa ähnlich
113 sie sind ja auch brüder
114 sabrina und ich sollen den kleinen koffer herbringen
115 onkel bert lächelt geheimnisvoll
116 wir müssen die augen schliessen
117 danach öffnet er den koffer
118 wir sind sehr gespannt
119 dann dürfen wir wieder gucken
120 da liegen zwei bunte pakete auf dem tisch
121 was da wohl drin ist
122 wir sind ganz aufgeregt und packen ein paket aus
123 ein komisches stofftier
124 zwei computerspiele
125 eine schwarze jeans
126 zwei becher für coladosen
127 turnschuhe
128 vier vögel für den weihnachtsbaum
129 eine leine für nero
130 zwei mützen und zwei sonnenbrillen mit spiegelglas
131 inzwischen hat sich nero ein kuchenstück gestohlen
132 das haben wir in der aufregung nicht gemerkt
133 vor den herbstferien hatten wir eine besondere prüfung
134 alle haben ihr fahrrad dazu mitgebracht
135 am dienstag stand ein polizist auf dem schulhof
136 er hat alle fahrräder geprüft
137 aber das hatten wir schon mit frau winter geübt
138 fast alles war in ordnung
139 nun hatten wir drei wochen zeit
140 wir sollten das fahren üben
141 die räder mussten geputzt werden und dann alle in ordnung sein

142 am freitag nach den herbstferien haben wir uns auf dem schulhof getroffen
143 alle hatten das fahrrad wieder dabei
144 auch der polizist war da
145 er hatte herrn sandmann mitgebracht
146 das ist ein fahrlehrer
147 die beiden haben uns die fahrradprüfung abgenommen
148 dabei ging es lustig zu
149 tina musste durch eine aufgemalte gasse fahren
150 aber sie lenkte ganz schief
151 dann kam ein kreis mit einer schlangenlinie dran
152 auf einer anderen strecke sollte man möglichst langsam fahren
153 dabei ist lars umgekippt
154 tina hat ganz laut gelacht
155 das war heute nur eine probe sagte herr sandmann
156 in der nächsten woche wird es ernst
157 im wald blüht es schon
158 der winter ist vorüber
159 im wald im gebüsch an den hecken und gräben blüht es schon
160 es sind die ersten frühlingsblumen
161 wir freuen uns darüber
162 diese blumen haben unter der erde zwiebeln oder verdickte wurzeln
163 darin speichern sie nahrung für den frühling
164 darum können sie schon so früh blühen
165 wenn man sie pflückt welken sie bald
166 auch in der vase im wasser halten sie nicht lange
167 man sollte die blumen lieber draussen blühen lassen
168 dann können sich alle darüber freuen
169 der zirkus kommt
170 gestern rollten sechs grosse wagen durch den ort
171 jetzt stehen sie auf dem festplatz
172 sie bilden einen grossen kreis
173 zehn männer bauen ein buntes zelt auf
174 in einer ecke kann man grosse käfige sehen
175 zwei kinder probieren eine übung auf einem pferd
176 wir dürfen zuschauen
177 in unserer klasse haben wir jetzt einen neuen schüler
178 er gehört zu den zirkusleuten
179 benny besucht ständig eine andere schule

180 am nachmittag beginnt die tierschau
181 das kostet nichts
182 morgen soll die erste vorstellung sein
183 der löwe und die maus
184 der löwe lag vor seiner höhle und schlief
185 auf dem felsen darüber spielte eine kleine maus
186 plötzlich fiel sie auf den löwen und weckte ihn
187 der löwe war ärgerlich aber er liess die maus laufen
188 eines tages war der könig der tiere in not
189 er hatte sich in einem netz verfangen
190 da kam das mäuslein und nagte das netz entzwei
191 jetzt musste sich der löwe bedanken
192 und er war froh dass er der maus damals nichts getan hatte
193 unsere katze ist wieder da
194 am freitag war unsere katze verschwunden
195 ich habe sie mit meinen freundinnen überall gesucht
196 niemand hatte unsere kitty gesehen
197 sie ist grauschwarz getigert
198 meine mutter hat im tierheim angerufen
199 aber dort war sie auch nicht
200 abends hat papa zufällig den besenschrank geöffnet
201 da kam ihm kitty entgegengesprungen
202 müll in der natur
203 die ganze klasse hat müll gesammelt
204 zwei stunden lang sind wir auf den sportplatz und in das kleine wäldchen gegangen
205 einige eltern waren auch dabei
206 frau hartmann hatte zehn grosse müllsäcke mitgenommen
207 was wir alles gefunden haben
208 auch ein fahrrad war dabei
209 tommy entdeckte sogar eine leere geldbörse
210 wir haben sechs müllsäcke gefüllt
211 unsere katze bellt wie ein kaninchen
212 das kaninchen singt wie ein wellensittich
213 der schwimmt und taucht wie eine robbe
214 unser huhn legt einen apfel
215 die banane hat rote bäckchen
216 zwei tomaten spielen fussball
217 abends geht die sonne auf
218 das schwein überholt den hund
219 auf der strasse fährt ein schiff

220 das flugzeug geht zu fuss
221 die strassenbahn fliegt über die autobahn
222 im gras reitet eine ameise
223 der storch telefoniert mit dem fisch
224 michael und mona vergessen nie ihre hausaufgaben
225 der igel schläft auf dem baum
226 frau meier findet keine fehler im diktat
227 jens hat in diesem jahr zweimal geburtstag
228 berlin liegt in japan
229 die turnstunde
230 am anfang der turnstunde wärmen sich alle auf
231 anne hüpft im kreis
232 steffi und kristin tanzen
233 dennis jagt vanessa
234 zwei kinder machen eine schiebekarre
235 oliver klettert am seil hoch
236 olaf überholt ihn am seil nebenan
237 andrea malt sebastian
238 andrea nimmt ein blatt papier und stifte
239 sie sieht sebastian genau an
240 der muss lachen
241 den kopf malt sie zunächst wie ein ei
242 dann setzt sie die augen ein
243 sie werden ganz blau
244 ohren und nase malt andrea hellrot
245 der mund lacht
246 man kann die zähne sehen
247 dann sieht sebastian das bild an
248 er lacht wieder
249 jetzt will er andrea malen
250 ich stehe jeden morgen früh auf
251 meine mutter weckt mich
252 dann gehe ich ins badezimmer
253 ich wasche mich und putze die zähne
254 dann ziehe ich mich an
255 für das frühstück lasse ich mir zeit
256 dann fühle ich mich auch besser in der schule
257 später bringt mich meine mutter zum schulbus
258 der ist leider oft sehr voll
259 und die grossen drängeln immer
260 in der schule treffe ich alle meine freunde

261 am meisten freue ich mich aber auf andrea
262 eigentlich ist die erdnuss gar keine richtige nuss
263 sie ist eine hülsenfrucht
264 wir essen sie gern mal zwischendurch oder beim fernsehen
265 am besten schmecken sie wenn sie geröstet sind
266 erdnüsse enthalten viele vitamine und andere wichtige stoffe
267 fledermäuse sind in der nacht unterwegs
268 sie haben übrigens mit mäusen oder vögeln nichts zu tun
269 sie wohnen unter leeren dächern in türmen und höhlen
270 am tag schlafen sie
271 dabei halten sie sich mit den füssen an der decke fest
272 der kopf hängt nach unten
273 manche leute haben vor fledermäusen angst
274 aber sie tun uns nichts
275 sie fressen käfer fliegen spinnen und andere krabbeltiere
276 wir sollten die fledermäuse schützen
277 unser hund heisst willi
278 er ist ein labrador mit fast ganz schwarzem fell
279 seine augen sind wunderschön braun
280 willi haben wir seit fünf jahren
281 papa sagt dass er jetzt im besten hundealter ist
282 mit willi wird es nie langweilig
283 er hält uns immer auf trab
284 denn er kann alles gebrauchen
285 mal nimmt er meine schuhe mal den handfeger
286 sogar spielsachen holt er sich
287 willi trägt alles zu seinem schlafplatz
288 wenn mal einer etwas sucht guckt er dort nach
289 meist hat man damit auch glück
290 einmal fand ich es aber nicht mehr lustig
291 da hatte er mein kuscheltier und mein schulheft gemopst
292 im mai und juni werden bei uns viele tierkinder geboren
293 trotzdem sieht man sie nur selten
294 die jungen werden von ihren eltern gut versteckt
295 das kleine reh heisst rehkitz
296 es wird im hohen gras verborgen
297 von dort läuft es erst weg wenn es schnell genug ist
298 junge häschen werden im freien geboren
299 sie haben gleich fell

Appendix B

HPSG

Remark

The purpose of this appendix is to give the interested reader a more detailed overview about HPSG, since it plays an important role in our lattice parsing approach.

Sections B.1 to B.7, and only these sections, were written by Tobias Kaufmann, who implemented the German HPSG grammar and parser used in this thesis. His work has not yet been published in English.

B.1 Introduction

Our approach to speech recognition requires a grammar which reliably accepts correct sentences (or phrases) and rejects incorrect ones. Developing a *precise* grammar makes specific demands on the grammar formalism to be used. Our German grammar is based on the Head-driven Phrase Structure Grammar (HPSG, [PS94]) formalism. In this section, it will first be explained why we preferred HPSG to more restricted formal grammars such as regular and context-free grammars. Next, we will introduce some of the more challenging language-specific and general phenomena which have to be modelled by a precise grammar. Finally, we will discuss some of the fundamental concepts on which HPSG is based and give a short characterization of our German grammar.

B.2 Why not Regular Grammars?

Regular grammars are the most restricted formal grammars defined in the *Chomsky hierarchy* (Chomsky type 3). For reasons to be explained below, regular grammars are obviously not suited for precisely modeling natural language. However, due to their simplicity they lend themselves to illustrating the consequences of a wrong choice of grammar formalism.

Regular grammars cannot model natural languages as they are unable to properly describe recursive structures. Recursive structures are omnipresent in natural language. An example for the English language are nested relative clauses:

(1) the cat_i hides_i

(2) the cat_i the dog_j chases_j hides_i

(3) the cat_i the dog_j the man_k hits_k chases_j hides_i

The subscripts indicate that each verb has a unique subject. A precise grammar needs to account for these dependencies in order to exclude the following ungrammatical sentences:

(4) *the cat the dog chases hides hides

(5) *the cat the dog chases hide

In example (4), the last verb has no subject, and in (5) the verb *hide* and its subject *the cat* do not agree.

One could argue that deeply nested sentences like (3) are very difficult to understand even for human listeners, and therefore a grammar for practical applications does not need to cover them. If the nesting level is limited, a regular grammar can simply enumerate all possible configurations. However, this technique substantially complicates the work of the grammar developer. To make things worse, there will generally be several interacting phenomena (e.g. nested relative clauses *and* agreement), such that the joint effects of these phenomena have to be enumerated. In the end, grammar writing will resemble more to enumerating the set of correct sentences than to describing the rules of a natural language.

Generally, if a grammar formalism does not provide the proper means to express the rules underlying a natural language, the task of writing precise large-coverage grammars will become virtually intractable.

B.3 Why not Context-Free Grammars?

The descriptive power of *context-free grammars* (CFG, Chomsky type 2) appears to be sufficient for modelling most natural languages. There are only a few known linguistic phenomena which are not context-free, for example the so-called cross-serial dependencies in Dutch and Swiss-German. But still CFGs are not very well suited for describing natural language. While they can naturally account for the recursive structure inherent to natural language, they do not lend themselves to modelling dependencies between constituents which do not appear on the right-hand side of the same production rule. Examples for such dependencies are given in the following section.

B.4 Some Phenomena to be Modelled

The following paragraphs present some of the linguistic phenomena which have to be described by a precise large-coverage grammar for German. For the purpose of illustration, English examples were chosen whenever possible.

Subcategorization refers to the phenomenon that a given verb requires specific syntactic arguments. For example, *sleep* requires a single noun phrase in nominative case (its subject). However, there is no usage of *sleep* which requires or allows a directional argument:

(6) the cat sleeps

(7) *the cat sleeps into the garden

The German language has a *relatively free word order*. In particular, the syntactic arguments may appear in almost any order:

(8) *dass [die frau] [dem mann] [ein buch] gab*
that [the woman] [the man] [a book] gave
'that the woman gave a book to the man'

(9) *dass [ein buch] [dem mann] [die frau] gab*
that [a book] [the man] [the woman] gave
'that the woman gave a book to the man'

In German clauses, several verbs form a *verbal complex.* Each verb contributes its own syntactic arguments to the verbal complex, and these arguments can again be freely permuted:

(10) *dass ihr der mann dieses buch zu lesen empfahl*
that her the man this book to read recommended
'that the man recommended her to read this book'

(11) *dass dieses buch ihr der mann zu lesen empfahl*
that this book her the man to read recommended
'that the man recommended her to read this book'

The verbal complex itself can be *discontinuous.* In the following example, the parts of the verbal complex are marked by subscripts indicating their position in the dependency hierarchy:

(12) *zu essen*$_3$ *hat*$_1$ *er den apfel versucht*$_2$
to eat$_3$ has$_1$ he the apple tried$_2$
'he has tried to eat the apple'

Another source of discontinuity is called *extraposition.* An extraposed phrase is separated from the phrase on which it depends and moved to the right. In the following example the relative clause *that climbed up the tree* is separated from *a cat*:

(13) a cat was saved [that climbed up the tree]

Finally, *extraction* refers to the phenomenon that something can be moved out of a phrase to the left, even across sentence boundaries. As an example we can look at the relative clause in (14). The word *whose cat* can be thought of having been moved from the position after *about* to the position before *wonder*:

(14) I wonder [whose cat] she was talking about

B.5 Head-driven Phrase Structure Grammar

There are many extensions of CFGs which were designed to better model natural language phenomena. *Head-driven Phrase Structure Grammar* can be considered as such an extension. HPSG employs context-free production rules, but is in fact an *unrestricted grammar* (Chomsky type 0). Typical HPSG grammars do not make full use of the Turing power offered by the formalism. Rather, they use HPSG's formal devices to express general rules underlying natural languages. Some of these formal devices will be presented subsequently.

Instead of non-terminal symbols, HPSG grammars use typed hierarchical data structures, so-called *attribute-value matrices* (AVMs). Fig. B.1 shows an AVM for the German verb *schläft* (*sleeps*). An AVM precisely describes the syntactic properties of a phrase or word. For instance, the value embedded under the SUBCAT attribute is a list containing all syntactic arguments of some word or phrase. For the AVM in Fig. B.1 this list contains a single element, which is a description of the verb's subject. To use such data structures rather than atomic non-terminal symbols has the following consequences:

- Grammar rules can access the information contained in the AVM *selectively*, i.e. they can operate on the information which is relevant to them and ignore all other information. This allows to write rules which are dedicated to very general linguistic phenomena. For example, there typically is a subcategorization rule which combines a phrase with another phrase if the latter matches some element of the former's SUBCAT list. This rule then can apply to verbs, adjectives, nouns, prepositions etc...
- AVMs allow HPSG to overcome the locality of a production rule, which is one of the problems of CFGs (cf. Sec. B.3). Information about a missing (i.e. moved) phrase can be introduced at some point into an AVM and propagated along several subsequent rule applications. Eventually, one of the derived phrases can be combined with a phrase matching the propagated description. A typical HPSG grammar employs several lists to handle phenomena like extraction and extraposition.

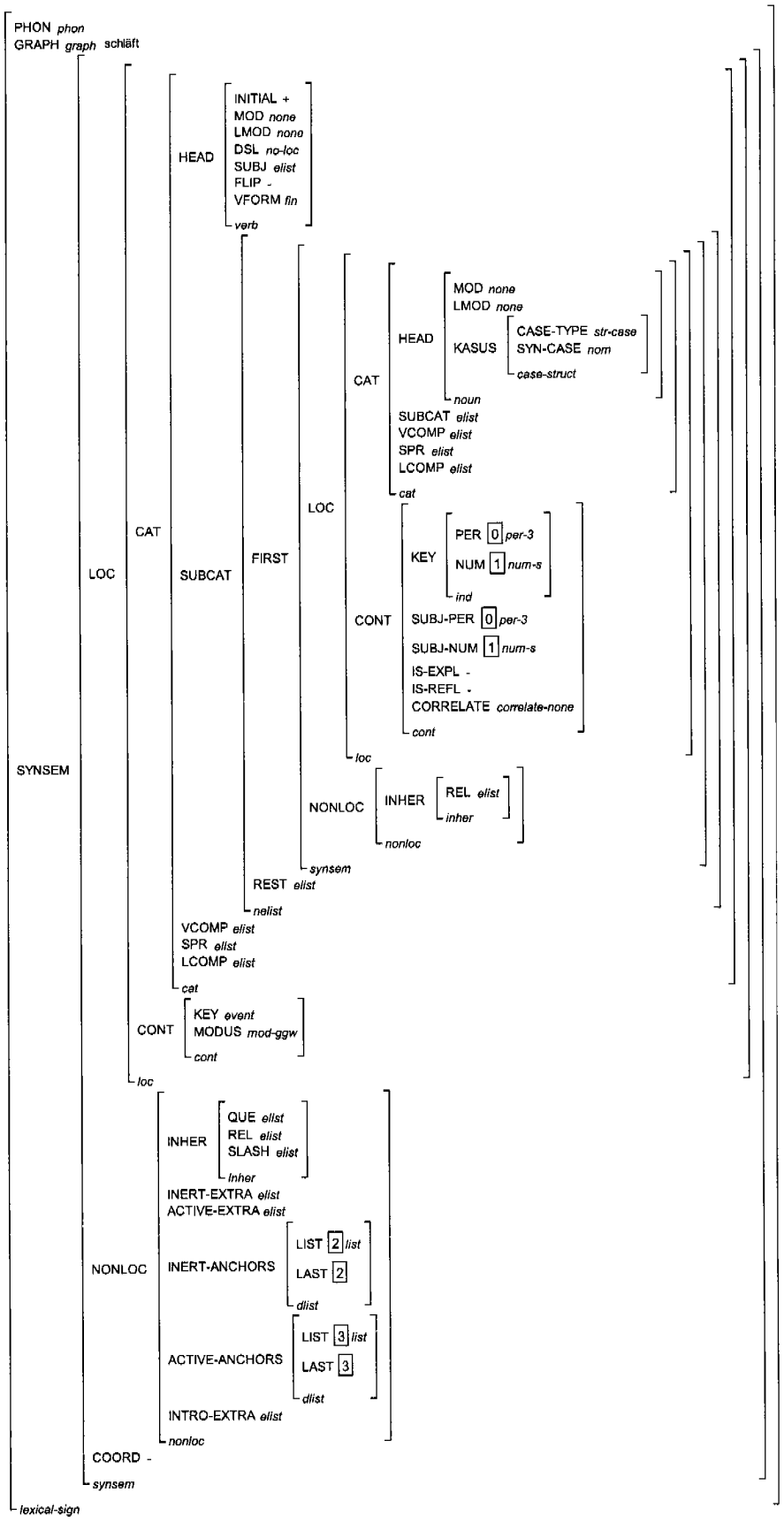


Figure B.1: *An AVM for the German verb "schläft" ("sleeps").*

- A substantial part of the grammar is not expressed in terms of rules but coded into the AVMs of the lexicon entries. For this reason, HPSG is called a *lexicalized grammar*. On the one hand this makes the structuring and maintaining of a lexicon a challenging task (see below), but on the other, it offers a convenient way to handle idiosyncrasies and exceptions. If some word behaves exceptionally in some way, it is often not necessary to change any grammar rules. Rather, one can just modify its AVM without affecting the rest of the grammar.

HPSG offers many additional formal devices. Many HPSG systems distinguish between *immediate dominance rules* which combine phrases and *linear precedence rules* which specify the relative order of the combined phrases. Some systems can directly represent *discontinuous constituents* [Mül96] which can be used to elegantly deal with some of the phenomena discussed in Sec. B.4. Some systems allow to impose *relational constraints* on AVMs. Such constraints typically are something like PROLOG predicates operating on AVM arguments. They are often used to define list operations such as concatenating lists and choosing or deleting list elements.

As mentioned above, by coding syntactic information in the representations of words, much of the grammar's complexity is transferred into the lexicon. HPSG offers several techniques to avoid redundancy in the lexicon. In particular, AVMs are organized in *inheritance hierarchies*: a partial description of an AVM with a specific linguistic interpretation is encapsulated in a macro-like construct. The specification of the AVM may itself be expressed in terms of macros. Another technique makes use of *lexical rules* to derive lexical entries from others. Among others, lexical rules are used to describe inflection and derivation.

B.6 Our German Grammar

The German grammar developed for this work is largely based on [Mül99]. It employs relational constraints and discontinuous constituents. Even though it consists of only 16 immediate dominance rules and 25 linear precedence rules it both covers a large fragment

of the German language and reliably rejects ungrammatical sentences. Table B.1 shows a classification of the immediate dominance rules.

5	general rules
11	construction-specific rules: verbal complex (3 rules) coordination (2 rules) prenominal and postnominal genitives (2 rules) relative clause (1 rule) interrogative clause (1 rule) optional determiners (1 rule) nominalization of adjectives (1 rule)

Table B.1: *Classification of the immediate dominance rules in our German grammar.*

B.7 Processing Issues

In the preceding sections we have mostly argued from a grammar developer's point of view. However, one could think of defining a grammar formalism tailored to the description of natural language which can be compiled into a formally simpler grammar, e.g. CFG. This has actually been done: A precursor of HPSG, *Generalized Phrase Structure Grammar* (GPSG, [GKPS82]) was designed as a context-free formalism which offers formal means to elegantly describe certain linguistic phenomena. A GPSG can directly be converted to a set of context-free production rules.

HPSG grammars are not context-free and therefore cannot generally be compiled into equivalent CFGs. However, an HPSG grammar can be approximated with a CFG [KK00]. This approximation usually results in a large set of context-free production rules. For example, [KKS00] reports to have approximated a Japanese HPSG consisting of 43 rule schemata by means of almost 20 millions of context-free production rules.

Bibliography

[BAHU+02] P. Beyerlein, X. Aubert, R. Haeb-Umbach, D. Klakow M. Harris, A. Wendemuth, S. Molau, H. Ney, M. Pitz, and A. Sixtus. Large vocabulary continuous speech recognition of broadcast news - the Philips/RWTH approach. *Speech Communication*, 37(1-2):109–131, May 2002.

[BB98] G. Bernardis and H. Bourlard. Improving posterior based confidence measures in hybrid HMM/ANN speech recognition systems. In *Proc. of ICSLP*, pages 775–778, Sydney, Australia, 1998.

[BCH+89] S. Boisen, Y.-L. Chow, A. Haas, R. Ingria, S. Roukos, and D. Stallard. The BBN spoken language system. In *Proceedings Speech and Natural Language Workshop*, pages 106–111, February 1989.

[BDB94] H. Bourlard, B. D'hoore, and J.M. Boite. Optimizing recognition and rejection performance in wordspotting systems. In *Proc. of ICASSP*, volume 1, pages 373–376, 1994.

[BDH+02] S. Brants, S. Dipper, S. Hansen, W. Lezius, and G. Smith. The TIGER treebank. In *Proceedings of the Workshop on Treebanks and Linguistic Theories*, 2002.

[BDPd+92] P.F. Brown, V.J. Della Pietra, P.V. deSouza, J.C. Lai, and R.L. Mercer. Class-based n-gram models of natu-

ral language. *Computational Linguistics*, 18(4):467–479, 1992.

[BFHM98] E. Brill, R. Florian, J. Henderson, and L. Mangu. Beyond N-grams: Can linguistic sophistication improve language modeling? In *COLING/ACL 1998*, pages 186–190, Montreal, Canada, 1998.

[BHK+97] T. Brants, R. Hendriks, S. Kramp, B. Krenn, C. Preis, W. Skut, and H. Uszkoreit. Das NEGRA-Annotationsschema. NEGRA project report, Universität des Saarlandes, Computerlinguistik, Saarbrücken, Germany, 1997.

[BKP05a] R. Beutler, T. Kaufmann, and B. Pfister. Integrating a non-probabilistic grammar into large vocabulary continuous speech recognition. In *Proceedings of the IEEE ASRU 2005 Workshop*, pages 104–109, San Juan (Puerto Rico), November 2005.

[BKP05b] R. Beutler, T. Kaufmann, and B. Pfister. Using rule-based knowledge to improve LVCSR. In *Proceedings of ICASSP*, pages 829–832, Philadelphia, March 2005.

[BP03] R. Beutler and B. Pfister. Integrating statistical and rule-based knowledge for continuous German speech recognition. In *Proceedings of the Eurospeech*, pages 937–940, Geneva, Switzerland, September 2003.

[Bur95] D. Burshtein. Robust parametric modeling of durations in hidden Markov models. In *Proc. of ICASSP*, volume 1, pages 548–551, Detroit, Michigan U.S.A, May 1995.

[Cam02] Cambridge University (S. Young et al.). *HTK V3.2.1: Hidden Markov Toolkit*, 2002. `http://htk.eng.cam.ac.uk`.

[CCP04] C. Collins, B. Carpenter, and G. Penn. Head-driven parsing for word lattices. In *Proceedings of the Meeting of the Association for Computational Linguistics*, pages 231–238, 2004.

[Cha97] E. Charniak. Statistical parsing with a context-free grammar and word statistics. In *Proceedings of AAAI/IAAI*, pages 598–603, 1997.

[Cho56] N. Chomsky. Three models for the description of language. *IEEE Transactions on Information Theory*, 2(3):113–124, September 1956.

[CR89] Y.-L. Chow and S. Roukos. Speech understanding using a unification grammar. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 727–730, 1989.

[CS89] Y.-L. Chow and R. Schwartz. The N-best algorithm: An efficient procedure for finding top N sentence hypotheses. In *Proceedings Speech and Natural Language Workshop*, pages 199–202, October 1989.

[DH73] R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis.* John Wiley & Sons, New York, 1973.

[DLW04] J. Duchateau, T. Laureys, and P. Wambacq. Adding robustness to language models for spontaneous speech recognition. In *Proceedings of Robust2004*, 2004.

[Dud90] *Duden "Aussprachewörterbuch"*, 3. Auflage. Bibliographisches Institut. Mannheim, Leipzig, Wien, Zürich, 1990.

[Dud99] *Duden "Das grosse Wörterbuch der deutschen Sprache in 10 Bänden"* , 3. Auflage. Mannheim, Leipzig, Wien, Zürich, 1999.

[DUHW05] J. Duchateau, D. H. V. Uytsel, H. V. Hamme, and P. Wambacq. Statistical language models for large vocabulary spontaneous speech recognition in Dutch. In *Proceedings of Interspeech*, pages 1301–1304, 2005.

[Dup93] P. Dupont. Dynamic use of syntactical knowledge in continuous speech recognition. In *Proceedings Third European Conference on Speech Communication and Technology*, pages 1959–1962, Berlin, 1993.

[EHRLR80] Lee D. Erman, Frederick Hayes-Roth, Victor R. Lesser, and D. Ray Reddy. The Hearsay-II Speech-Understanding System: Integrating Knowledge to Resolve Uncertainty. *Computing Surveys*, 12(2):213–253, June 1980.

[ES76] U. Engel and H. Schumacher. Kleines Valenzlexikon deutscher Verben. Gunter Narr Verlag Tübingen, 1976. Forschungsberichte des Instituts für deutsche Sprache, Nummer 31.

[Eur00] The European Telecommunications Standards Institute (ETSI). *ETSI STQ WI007 DSR Front-End Feature Extraction Algorithm*, 1.1.2 edition, April 2000.

[GC89] L. Gillick and S. Cox. Some statistical issues in the comparison of speech recognition algorithms. In *ICASSP*, pages 532–535, 1989.

[GKPS82] G. Gazdar, E. Klein, G. K. Pullum, and I. Sag. *Generalized Phrase Structure Grammar.* Cambridge: Massachusetts: Harvard University Press, 1982.

[GLA02] J.L. Gauvain, L. Lamel, and G. Adda. The LIMSI broadcast news transcription system. *Speech Communication*, 37(1-2):89–108, 2002.

[God92] D. Goddeau. Using probabilistic shift-reduce parsing in speech recognition systems. In *Proceedings International Conference on Spoken Language Processing*, pages 321–324, 1992.

[Gör88] G. Görz. *Strukturanalyse natürlicher Sprache.* Addison-Wesley, 1988.

[GVN+01] H. Glotin, D. Vergyri, C. Neti, G. Potamianos, and J. Luettin. Weighting schemes for audio-visual fusion in speech recognition. In *Proceedings of ICASSP*, volume 1, pages 173–176, 2001.

[HAH01] X. Huang, A. Acero, and H.-W. Hon. *Spoken Language Processing.* Prentice Hall PTR, 2001.

[Hau00] R. Hausser. *Grundlagen der Computerlinguistik.* Springer, 2000.

[HJ03] K. Hall and M. Johnson. Language modeling using efficient best-first bottom-up parsing. In *Proceedings of the IEEE Automatic Speech Recognition and Understanding Workshop*, pages 507–512, 2003.

[HJJ+99] M. Harper, M. Johnson, L. Jamieson, S. Hockema, and C. White. Interfacing a CDG parser with an HMM word recognizer using word graphs. In *Proc. ICASSP*, volume 2, pages 733–736, March 1999.

[HJM+94] M. Harper, L. Jamieson, C. Mitchell, G. Ying, S. Potisuk, P. Srinivasan, R. Chen, C. Zoltowski, L. McPheters, B. Pellom, and R. Helzerman. Integrating language models with speech recognition. In *"Proceedings of the AAAI-94 Workshop on Integration of Natural Language and Speech Processing*, 1994.

[HRB02] R. Hecht, J. Riedler, and G. Backfried. Fitting German into N-gram language models. In *TSD 2002, LNAI*, volume 2448, pages 341–345, 2002.

[HW94] A. Hauenstein and H. Weber. An investigation of tightly coupled time synchronous speech language interfaces using a unification grammar. In *Proceedings AAAI Workshop on Integration of Natural Language and Speech Processing*, pages 42–48, 1994.

[HWEP00] T. Hain, P. Woodland, G. Evermann, and D. Povey. The CU-HTK March 2000 Hub5e Transcription System. In *Proc. Speech Transcription Workshop*, 2000.

[Jel98] F. Jelinek. *Statistical Methods for Speech Recognition.* MIT Press, Cambridge, MA, 1998.

[JH99] M. T. Johnson and M. P. Harper. Near minimal weighted word graphs for post-processing speech. In *Proceedings of the IEEE Automatic Speech Recognition and Understanding Workshop*, pages 249–252, Keystone, USA, 1999.

[JHJ98] M.T. Johnson, M.P. Harper, and L.H. Jamieson. Interfacing acoustic models with natural language processing systems. In *Proceedings of the International Conference on Spoken Language Processing*, pages 2419–2422, 1998.

[JWS+95] D. Jurafsky, C. Wooters, J. Segal, A. Stolcke, E. Fosler, G. Tajchman, and N. Morgan. Using a stochastic context-free grammar as a language model for speech recognition. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 189–192, 1995.

[Kap73] R. M. Kaplan. A General Syntactic Processor. In R. Rustin, editor, *Natural Language Processing*, pages 193–240. Algorithmics Press, New York, 1973.

[Kau05a] T. Kaufmann. *Ein HPSG-System zur Anwendung in der Spracherkennung.* Zwischenbericht zum Nationalfonds-Projekt 105211-104078/1: Rule-Based Language Model for Speech Recognition. Institut TIK, ETH Zürich, Januar 2005.

[Kau05b] T. Kaufmann. *Evaluation von Grammatikformalismen in Hinblick auf die Anwendung in der Spracherkennung.* Zwischenbericht zum Nationalfonds-Projekt 105211-104078/1: Rule-Based Language Model for Speech Recognition. Institut TIK, ETH Zürich, September 2005.

[Kau06] T. Kaufmann. The ETH HPSG grammar: Test sentences and covered grammatical phenomena. `http://www.tik.ee.ethz.ch/~kaufmann/grammar/test.html`, 2006.

[Kay82] M. Kay. Algorithm schemata and data structures in syntactic processing. In S. Allén, editor, *Text Processing: Text Analysis and Generation, Text Typology and Attribution*, pages 327–358. Almqvist and Wiksell International, Stockholm, Sweden, 1982.

[Kie00] B. Kiefer, et al. Efficient and robust parsing of word hypotheses graphs. In W. Wahlster, editor, *Verbmobil.*

Foundations of Speech-to-Speech Translation, pages 280–295. Springer, Berlin, Germany, artificial intelligence edition, 2000.

[KK00] B. Kiefer and H.-U. Krieger. A context-free approximation of head-driven phrase structure grammar. In *Proceedings of the 6th International Workshop on Parsing Technologies*, pages 135–146, 2000.

[KKK+99] W. Kasper, B. Kiefer, H.-U. Krieger, C. Rupp, and K. L. Worm. Charting the depths of robust speech parsing. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics, ACL-99*, pages 405–412, 1999.

[KKS00] B. Kiefer, H.-U. Krieger, and M. Siegel. An HPSG-to-CFG approximation of Japanese. In *Proceedings of COLING*, volume 2, pages 1046–1050, Saarbrücken, Deutschland, 2000. Morgan Kaufmann Publishers.

[KOR92] A. Kannan, M. Ostendorf, and J.R. Rohlicek. Weight estimation in n-best rescoring. In *Proceedings of the DARPA Speech and Natural Language Workshop*, pages 455–456, February 1992.

[KS89] T. Kawabata and K. Shikano. Island-driven continuous speech recognizer using phone-based HMM word spotting. In *Proceedings of ICASSP*, pages 461–464, 1989.

[KTS89] K. Kita, Kawabata T., and H. Saito. HMM continuous speech recognition using predictive LR parsing. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 703–706, 1989.

[Lip97] R. P. Lippmann. Speech recognition by machines and humans. *Speech Communication*, 22:1–15, 1997.

[Low76] B. T. Lowerre. *The HARPY Speech Recognition System.* PhD thesis, Carnegie-Mellon University, Pittsburg, 1976.

[LST92] J. Lafferty, D. Sleator, and D. Temperley. Grammatical trigrams: A probabilistic model of link grammar. In

Proceedings of the AAAI Fall Symposium on Probabilistic Approaches to Natural Language, pages 89–97, 1992.

[MAD+95] R. Moore, D. Appelt, J. Dowding, J. Gawron, and D. Moran. Combining linguistic and statistical knowledge sources in natural language processing for ATIS. In *Proceedings of the ARPA Spoken Language Systems Technology Workshop*, pages 261–264, 1995.

[MAD03] K. McTait and M. Adda-Decker. The 300k LIMSI German broadcast news transcription system. In *In Proceedings of EUROSPEECH*, pages 213–216, 2003.

[MM90] H. Murveit and R. Moore. Integrating natural language constraints into HMM-based speech recognition. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 573–576, Albuquerque, 1990.

[Moo99] R. Moore. Using natural-language knowledge sources in speech recognition. In K. Ponting, editor, *Computational models of speech pattern processing*. Springer, 1999.

[MP02] A. Morris and H. Payne, S. Bourlard. Low cost duration modelling for noise robust speech recognition. In *Proceedings of ICSLP*, 2002.

[MPM89] R. Moore, F. Pereira, and H. Murveit. Integrating speech and natural-language processing. In *Proceedings Speech and Natural Language Workshop*, pages 243–247, Philadelphia, Pennsylvania, 1989.

[MSM94] M. P. Marcus, B. Santorini, and M.A Marcinkiewicz. Building a large annotated corpus of english: The Penn Treebank. *Computational Linguistics*, 19(2):313–330, 1994.

[Mül96] S. Müller. The Babel-system – an HPSG prolog implementation. In *Proceedings of the Fourth International Conference on the Practical Application of Prolog*, pages 263–277, London, 1996.

[Mül99] S. Müller. *Deutsche Syntax deklarativ: Head-Driven Phrase Structure Grammar für das Deutsche.* Niemeyer, 1999.

[Mül01] I. Müller. *OKiDOKi die Lernhilfe.* Schroedel, 2001.

[Nak89] S. Nakagawa. Speaker-independent continuous-speech recognition by phoneme-based word spotting and time-synchronous context-free parsing. *Computer Speech and Language*, 3(3):277–299, July 1989.

[NBKvN97] M.-J. Nederhof, G. Bouma, R. Koeling, and G. van Noord. Grammatical analysis in the OVIS spoken-dialogue system. In *Proceedings ACL/EACL Workshop on Spoken Dialog Systems*, pages 66–73, 1997.

[Net96] K. Netter. *Functional Categories in an HPSG for German.* PhD thesis, University of the Saarland, 1996.

[NM65] J. A. Nelder and R. Mead. A simplex method for function minimization. *Computer-Journal*, 7:308–313, 1965.

[OKA+91] M. Ostendorf, A. Kannan, S. Austin, O. Kimball, R. Schwartz, and J.R. Rohlicek. Integration of diverse recognition methodologies through reevaluation of N-best sentence hypotheses. In *Proceedings of Speech and Natural Language Workshop*, pages 83–87, Pacific Grove, California, 1991.

[ON97] S. Ortmanns and H. Ney. A word graph algorithm for large vocabulary continuous speech recognition. *Computer Speech and Language*, 11(1):43–72, 1997.

[Pau89] D.B. Paul. A CSR-NL interface specification. In *Proceedings Speech and Natural Language Workshop*, pages 203–214, Cape Cod, Massachusetts, October 1989.

[Pho90] PhonDat 1 - acoustic database of spoken standard German, 1990. Institut für Phonetik und Sprachliche Kommunikation, Munich.

[Pot45] R. K. Potter. Visible patterns of sound. *Science*, 102:463–470, 1945.

[PR96] T. Pfau and G. Ruske. Inkrementelle Erstellung von Worthypothesengraphen. Verbmobil-Report 107, Technische Universität München, 1996.

[PS87] C. J. Pollard and I. A. Sag. *Information-based Syntax and Semantics, Vol. 1.* Number 13 in CSLI Lecture Notes. CSLI Publications, Stanford University, 1987. Distributed by University of Chicago Press.

[PS94] C. J. Pollard and I.A. Sag. *Head-Driven Phrase Structure Grammar.* University of Chicago Press, Chicago, 1994.

[PTVF02] W.H. Press, S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery. *Numerical Recipes in C*, chapter 10.5. Cambridge University Press, 2002.

[PW80] F.C.N. Pereira and D.H.D. Warren. Definite clause grammars for language analysis - a survey of the formalism and a comparison with augmented transition networks. *Artificial Intelligence*, 13:231–278, 1980.

[Rab89] L. Rabiner. Tutorial on hidden Markov models and selected applications in speech recognition. *Proc. of the IEEE*, pages 257–286, February 1989.

[RGH+01] M. Rayner, G. Gorrell, B.A. Hockey, J. Dowding, and J. Boye. Do CFG-based language models need agreement constraints? In *Proceedings of the 2nd NAACL*, 2001.

[Roa01] B. Roark. *Robust Probabilistic Predictive Syntactic Processing.* PhD thesis, Brown University, 2001.

[Ros96] R. Rosenfeld. A maximum entropy approach to adaptive statistical language modelling. *Computer Speech & Language*, 10(3):187–228, 1996.

[Ros00] R. Rosenfeld. Two decades of statistical language modeling: Where do we go from here? *Proceedings of the IEEE*, 88(8):1270–1278, August 2000.

[SA90] R. Schwartz and S. Austin. Efficient, high-performance algorithms for N-best search. In *Proceedings of the Speech and Natural Language Workshop*, pages 6–11, June 1990.

[SBB+00] A. Stolcke, H. Bratt, J. Butzberger, H. Franco, V. Gadde, M. Plauche, C. Richey, E. Shriberg, K. Sonmez, J. Zheng, and F. Weng. The SRI march 2000 Hub-5 conversational speech transcription system. In *Proc. Speech Transcription Workshop*, 2000.

[SDR87] S. Steel and A. De Roeck. Bidirectional chart parsing. In *Proc. of the 1987 AISB Conference*, pages 223–235, University of Edinburgh, April 1987. John Wiley & Sons.

[Sen89] S. Seneff. TINA: A probabilistic syntactic parser for speech understanding systems. In *Proceedings Speech and Natural Language Workshop*, pages 168–178, Philadelphia, PA, 1989.

[SFI88] O. Stock, R. Falcone, and P. Insinnamo. Island parsing and bidirectional charts. In *Proc. of the 12th COLING*, pages 636–641, Budapest, Hungary, 1988.

[SO99] A. Sixtus and S. Ortmanns. High quality word graphs using forward-backward pruning. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 593–596, 1999.

[SRVDH+97] H. Strik, A. Russel, H. Van Den Heuvel, C. Cucchiarini, and L. Boves. A spoken dialog system for the Dutch public transport information service. In *International Journal of Speech Technology*, number 2, pages 121–131, 1997.

[SS94] A. Stolcke and J. Segal. Precise N-gram probabilities from stochastic context-free grammars. In *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*, pages 74–79, 1994.

[Sto02] A. Stolcke. *SRILM – The SRI Language Modeling Toolkit, Release 1.4.4.* SRI Speech Technology and Re-

search Laboratory, 2002. `http://www.speech.sri.com/projects/srilm`.

[SW97] I.A. Sag and T. Wasow. *Syntactic Theory: A Formal Introduction.* CSLI Publications, Stanford, 1997.

[Tra95] C. Traber. *SVOX: The Implementation of a Text-to-Speech System for German.* PhD thesis, No. 11064, Computer Engineering and Networks Laboratory, ETH Zurich (TIK-Schriftenreihe Nr. 7, ISBN 3 7281 2239 4), March 1995.

[Usz02] H. Uszkoreit. New chances for deep linguistic processing. In *Proceedings of the 19th International Conference on Computational Linguistics*, 2002.

[Ver00] D. Vergyri. Use of word level side information to improve speech recognition. In *Proceedings of ICASSP*, volume 3, pages 1823–1826, 2000.

[vNBKN99] G. van Noord, G. Bouma, R. Koeling, and M.-J. Nederhof. Robust grammatical analysis for spoken dialogue systems. *Natural Language Engineering*, 5(1):45–93, 1999.

[VTB00] D. Vergyri, S. Tsakalidis, and W. Byrne. Minimum risk acoustic clustering for multilingual acoustic model combination. In *Proceedings of ICSLP*, volume 3, pages 873–876, 2000.

[Wah00] W. Wahlster, editor. *Verbmobil: Foundations of Speech-to-Speech Translation.* Springer, Berlin, Germany, 2000.

[WH03] W. Wang and M. P. Harper. Language modeling using a statistical dependency grammar parser. submitted to the 2003 International Workshop on Automatic Speech Recognition and Understanding, 2003.

[WLH02] W. Wang, Y. Liu, and M.P Harper. Rescoring effectiveness of language models using different levels of knowledge and their integration. In *Proceedings of the IEEE*

International Conference on Acoustic, Speech and Signal Processing, pages 785–788, 2002.

[WMH00] Y. Y. Wang, M. Mahajan, and X. Huang. A unified context-free grammar and n-gram model for spoken language processing. In *Int. Conf. on Acoustics, Speech and Signal Processing*, pages 1639–1642, 2000.

[WSH04] W. Wang, A. Stolcke, and M.P. Harper. The use of a linguistically motivated language model in conversational speech recognition. In *Proceedings IEEE Intl. Conf. on Acoustics, Speech and Signal Processing*, pages 261–264, 2004.

[XCJ02] P. Xu, C. Chelba, and F. Jelinek. A study on richer syntactic dependencies for structured language modeling. In *ACL 2002*, pages 191–198, 2002.

[ZGG+91] V. Zue, J. Glass, D. Goodine, H. Leung, M. Phillips, J. Polifroni, and S. Seneff. Integration of speech recognition and natural language processing in the MIT Voyager system. In *Proc. ICASSP 91*, pages 713–716, 1991.

Curriculum Vitae

1975	Born on February 3 in Basel, Switzerland.
1982–1986	Primarschule Muttenz.
1986–1991	Progymnasium Muttenz.
1991–1994	Gymnasium Muttenz, Matura Typus C.
1995–2002	Studies in computer science, ETH Zurich.
April–June 2000	Internship as software engineer at Supercomputing Systems AG, Zurich.
June 2001	Sun Certified Java Programmer for Java 2 Platform.
spring 2002	Diploma in computer science (Dipl. Informatik-Ing. ETH).
2002–2007	Research assistant and PhD student at the Speech Processing Group at the Computer Engineering and Networks Laboratory, ETH Zurich.