Language Modeling for Speech Recognition of Czech

by

Pavel Krbec

Submitted to the Institute of Formal and Applied Linguistics at the Faculty of Mathematics and Physics, Charles University, Prague in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

CHARLES UNIVERSITY IN PRAGUE

March 2005

Author . . . .
Institute of Formal and Applied Linguistics at the Faculty of
Mathematics and Physics, Charles University, Prague
March, 2005

Certified by . . . .
Doc. RNDr. Jan Hajič, Dr.
Associate Professor
Thesis Supervisor

Accepted by . . . .
Prof. PhDr. Jarmila Panevová, DrSc.
Chairman, Department Committee on Graduate Students

I certify that this doctoral thesis is all my work, and that I used only the cited literature. The thesis is freely available for all who can use it.

Prague, March 11, 2005

Language Modeling for Speech Recognition of Czech

by

Pavel Krbec

Submitted to the Institute of Formal and Applied Linguistics at the Faculty of
Mathematics and Physics, Charles University, Prague
on March, 2005, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

In this thesis, I have designed and implemented a new language model for speech recognition.

The innovative part of the language model is the integration of the HMM-tagger component designed by myself. The HMM tagger can be used as a stand-alone disambiguation tool and, when combined with the hand written rules, it is currently the best disambiguation tool for Czech language in terms of error rate.

I have performed a speech recognition experiment on a highly inflectional language (Czech) where I tested the proposed language model. I have shown that the accuracy of the novel language model outperforms other state–of–the–art Czech language models.

Thesis Supervisor: Doc. RNDr. Jan Hajič, Dr.
Title: Associate Professor

Acknowledgments

I would like to thank to my advisor Jan Hajič for his scientific guidance and for persuading me to start the PhD study in the field of statistical linguistics.

I was fortunate enough to work with Prof. Frederick Jelinek and some of the best professionals in the area of speech recognition in the Center for Language and Speech Processing (CLSP) in Baltimore. I found excellent friends and colleagues in Asela Gunawardana (in his case also a great rugby teammate), Ciprian Chelba, Dimitra Vergyri, Vaibhava Goel and Peng Xu during my stay at CLSP. It was also there where I have realized what I would like to accomplish in the field of speech recognition. I would also like to express my gratitude to my Czech colleagues Pavel Ircing, Petr Podveský, Pavel Kvetoň, Nino Peterek, Martin Čmejrek and Jan Cuřín who have been always willing to share and exchange ideas on my research topic. Special credit goes to all my colleagues at IBM Research and in particular to the embedded speech recognition group, which is the best research team I could have ever joined.

Finally I would like to thank to my wife Anna. Without her support and motivating encouragement the thesis would never be finished.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Automatic speech recognition (ASR) is a process when a speech recognizer transcribes speech into text. The recognizer is usually based on a finite vocabulary which restricts the set of words to be transcribed. The term "automatic" means that the actual process of transcribing is done without any human aid.

1.1 History of Speech Recognition

The first machine to recognize human speech was a celluloid dog, the "Radio Rex". The simple electromechanical toy from 1920 was capable of jumping, when it's name was spoken.

In 1952, as government funding research began to gain momentum, Bell Laboratories developed an automatic speech recognition system that successfully identified the digits 0-9 spoken to it over the telephone. The struggle to create a robust speech recognizer continued through sixties with no immediate success [47]. The main problem of these early systems was that they were able to recognize discrete speech only[1], not continuous speech.

In 1969 John Pierce[2] of Bell Laboratories made the statement that automatic speech recognition will not be a reality for several decades because it requires artificial

[1] Speech, where words are separated by longer pauses than usual, to make the recognition easier.

[2] John Pierce's satellite research was awarded with the Draper Prize, one of engineering's top honors, in 1995.

intelligence. Only few years later the "Hidden Markov Modeling (HMM)" approach to speech recognition was invented by Lenny Baum of Princeton University and shared with several ARPA (Advanced Research Projects Agency) contractors including IBM.

The real breakthrough came in 1971 when DARPA (Defense Advanced Research Projects Agency) established the Speech Understanding Research (SUR) program to develop a computer system that could understand continuous speech. This was the largest speech recognition project ever.

In 1984 IBM demonstrated the world's first 5000-word vocabulary speech recognition system, achieving 95% accuracy. Running on three, six-foot-tall array processors and a 4341 mainframe, with a user interface running on an Apollo computer, this system could take discrete (word–at–a–time) dictation from a speaker trained to the system. The same company introduced the first dictation system, called IBM Speech Server Series.

As the available memory size and CPU power increased, first commercial applications appeared in the middle of nineties of the 20th century. It was IBM again which introduced ViaVoice speech recognition software as part of the operating system OS/2 in 1996. In 1997 Dragon introduced product called "Naturaly Speaking", the first continuous speech recognition package available.

1.2 Speech Recognition Today

There are three main areas of application, each requiring a different approach to speech recognition technology.

The first one is the embedded speech recognition. For certain devices the traditional input via keyboard is impossible. The limiting factor can be the size of the device or the way the device is operated. Hardware resources are usually limited and it is common that the FPU (floating point unit) is not present. Embedded speech recognition is, based on the current marketing studies, the most promising field for speech recognition in the near future. Target devices are smart–phones, hand–held computers, navigation systems, or medical devices requiring hands free operation.

Automotive industry belongs also to the embedded group.

The second area where the need of speech recognition was quickly discovered is the customer support call centers. The telephony applications usually combine speech recognition and speech synthesis into the IVR (Interactive Voice Response) systems. These "people–free" systems are useful and cost effective for the companies that employ them. The hardware platform for telephony applications is usually a cluster of servers or a supercomputer capable of handling several sessions at the same time.

The last and from the scientific point of view the most interesting area is the large vocabulary continuous speech recognition (LVCSR). The famous author of the cryptographic software PGP Phil Zimmermann discusses the moral aspects of running speech recognition software on all phone calls [57] and analyzing the calls for subversive traffic. Fortunately the target application of most LVCSR systems is much less Orwellian. This thesis deals with problems directly related to LVCSR systems.

1.3 Motivation for Speech Recognition of Czech

Current state-of-the-art technology in speech processing allows to build real time speech recognition systems with vocabularies containing tens of thousands words. The HMM approach, as used in all modern recognizers, has permitted us to improve the error rates significantly over the last decade by simply collecting more training data. Further research in acoustic modeling has led to new adaptation techniques. It was both supervised and unsupervised acoustic adaptation, that allowed to build true speaker-independent systems [22]. As discussed in section 1.1 the speech recognition effort started in the United States and it was American English which was in the spotlight in the early stages.

The scientific community started to investigate other languages and modifications of recognizers were introduced, which delivered good accuracy results for those new languages. By introducing the pitch into the acoustic model, for example, the HMM framework started to work well for Chinese. It is not the acoustic of the language which makes the recognition hard. It is the complexity of the language which makes recognition more challenging.

The extra complexity of a given language when compared to English can be attributed to two major aspects. It is the free word order which makes things complicated - English on the other side has very strict word ordering. The other factor is the size of the vocabulary. The size of the vocabulary not only slows down the speed of a recognizer, it also introduces a much harder problem – the data sparseness as discussed in chapter 4.

Czech is a language where both of the above mentioned aspects are combined. It is highly inflectional and thus the vocabulary size for achieving reasonable vocabulary coverage (the percentage of words in the running text that belong to the vocabulary) needs to be extremely high. The free word order can be demonstrated on the sentence "Pavel má tlusté vepřové rád."[3] This sentence can not be easily said in any other word order in English. In Czech nearly all the 120 permutations are possible. We will discuss in chapter 4 the proposed solutions to both free word order and data sparseness. Our intention is to introduce a method that will solve both these aspects for Czech. In case we succeed, the same method can be adopted to other inflective languages.

[3] Pavel likes fat pork.

Chapter 2

Hidden Markov Model

2.1 Introduction

The work in this thesis is based on the concept of hidden Markov model (HMM). This concept is used in many different contexts, other than speech recognition only. We will apply it to the acoustic and language modeling, to the search in the recognizer, and to the tagger.

2.2 Hidden Markov Model

2.2.1 Definition

A hidden Markov model is defined by the output observation alphabet

$$O = \{o_1, o_2, \ldots, o_M\}, \tag{2.1}$$

by the set of states representing the state space Ω (for our purpose s_t denotes state at time t)

$$\Omega = \{1, 2, \ldots, N\}, \tag{2.2}$$

and by the transition probability matrix

$$A = \{a_{ij}\}, \tag{2.3}$$

where a_{ij} is the probability of taking a transition from state i to state j, i.e., $P(s_t = j \mid s_{t-1} = i)$. The definition further requires the output probability matrix B

$$B = \{b_i(k)\}, \tag{2.4}$$

where $b_i(k)$ is the probability of emitting symbol o_k when state i is entered. Let $X = X_1, X_2, \ldots, X_t, \ldots$ be the observed output of the HMM. The state sequence $S = s_1, s_2, \ldots, s_t, \ldots$ is hidden, and $b_i(k)$ can be rewritten as:

$$b_i(k) = P(X_t = o_k \mid s_t = i). \tag{2.5}$$

We need to further define the initial state probability distribution $\pi = \{\pi_i\}$ where

$$\pi_i = P(s_0 = i) \qquad 1 \leq i \leq N. \tag{2.6}$$

Since a_{ij}, $b_i(k)$ and π_i are all probabilities, they must satisfy the following constrains:

$$a_{ij} \geq 0, \quad b_i(k) \geq 0, \quad \pi_i \geq 0 \quad \forall i, j, k \tag{2.7}$$

$$\sum_{j=1}^{N} a_{ij} = 1 \tag{2.8}$$

$$\sum_{k=1}^{M} b_i(k) = 1 \tag{2.9}$$

$$\sum_{i=1}^{N} \pi_i = 1 \tag{2.10}$$

The specification of an HMM thus includes two constants N and M, representing the total number of states and the size of observation alphabets, observation alphabet O, and three matrices of probabilities A, B, π. For the sake of simplicity we will use the following notation

$$\Phi = (A, B, \pi), \tag{2.11}$$

to indicate the whole parameter set of an HMM.

In the first–order hidden Markov model which we are using, there are two assumptions. The first is the Markov assumption for the Markov chain:

$$P(s_t \mid s_1, s_2, \ldots, s_{t-1}) = P(s_t \mid s_{t-1}), \tag{2.12}$$

and the second is the output independence assumption:

$$P(X_t \mid X_1, X_2, \ldots, X_{t-1}, s_1, s_2, \ldots, s_t) = P(X_t \mid s_t). \tag{2.13}$$

These assumptions might look as drastically simplifying the complex nature of the model. However, in practice, they make evaluation, decoding, and training feasible.

2.2.2 Evaluation Problem

Given the HMM model Φ and a sequence of observations $X = X_1, X_2, \ldots, X_T$, what is the probability $P(X \mid \Phi)$, i.e. what is the probability that the observations were generated by the model?

The intuitive way of computing $P(X \mid \Phi)$ is to first enumerate all possible state sequences S of the corresponding length T that generate observation sequence X, and then to sum all the probabilities. This is expressed formally as

$$P(X \mid \Phi) = \sum_{allS} P(S \mid \Phi) P(X \mid S, \Phi). \tag{2.14}$$

By applying the Markov assumption 2.12 for one particular state sequence $S = \{s_1, s_2, \ldots, s_T\}$, where s_1 is the initial state, we can express

$$P(S \mid \Phi) = \pi_{s_1} a_{s_1 s_2} \ldots a_{s_{T-1} s_T}. \tag{2.15}$$

Similarly by applying the output–independent assumption 2.13 we rewrite the term $P(X \mid S, \Phi)$ as

$$P(X \mid S, \Phi) = b_{s_1}(X_1) b_{s_2}(X_2) \ldots b_{s_T}(X_T). \tag{2.16}$$

By substituting equations 2.15 and 2.16 into 2.14 we get

$$P(X \mid \Phi) = \sum_{allS} P(S \mid \Phi)P(X \mid S, \Phi) = \sum_{allS} \pi_{s_1} b_{s_1}(X_1) a_{s_1 s_2} b_{s_2}(X_2) \dots a_{s_{T-1} s_T} b_{s_T}(X_T). \tag{2.17}$$

The naive evaluation of Eq. 2.17 requires enumeration of $O(N^T)$ possible state sequences. Fortunately there exists a much more efficient algorithm known as forward algorithm, which is capable of computing Eq. 2.17 in $O(N^2T)$. The detailed description of the algorithm can be found in [31] and [34].

2.2.3 Decoding Problem

The forward algorithm, as discussed in 2.2.2, computes the probability that an HMM generates an observation sequence by summing up the probabilities of all possible paths. The downside of this is that it does not provide the corresponding hidden state sequence. In many applications, as we will show in this thesis, it is desirable to find such a path. As a matter of fact, finding the best state sequence is the Holy Grail in searching in speech recognition. Mathematically speaking we are looking for the state sequence $S = \{s_1, s_2, \dots, s_T\}$ that maximizes $P(S, X \mid \Phi)$.

With this formulation we can use a formal technique based on dynamic programming, known as Viterbi algorithm [54]. The algorithm can be broken to three main parts. The first part is the initialization 2.18 of the accumulated probability V and the backtracking information B for each node in the first timeframe:

$$V_1(i) = \pi b_i(X_1), \qquad B_1(i) = 0, \qquad 1 \leq i \leq N \tag{2.18}$$

The second part of the algorithm is the induction step, where we update the accumulated scores V corresponding to the equation 2.19:

$$V_t(j) = \max_{1 \leq i \leq N} \Big[V_{t-1}(i) a_{ij} \Big] b_j(X_t), \qquad 2 \leq t \leq T \quad 1 \leq i, j \leq N \tag{2.19}$$

Similar update has to happen for the backtracking history according to equation 2.20

for every node:

$$B_t(j) = \arg \max_{1 \le i \le N} \left[V_{t-1}(i) a_{ij} \right], \qquad 2 \le t \le T \quad 1 \le i, j \le N \tag{2.20}$$

The last part of the algorithm is the termination. The best score and the corresponding backtracking information can be accessed as

$$BestScore = \max_{1 \le i \le N} [V_T(i)] \qquad BestPath_T = \arg \max_{1 \le i \le N} \left[B_T(i) \right] \tag{2.21}$$

The individual nodes forming the best hidden sequence in the backtracking history can be accessed as

$$BestPath_t = B_{t+1}(BestPath_{t+1}) \tag{2.22}$$

where $BestPath_1, BestPath_2, \ldots, BestPath_T$ are the nodes we were after.

This was the theory of the Viterbi search with complexity $O(N^2T)$. The full Viterbi search is unfortunately still unfeasible for the purpose of LVCSR as the amount of the HMM nodes in the speech recognizer is simply too big. Techniques such as pruning which allow faster computation are introduced in section 3.3.4. The implementation of the Viterbi search as used in the HMM tagger is discussed in chapter 5.

2.2.4 Training Problem

The last and, unfortunately, also the most difficult problem remaining is how to estimate the model parameters $\Phi = (A, B, \pi)$ given some training data. To train an HMM from M training data sequences is equivalent to finding the HMM parameter vector Φ that maximizes the joint probability

$$\prod_{i=1}^{M} P(X_i \mid \Phi). \tag{2.23}$$

In fact this is so hard that there is no known analytical method that maximizes the joint probability of the training data in a closed form.

Nevertheless, the situation is not yet critical as there is an iterative Baum-Welch algorithm [56], also known as the forward–backward algorithm. The good property of the algorithm is that it guarantees a monotonic likelihood improvement on each iteration. The unfortunate property is that it converges to a local (not global) maximum. More about the algorithm, together with the proof of convergence, can be found in [34].

2.2.5 Null Transitions

In practice, and that will be also our case later, it is convenient to introduce a null transition in some parts of the HMM network. It allows us to traverse the HMM without consuming any observation symbol X_i.

To incorporate the null transition (null arc) into the introduced framework we will need to modify the Viterbi algorithm, provided that no loops of empty transitions exist. If we denote the null transition between states i and j as a^{ϵ}_{ij}, the null transitions need to satisfy the modified constraints 2.8:

$$\sum_{j=1}^{N} a_{ij} + a^{\epsilon}_{ij} = 1 \qquad \forall i \tag{2.24}$$

The modification of the Viterbi induction step 2.19 will become

$$V_t(j) = \max\left[\max_{1\leq i\leq N}\left[V_{t-1}(i)a_{ij}\right]b_j(X_t)\ ;\ \max_{1\leq i\leq N}\left[V_t(i)a^{\epsilon}_{ij}\right]\right], \qquad 2 \leq t \leq T \quad 1 \leq i,j \leq N \tag{2.25}$$

Equation 2.25 appears to have an infinite recursion in it. In reality it uses the value of the same time $V_t(i)$, provided that i is already computed. This can be achieved (see [34]), as the empty transitions do not form a loop.

Chapter 3

Speech Recognition Engine

3.1 Overview

The speech recognition engine[1] which is discussed here is based on the statistical approach. The fundamental idea is the so–called noisy–channel model [8] as illustrated by figure 3-1. The speaker in this model consists of two parts. The source of the communication is in the speaker's mind. The speaker has to translate his or her thoughts into the word sequence W that will be pronounced by the vocal apparatus (speech producer).

The speech recognizer is also decomposed into two parts in this model. The acoustic waveform (speech) is processed by the acoustic processor. The output of the acoustic processor is a sequence of acoustic observations A. The purpose of the last component (linguistic decoder) is to find the most probable sequence of words $\hat{\mathrm{W}}$ given the input A. In the ideal case we will see the original sequence W on the output again.

[1] speech recognizer and speech recognition engine have both the same meaning in this thesis

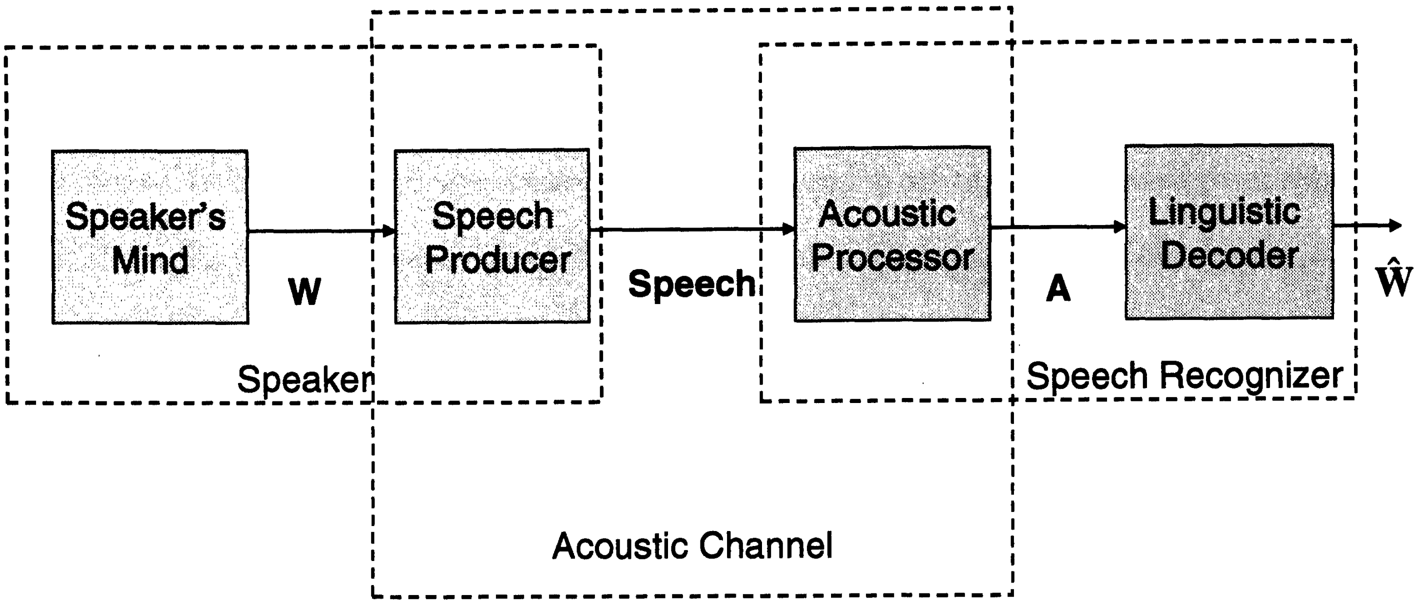

Figure 3-1: Noisy channel model of speech recognition as described in [34]

3.2 A Mathematical Formulation

Our approach is statistical, so probabilities will be used in the definition of the problem. Let

$$A = a_1, a_2, \ldots, a_T \tag{3.1}$$

be a sequence of acoustic symbols which correspond to the utterance spoken. The index of the individual symbol thus corresponds to the time. In a similar way let

$$W = w_1, w_2, \ldots, w_n \qquad w_i \in V \tag{3.2}$$

denote a sequence of n words where each word belongs to a vocabulary V. The term $P(W \mid A)$ denotes the probability that words W were spoken, given that the acoustic symbols A were observed. The task of the recognizer is to find the best sequence of words $\hat{W}$ that satisfy

$$\hat{W} = \arg\max_{W} P(W \mid A). \tag{3.3}$$

We shall note that by using this formula we accept the fact that one error is equally bad as many. This formula does not guarantee the best word error rate [23]. We will keep ignoring this fact for the rest of the thesis and rewrite the formula 3.3 by using Bayes formula as

$$\hat{W} = \arg\max_{W} P(W \mid A) = \arg\max_{W} \frac{P(W)P(A \mid W)}{P(A)}, \tag{3.4}$$

where $P(W)$ is the probability that the words W will be spoken. $P(A \mid W)$ is the probability that when the speaker says W, the acoustic symbols A will be observed. $P(A)$ is the probability that A will be observed. Since we have only one A (we are given one utterance only), we can ignore the term $P(A)$. For the sake of finding the best phrase $\hat{W}$ the formula 3.4 gets reduced to

$$\hat{W} = \arg\max_{W} P(W)P(A \mid W). \tag{3.5}$$

3.3 Components of a Speech Recognition Engine

Based on figure 3-1 and formula 3.5 we can divide the recognizer into four basic components.

- Acoustic processor (acoustic front end).
- Acoustic model that computes the term $P(A \mid W)$.
- Language model that computes the probability $P(W)$.
- The hypothesis search.

The language model is what we are after in this thesis and so language modeling will be discussed in detail in chapter 4. It still is desirable that the reader gets a brief description of the individual components of the recognition engine.

3.3.1 Acoustic Processor

As we can see on figure 3-1 the recognizer gets the input in the form of an acoustic waveform (this is what sound and speech is). Thus we need a front end capable of transforming the analog waveform into the digital symbols a_i. To achieve this, the

Acoustic Processor includes a microphone, a means of sampling the electrical output of the microphone and an algorithm for the acoustic features extractions.

It is known that satisfying speech recognition results are impossible to obtain without at least good acoustic processor, but the front end design belongs to the field of DSP (digital signal processing) and is beyond the scope of this thesis. Further reading on DSP can be found in [45] or [44].

3.3.2 Acoustic Model

The acoustic model is responsible for computing the probability $P(A \mid W)$. The amount of all possible pairs W and A is too large (the direct access to this precomputed value is unfeasible) so we need a stochastical model. We shall note that the whole process of modeling $P(A \mid W)$ takes into the account the way the speaker pronounces the different words W, the acoustic environment (such as background noise) and the acoustic processing as done by the acoustic processor. The acoustic models as used today are usually based on hidden Markov models (HMM). The alternatives such as dynamic time warping and neural networks are possible but were not used in the thesis.

In the speech recognition experiments which are discussed later in this thesis the hidden Markov model unit states correspond to a triphone (phone in the given left or right context). It is assumed that each acoustic observation a_i has been generated by such unit. To put things into the perspective, the acoustic observations a_i correspond to the observations X_i (to follow the notation in chapter 2.2). The acoustic observations generated by the acoustic processor are multidimensional, and so it remains to be clarified how to generalize the HMMs to the case where the output symbols X_i are substituted by the vectors of real numbers. In most LVCSR systems the output probability $b_j(\mathbf{x})$ is defined using multivariate Gaussian mixture density functions as

$$b_j(\mathbf{x}) = \sum_{k=1}^{M} c_{jk} N(\mathbf{x}, \mu_{jk}, \mathbf{\Sigma}_{jk}), \tag{3.6}$$

where $N(\mathbf{x}, \mu_{jk}, \mathbf{\Sigma}_{jk})$ denotes a single Gaussian density function with mean vector μ_{jk}

and covariance matrix $\mathbf{\Sigma}_{jk}$ for state j. M denotes the number of Gaussian mixture density functions, and c_{jk} is the weight of the k-th mixture component satisfying:

$$\sum_{k=1}^{M} c_{jk} = 1. \tag{3.7}$$

The single Gaussian density function is defined as

$$N(\mathbf{x}, \mu_{jk}, \mathbf{\Sigma}_{jk}) = \frac{1}{\sqrt{(2\pi)^n |\Sigma|}} e^{-\frac{1}{2}(\mathbf{x}-\mu)'\mathbf{\Sigma}^{-1}(\mathbf{x}-\mu)}. \tag{3.8}$$

where n is dimension of the observation vector $\mathbf{x}$.

3.3.3 Language Model

Language model (LM) is a probabilistic model which allows us to compute the probability $P(W)$ for any word sequence W as needed by the equation 3.5. In the formal language theory approach (e.g. [17]), the term $P(W)$ can be regarded as 1 or 0, depending on whether the word sequence W has been accepted by the grammar of the language or not. This binary behavior is not practical for us. One reason why we want always assign some nonzero probability is that the spoken language which we still want to recognize is often ungrammatical.

The main help of the language model to the recognizer is that it discriminates the unlikely word sequences. The language model shall be able to assign a higher probability to the word string *I have read Stendhal's Red and Black.* [51] then to the alternative word sequence with the same acoustic *I have red Stendhal's Read and Black.* This behavior will not only make the recognizer more accurate, it will also allow us to constrain the search space by ignoring the non–promising hypothesis.

Using the Bayes rule we can rewrite the term $P(W)$ as

$$\begin{aligned} P(W) &= P(w_1, w_2, \ldots, w_n) \\ &= P(w_1)P(w_2 \mid w_1)P(w_3 \mid w_1, w_2) \ldots P(w_n \mid w_1, w_2, \ldots, w_{n-1}) \\ &= \prod_{i=1}^{n} P(w_i \mid w_1, \ldots, w_{i-1}) \end{aligned} \tag{3.9}$$

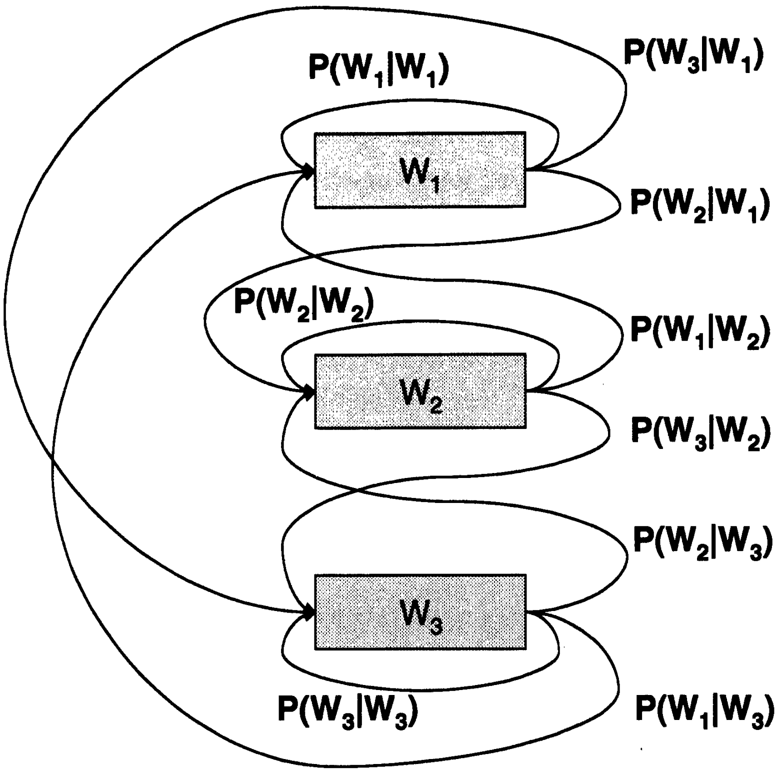

Figure 3-2: Bigram language model when the dictionary consists of three words only

where $P(w_i \mid w_1, \dots, w_{i-1})$ is the probability that w_i will follow, given that the words $w_1, \dots, w_{i-1}$ were spoken previously. We will continue in the detailed approaches to language modeling in chapter 4. A new proposed language model has been designed and tested for Czech language as described in section 4.5.

3.3.4 The Hypothesis Search

We know already, or at least we have the vague idea, how to compute probabilities $P(W)$ and $P(A \mid W)$ from the equation 3.5. We pointed out the connection of the acoustic model and HMMs in section 3.3.2. The chapter 4 and section 4.5 explain how to incorporate the discussed language models into the framework of HMMs. For the moment we can approximate the equation 3.9 as follows:

$$P(w_1, w_2, \dots, w_n) = \prod_{i=1}^{n} P(w_i \mid w_1, \dots, w_{i-1}) \approx \prod_{i=1}^{n} P(w_i \mid w_{i-1}). \tag{3.10}$$

This approximation is called the bigram model and we immediately see that this approximation satisfies the Markov assumption 2.12.

What follows is that we create a word level bigram network as shown on figure

3-2. Each word in this network can be decomposed to the corresponding phones based on its baseform[2]. Each phone is substituted with the corresponding triphone based on its context (ie, *k ae t*$\rightarrow$ $\langle$*silence-k+ae*$\rangle$ $\langle$*k-ae+t*$\rangle$ $\langle$*ae-t+silence*$\rangle$. The individual triphones can be further decomposed to some fixed number of HMM states (three states for each triphone in our case).

By doing this we have created a huge HMM. The transition probabilities A are both the bigram probabilities on the word boundaries and the probabilities learned from the forward-backward training 2.2.4 in the intra word context. The output probabilities B are computed by the acoustic model. We shall note that the figure 3-2 is a drastically simplified version of the full network used for the decoding (we did not complicate the diagram by introducing the corresponding initial and final states or by incorporating the optional silences between the words).

The Viterbi search as described in 2.2.3 will reveal a sequence of the individual HMM states forming the best scoring path. The states, we have found, represent the triphones. This means that we also know which phones and words W correspond to this best path. The often forgotten fact is that the words which we have found by performing the Viterbi search are not necessary the words maximizing the equation 3.5. To find the real maximizing sequence we need to sum up all the paths that correspond to the given word sequence. This is unfeasible in the world of LVCSR[3]. Fortunately for us, it is rare that the maximizing word sequence differs from the word sequence corresponding to the best path.

As we are usually interested in the word sequence only, we can modify the backtracking mechanism of the Viterbi search, so that only the word to word transitions will be stored in formula 2.20. The speed of Viterbi algorithm can be further improved by introducing pruning schemes. The usual approach is to modify the equation 2.19 so that we evaluate just the top Q candidates, where Q is fixed. A similar pruning scheme takes into the account just the states which are inside a given beam compared to the best scoring state.

[2] Each word in the vocabulary is presented with a phonetic baseform (ie, cat | k ae t).

[3] It is feasible for the isolated word recognition.

More sophisticated pruning methods, e.g. fast match in [41], can increase the search speed dramatically. We shall note that Viterbi search is not the only option. There are other algorithms, such as A^*, [36] being used in some recognizers [7].

3.4 Measuring the Quality of the Speech Recognizer

The purpose of this thesis is to introduce a new language model which will improve the quality of speech recognition for inflective languages, namely for Czech. We have to clearly state what we mean by the word *improve*. Our target is to improve the measure called accuracy (Acc) which is defined as

$$Acc = \frac{N - I - D - S}{N} \cdot 100\%, \tag{3.11}$$

where N is the total number of words in the correct sentence W. In the experiments we use hand annotated data for testing the speech recognizer, so we know the correct sequence W. The terms I, D, and S denote the numbers of insertions, deletions and substitutions respectively. To compute the numbers of I, D, and S we need to align a recognized word string $\hat{W}$ against the correct word string W. This alignment computation is also known as the maximum substring matching problem, which can be easily handled by the dynamic programming algorithm.

The alternative approach is to use a measure called word error rate (WER) defined as

$$WER = 100 - Acc. \tag{3.12}$$

In some special application we might not consider insertions as errors and this measure is in that case called correctness. For the purpose of dictation, however, insertions are as bad as deletions.

The final measure worth mentioning is the sentence error rate (SER), which indicates the percentage of sentences whose transcriptions have not matched in an exact manner those of reference.

Let us investigate a real output of the speech recognizer compared to the correct script of the utterance following the notion introduced in section 3.2.

$\hat{W}$: ZÁPLAVY POSTIHLY DESÍTKY OBCÍ OPĚT VĚTŠÍCH MĚST

W: ZÁPLAVY POSTIHLY DESÍTKY OBCÍ A PĚT VĚTŠÍCH MĚST[4]

We can see that the recognizer made an error by substituting words "OPĚT"[5] and "PĚT"[6] and by deleting the word "A"[7]. The accuracy of this recognized utterance is thus $Acc = \frac{8-0-1-1}{8} \cdot 100 = 75\%$

The accuracy is an accepted measure in most of the speech recognition contests, but it can be sometimes wrong to measure the quality of the speech recognition in terms of accuracy only. There are speech enabled systems (such as a bank account voice control), where sacrificing few percent points of the accuracy can be accepted if the confidence of the recognized utterance is not high enough. It is the quality of the rejection mechanism [25] which has a high importance in this case. The various methods of the confidence score are described in [49].

[4] Floods have stroked tens of villages and five larger cities.

[5] again

[6] five

[7] and

Chapter 4

Language Modeling

4.1 Introduction to Language Modeling

We have shown that the language model is an important part of the speech recognizer in chapter 3.3. It is important to note that in some of the speech recognition applications the language model is not needed at all [10]. These are mainly the "command and control" types of applications where the equivalent of a language model is a grammar written in some formalism as for example "Java Speech Grammar Format" (JSGF)[5]. In case that we are interested in LVCSR systems we simply cannot ignore the language model.

Let us demonstrate the effect of the language model on the following utterance:

W: NEBEZPEČNÝ POŽÁR VLAKU S EXPLOZIVNÍM PROPANEM NADÁLE ZUŘÍ NA SEVERU NORSKA[1]

This sentence was decoded by using the language model presented in section 4.5 as

$\hat{W}$: NEBEZPEČNÝ POŽÁR S EXKLUZÍVNÍM PANEM NADÁLE ZUŘÍ NA SEVERU NORSKA

Finally the same sentence with no language model at all.

$\hat{W}$: NEBEZPEČNÝ V UŽ VLAK ÚST EXPLOZÍ NÍM TR O PANEM NADÁLE ZUŘÍ NA SEVERU NORSKÁ D

[1] In northern Norway a train containing explosive propane gas continues to burn.

By carefully examining the output of the recognizer when no language model has been used we can see that the recognizer has a strong tendency to model the acoustic evidence with short words (*TR O PANEM* versus *PROPANEM*). Using the language model has led to much better result, but we have to pay for the fact that we use a limited vocabulary and thus we did not have some words (*EXPLOZIVNÍM*) in the vocabulary. Examples of the recognized utterances are included in the attachment B.

4.2 Language Modeling and Natural Language

As we have covered in section 3.3.3, the language model probability $P(W)$ can be decomposed as

$$\begin{aligned} P(W) \quad &= P(w_1, w_2, \ldots, w_n) \\ &= P(w_1)P(w_2 \mid w_1)P(w_3 \mid w_1, w_2) \ldots P(w_n \mid w_1, w_2, \ldots, w_{n-1}) \\ &= \prod_{i=1}^{n} P(w_i \mid w_1, \ldots, w_{i-1}) \end{aligned} \tag{4.1}$$

In the formula 4.1 we accept the fact the choice of w_i depends only on the history $w_1, \ldots, w_{i-1}$. The models based on the formula 4.1, where the length of history is fixed, can be modeled by means of the Markov models. This can be achieved by grouping the word histories into equivalence classes so that the probability of a given word depends on the preceding state (class) only. The practical demonstration of this approach is shown in section 5.3. But is this really how we speak? Do we really create a sentence by producing the first word and based on this word we continue to the second word and so on?

Noam Chomsky shows in [17] that English is not a finite state language. This is his argument: because English contains constructions that are not regular then English is not regular. This is demonstrated on the reversal language defined as ww^R (where w consists of a or b)[2].

As stated, the argument is fallacious, as we can consider the regular language

[2] This language generates sentences *aa*, *bb*, *abba*, *baab*, *aaaa*, *bbbb*, *aabbaa*, *abbbba*, ...

$(a \mid b)^*$. This language contains the language ww^R (in the meaning that all sentences generated by ww^R can be generated also by $(a \mid b)^*$.

However, the proof that English[3] is not a finite state language can be done correctly [50], and so we have to accept the fact that the model

$$\begin{aligned} P(W) &= P(w_1, w_2, \ldots, w_N) \\ &\approx \prod_{i=1}^{N} P(w_i \mid w_{i-n+1}, \ldots, w_{i-1}) \end{aligned} \tag{4.2}$$

with fixed history of length n is not good enough for modeling the language properly. We will ignore this fact and use a variant of approximation 4.2, as any means which lead to a better accuracy of the speech recognizer are acceptable for us.

The formula is not the only one used in the world of language modeling. The most common alternative approach is the use of probabilistic context–free grammars (PCFG). In the PCFG, we have to address the similar problems as we did for HMMs in chapter 2. The probability of the sequence $W = w_1, w_2, \ldots, w_n$ in PCFG is computed as $P(S \Rightarrow W \mid G)$, where S is the starting symbol of the grammar and G the probabilistic grammar. The sign $\Rightarrow$ denotes the derivation sequence of one or more steps using the rules of grammar G.

The problem of using PCFG [31] is that the rules of the grammar and the search for the best derivation are hard to incorporate into the recognizer. The best state–of–the–art result of using the PCFG in a speech recognizer (structured language model) is described in [15]. The structured language modeling (SLM) shows a full 1.0% absolute improvement (13.7% to 12.7%) in WER [16] over the baseline trigram model for the WSJ DARPA93 HUB1 test setup.

4.3 The Traditional N-Gram Approach

The n-gram approach is based on formula 4.2. The value of n in this formula denotes the order of the n-gram. We use the terms unigrams, bigrams and trigrams for values

[3] The same can be done for Czech.

N-gram model	Parameters to be estimated
unigram	62000
bigram	3.844×10^9
trigram	2.383×10^{14}
fourgram	1.478×10^{19}

Table 4.1: The amount of n-grams needed for a vocabulary of 62,000 words

$n = 1$, $n = 2$ and $n = 3$ respectively. The individual probabilities of n-grams will be estimated by using the relative frequencies in the training data as follows:

$$P(w_i \mid w_{i-n+1}, \ldots, w_{i-1}) = \frac{N(w_{i-n+1}, \ldots, w_{i-1}, w_i)}{N(w_{i-n+1}, \ldots, w_{i-1})} \tag{4.3}$$

where N denotes the count of the corresponding word n-tuples in the training data. The bigram and trigram models are the most common language models to be found in the LVCSR systems today.

One of the first decision we have to do in the n-gram language modeling is the choice of the corresponding n. We use a fixed vocabulary of words in the speech recognizer and so we can examine how many parameters each of the n-gram models needs to estimate (see table 4.1). The amount of bigrams is huge already, and there is no chance at all that we will have enough data to see all the possible trigrams. In fact the vast majority of trigrams will never occur in the language as it forms absolutely ungrammatical constructions. The fact that there will be plenty of unseen n-grams is called "data sparseness problem".

4.3.1 Data Sparseness Problem

Data sparseness poses problem for nearly all statistical methods. We have illustrated in the table 4.1 that the amount of the traning data will be never sufficient for the trigram model[4]. There was an experiment performed in 1970 at IBM Research, where a corpora of patent descriptions has been divided into a training and test data. It was discovered that 23% of the trigrams appearing in the test set never occurred in

[4] "There is no data like more data" (Bob Mercer at Arden House, 1985)

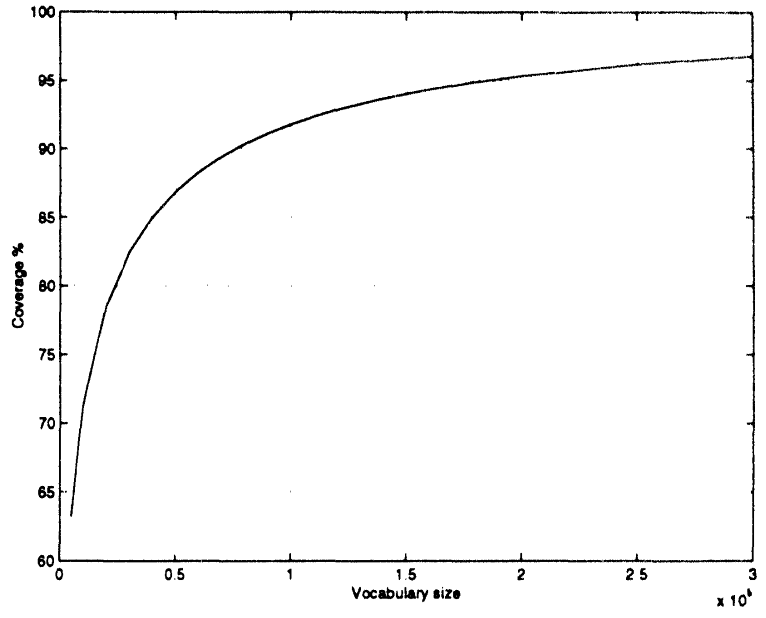

Figure 4-1: Vocabulary self-coverage for the Czech National Corpus

the training. This means that a speech recognizer operating according to formula 4.3 will have a guaranteed error rate at least 23%.

It is thus necessary to introduce some mechanism which will "smooth" the trigram frequencies. We will show two ways how to smooth the language model. For the n-gram based model in the speech recognizer we have used the Katz backoff model in section 4.4. The HMM tagger uses a linear interpolation smoothing method described in section 5.3.2.

4.3.2 Data Sparseness and Czech

The problem of data sparseness is common across all languages as the numbers in table 4.1 depend on the size of vocabulary only. It is the size of the vocabulary needed which can make the data sparseness even more troublesome. Let us examine figure 4-1, which shows the vocabulary coverage for the Czech National Corpus. The striking observation is that for a vocabulary of 60,000 words we get coverage of only 88.3%. This means that the recognizer for the unrestricted domain (the Czech National Corpus is a balanced representative corpus of contemporary written Czech) will have WER at least 11.7%. The same size of vocabulary for British English (British National Corpus [12]), will guarantee us the coverage of nearly 99% [55] [32].

The low coverage we observe for the Czech dictionary is due to the fact that Czech is richly inflected language. In the case of nouns the corresponding case, number and gender are usually distinguished by a different ending. For verbs we distinguish three persons in both singular and plural. The form of the verb is again usually distinguished by a different ending. The quick guide to Czech morphology [27] is described by positional tags (see table 5.2 and appendix A.1).

The data sparseness in Czech is not caused by the rich inflectional nature of the language only. The other phenomenon is the free word order. There is no strict order (as for English) for constituents such as subject, object, possessor, etc. The aspect which makes the language with a free word order understandable is the use of agreement (noun and it's adjectival attribute must agree in gender, case and number for example). We will show later in our experiments, that by using a language model which includes the morphological information, we gain in accuracy. Our idea is that by enriching the language model by the morphological tags (see chapter 5) we will be able to solve two problems at once.

1. There are not so many tags as words in the Czech morphology. This will make easier to collect reliable statistics.

2. Let us consider two different words occurring in the test data in different contexts: "thajskou"[5] and "československou"[6]. These two adjectives share the same morphological category, which can be specified by the tag `AAFS7----1A----`. To translate this tag into a human readable form; the both words are adjectives, feminine, singular and the case is instrumental. Now the training data for the language model stand from the Czech Republic and thus contain many occurrences of the word "československou". That is not the case for the word "thajskou" (Czech newspapers do not tend to write about Thailand so often). By using the enriched language model, we will be able to share at least the fact that for these two words we expect a similar morphological context.

[5] Thai

[6] Czechoslovakian

4.4 Good-Turing Estimate and Katz Backoff Model

We will discuss the Katz smoothing model in this section since it was used as the smoothing mechanism for the word based n-gram language model in our experiments. An alternative smoothing method has been used for the HMM tagger and is described in section 5.3.2.

4.4.1 Good-Turing Estimate

The Good-Turing estimate is a smoothing technique to deal with infrequent n-grams. It is usually not used by itself for the task of n-gram smoothing because it does not include the combination of higher order models with lower order models (as necessary for good performance). However, it is used as the main mechanism in several more complex smoothing techniques.

The Good-Turing estimate states that for any n-gram that occurs r times, we should pretend that it occurs r^* times as follows:

$$r^* = (r+1)\frac{n_{r+1}}{n_r} \tag{4.4}$$

where n_r denotes the number of n-grams that occur exactly r times in the training data. However we are interested in a probability and thus we need to convert the r^* count to a probability. We normalize for an n-gram g, that occurs r times

$$P(g) = \frac{r^*}{N} \tag{4.5}$$

where $N = \sum_{r=0}^{\infty} n_r r^*$ is equal to the number of samples in the training data [24] (as we can rewrite $N = \sum_{r=0}^{\infty} n_r r^* = \sum_{r=0}^{\infty}(r+1)n_{r+1} = \sum_{r=0}^{\infty} n_r r$). The formula 4.5 is usually referred to as Good-Turing probability estimate.

4.4.2 Katz Smoothing

Katz smoothing further extends the idea of Good-Turing estimate 4.5 by incorporating the combination of higher-order models with lower-order models. We will demonstrate the principal features of the Katz smoothing on a bigram example. Katz smoothing [35] is using the Good-Turing estimate for nonzero counts in the following way:

$$C^*(w_{i-1}, w_i) = \begin{cases} d_r r & \text{if } r > 0 \\ \alpha(w_{i-1}) P(w_i) & \text{if } r = 0 \end{cases} \tag{4.6}$$

where d_r is the discount ratio. We discount according to the ratio for $r > 0$ and that leaves some quantity (the counts subtracted from the nonzero counts) to be distributed among the zero-count bigrams according to the next lower-order distribution (unigrams in our case). The value of $\alpha(w_{i-1})$ compensates the total amount of counts in the data and its value is computed so that the smoothed bigram satisfies the probability constraint:

$$\alpha(w_{i-1}) = \frac{1 - \sum_{w_i:C(w_{i-1},w_i)>0} P^*(w_i|w_{i-1})}{\sum_{w_i:C(w_{i-1},w_i)=0} P(w_i)} = \frac{1 - \sum_{w_i:C(w_{i-1},w_i)>0} P^*(w_i|w_{i-1})}{1 - \sum_{w_i:C(w_{i-1},w_i)>0} P(w_i)} \tag{4.7}$$

where the $P^*(w_i|w_{i-1})$ is computed from the altered counts, that is:

$$P^*(w_i|w_{i-1}) = \frac{C^*(w_{i-1}, w_i)}{\sum_{w_x} C^*(w_{i-1}, w_x)} \tag{4.8}$$

In the implementation as proposed by Katz in [35] $d_r = 1$ for reliable counts $r > k$, where k is a constant chosen empirically. The discount ratios for the less reliable counts ($r \leq k$) are derived from the Good-Turing estimate so that:

1. The discounts coefficients d_r are proportional to the discount coefficients defined by the Good-Turing estimate.

2. The total number of counts discounted in the global bigram distribution is equal to the total number of counts that should be assigned to bigrams with zero counts according to the Good-Turing estimate.

These two constrains can be formally written as

$$d_r = \mu \frac{r^*}{r}, \tag{4.9}$$

where $r \in \{1 \dots k\}$ and μ is a constant. The Good-Turing estimate predicts that the total mass assigned to bigrams with zero counts is $n_0 \frac{n_1}{n_0} = n_1$, and thus the second constraint corresponds to:

$$\sum_{r=1}^{k} n_r(1 - d_r)r = n_1. \qquad (4.10)$$

The unique solution is given by

$$d_r = \frac{\frac{r^*}{r} - \frac{(k+1)n_{k+1}}{n_1}}{1 - \frac{(k+1)n_{k+1}}{n_1}}. \qquad (4.11)$$

Katz smoothing for the higher-order n-gram models is defined in a similar way. The Katz n-gram model of order n is defined by the means of the $(n-1)$ gram model. The recursive procedure ends up with unigram model which is the maximum likelihood model. To sum the whole procedure up we have

$$P_{Katz(w_{i-1},w_i)} = \begin{cases} \frac{C(w_{i-1},w_i)}{C(w_{i-1})} & \text{if } r > k \\ d_r \frac{C(w_{i-1},w_i)}{C(w_{i-1})} & \text{if } k \geq r > 0 \\ \alpha(w_{i-1})P(w_i) & \text{if } r = 0 \end{cases} \qquad (4.12)$$

where d_r is defined by formula 4.11 and $\alpha(w_{i-1}) = \frac{1 - \sum_{w_i:r>0} P_{Katz}(w_i|w_{i-1})}{1 - \sum_{w_i:r>0} P(w_i)}$.

4.5 Combination of a Hidden Tag Model and a Traditional N-Gram Model

To the best of our knowledge, the first introduction of the tagger as a speech recognition language model component was in [11] without improving results over the baseline bigram model. The idea has been further explored in [34] where the author proposes the interpolation with a trigram model based on formula 4.13.

$$P(W) = \lambda P(w_i|w_{i-2}, w_{i-1}) + (1 - \lambda)Q(w_i|g(w_{i-2}), g(w_{i-1})), \qquad (4.13)$$

where $g(w_i)$ is the tagging function. The importance of formula (4.13) for languages with the data sparseness problem is that the new component Q can have enough evidence to give us reliable statistics about the word sequence W as the size of the tag set tends to be much smaller then the size of the word vocabulary itself.

The problem with approach (4.13) is that the tagging function $g(w_i)$ depends on all words of the utterance (supposing that the tagging component is performed by an HMM tagger as described in chapter 5). The standard solution is to replace the probability Q by a new probability Q^*:

$$Q^*(w_i|w_1,\ldots,w_{i-1}) \doteq \sum_{t_1,t_2} Q(w_i|t_1,t_2)T\Big(g(w_{i-2}) = t_2, g(w_{i-1}) = t_1 \mid w_1,\ldots,w_{i-1}\Big), \tag{4.14}$$

where $T\Big(g(w_{i-2}) = t_2, g(w_{i-1}) = t_1 \mid w_1,\ldots,w_{i-1}\Big)$ is the corresponding forward probability of the HMM with states corresponding to pairs of tags $\langle t_1, t_2\rangle$ based on a transition probability $P_t(t_i \mid t_{i-2}, t_{i-1})$ and the output probability $P_o(w_i \mid t_{i-1}, t_i)$. The probabilities of the HMM tagger are discussed in detail in section 5.3, the forward algorithm is described in section 2.2.2. We did not use this proposed solution as the computation of the Q^* during the decoding tends to be unfortunate due to several practical reasons:

- The whole computation of P(W) in the n-gram model is done in the logarithmic domain (so that we do not have to deal with the underflow problems when multiplying the probabilities). The logarithmic domain allows us not to use the multiplication in the decoder at all. The forward algorithm requires adding the probabilities and thus we have to simulate the required sum in the logarithmic domain.

- In our setup (see [37]) we are using the morphological analyzer [27] so that for every input word only the list of possible tags is considered. The nature of how Q^* gets computed will thus prefer hypothesis where the list of the plausible tags is bigger. We do not believe that this is the desired behavior and this might be one of the reasons why using the formula 4.14 did not bring any major success.

Our solution is based on maximizing the probability of the sequence of tag-word pairs instead of maximizing the sequence of words only. One of the consequences of this approach is that we do not use the forward probabilities at all. We can compute the best sequence by using the Viterbi algorithm describeo in section 2.2.3. It is important to note that having both the n-gram model and the tag model Viterbi based, the decoding step gets much straightforward then compared to formula 4.13.

By using this proposed approach the new $Q(W,T)$ function becomes:

$$Q^{**}(w_i, t_i|w_1, \ldots, w_{i-1}) = P_t(t_i|t_{i-1}, t_{i-2})P_o(w_i|t_i, t_{i-1}), \qquad (4.15)$$

where P_t and P_o are the corresponding transition and output probabilities distributions (see formulas 5.14 and 5.15) of the HMM-tagger, and t_i, t_{i-1}, t_{i-2} are the tags corresponding to words w_i, w_{i-1}, w_{i-2}. Instead of using the formula 4.13 we introduce the cost function

$$Cost(W) = log\Big(P(w_i|w_{i-2}, w_{i-1})\Big) + f_{tag}log\Big(Q^{**}(w_i, t_i|w_1, \ldots, w_{i-1})\Big), \qquad (4.16)$$

where the scaling factor f_{tag} needs to be optimized for the best accuracy as shown in section 6.5.

One advantage of the approach 4.16 is that for each decoded utterance we get both the words and the corresponding morphological category for each word. Let us demonstrate the effect of the proposed approach on the following utterance:

W: TÍM PADL PRVNÍ ITALSKÝ KABINET. [7]

This utterance has been incorrectly recognized by the traditional n-gram model as

$\hat{W}$: TÍM PADLA [8] PRVNÍ ITALSKÝ [9] KABINET. [10]

The acoustic difference between the two candidate words *padla* and *padl* is very small in the given context. Thus it must be the language model which will play the main

[7] As a consequence of this the first Italian government has collapsed.

[8] collapsed / verb feminine

[9] Italian / adjective masculine

[10] government / noun masculine

role in recognizing the word *padl* in this example. By using the standard trigram model we can quantify the difference between the two hypotheses as

$$P(\text{PADLA} \mid \langle \text{s} \rangle,\ \text{TÍM}\ )P(\text{PRVNÍ} \mid \text{TÍM}, \text{PADLA})P(\text{ITALSKÝ} \mid \text{PADLA}, \text{PRVNÍ}) \tag{4.17}$$

versus

$$P(\text{PADL} \mid \langle \text{s} \rangle,\ \text{TÍM}\ )P(\text{PRVNÍ} \mid \text{TÍM}, \text{PADL})P(\text{ITALSKÝ} \mid \text{PADL}, \text{PRVNÍ}) \tag{4.18}$$

The only "suspicious" trigram in the incorrect hypothesis has the probability $P($ITALSKÝ PADLA, PRVNÍ$)$. This trigram has been unseen in the training data. That was also the case for the alternative trigram $P($ITALSKÝ $\mid$ PADL, PRVNÍ$)$. The language model has thus used the backoff bigram probabilities in both cases. The unfortunate outcome is that the training data had more evidence for the wrong alternative 4.17.

The language model based on 4.16 recognized the utterance correctly and assigned the following morphological categories (see table 5.2 and appendix A.1).

(*TÍM*, `PDZS7----------`) (*PADL*, `VpYS---XR-AA--1`) (*PRVNÍ*, `CrIS1----------`)
(*ITALSKÝ*, `AAIS1----1A----`) (*KABINET*, `NNIS1-----A----`).

The correct recognition has occurred due to the fact that we use probability Q^{**}. By using it we get a low probability for the above mentioned "suspicious" trigram

$$\begin{aligned} &Q^{**}((\text{ITALSKÝ}, \texttt{AAIS1----1A----}) \mid \text{PADLA}, \text{PRVNÍ}) = \\ &= P_t(\texttt{AAIS1----1A----} \mid \texttt{VpQW---XR-AA--1}, \texttt{CrIS1----------}) \times \\ &\times P_o(\text{ITALSKÝ} \mid \texttt{CrIS1----------}, \texttt{AAIS1----1A----}) \end{aligned} \tag{4.19}$$

but we get a higher probability for the correct alternative

$$\begin{aligned} &Q^{**}((\text{ITALSKÝ}, \texttt{AAIS1----1A----}) \mid \text{PADL}, \text{PRVNÍ}) = \\ &= P_t(\texttt{AAIS1----1A----} \mid \texttt{VpYS---XR-AA--1}, \texttt{CrIS1----------}) \times \\ &\times P_o(\text{ITALSKÝ} \mid \texttt{CrIS1----------}, \texttt{AAIS1----1A----}) \end{aligned} \tag{4.20}$$

The critical information helping us in this case is the probability P_t. We have the evidence of seeing verb masculine followed by numeral masculine inanimate and followed by adjective masculine inanimate in the training data. On the other side we have no evidence at all of seeing verb *feminine* followed by numeral masculine inanimate and followed by adjective masculine inanimate.

Chapter 5

HMM tagger

5.1 Introduction

The task of a tagger is to assign part of speech (POS) tags to words reflecting their morphological category. But often, words can belong to different morphological categories in different contexts. For instance, the word form "obchod"[1] can have two readings: in the sentence "Náš soused otevřel nový obchod."[2] word form "obchod" is a noun in a fourth case singular, but in the sentence "Ten obchod má již zavřeno."[3] it is a noun in first case singular.

A POS-tagger should determine all possible readings for all the words, and assign the right reading given the context. It will be our goal to design and implement a tagger which will be suitable for the use in speech recognition in this chapter.

5.2 The Problem of Tagging

Let us suppose that the language (Czech in our case) has defined a set of tags attached to word forms. Let us a have a sentence W

$$W = w_1, w_2, \ldots, w_n \qquad w_i \in V_W \tag{5.1}$$

[1] shop

[2] Our neighbor has opened a new shop.

[3] The shop is closed already.

and a sequence of tags T of the same length

$$T = t_1, t_2, \ldots, t_n \qquad t_i \in V_T, \tag{5.2}$$

where V_W and V_T are vocabularies of all word forms and tags respectively. We will call the pair (W, T) an alignment. The word w_i has been assigned the tag t_i in this alignment.

We assume that the tags have some linguistic meaning in the language, so that among all the possible alignments for the sentence W there is one correct from the grammatical point of view. This assumption is needed for the training and the evaluation of the tagger. As we can see in section 4.5 the tagger will be perfectly able to find some alignment even when the correct alignment does not exist[4]. A tagging function is a function g

$$g : W \to T = g(W), \tag{5.3}$$

that selects a sequence of tags given a word sequence W, and thus it also defines the alignment.

The standard measure used to compare taggers is accuracy at word level, telling us percentage of words correctly tagged. To make a proper judgment about the quality of the given tagger we must use the same tag set V_T for all the taggers we compare.

5.3 Probabilistic Formulation

It is our intention to use the tagger as a component for a statistical language model for a speech recognizer. It is thus natural that the method of computing the alignment (W, T) will be presented in the framework of HMMs [9] [18] as introduced in chapter 2.2. Other methods used for part of speech tagging are described in [30].

The HMM tagger is based on the model of text production. We pretend that people do not think in words, instead they think in the morphological classes (tags). The written text W we see is hiding the original sequence T which is what the author

[4] N-gram models do not guarantee grammatical sentences.

had originally on his or her mind.

This idea which stays behind the HMM tagger is somehow fantastic and unrealistic, but it is derived from the noisy channel approach as used in speech recognition, where it is known to work. We will demonstrate that it works for the tagging problem as well.

We will start with a similar equation as in the problem of speech recognition as described in 3.3.

$$\hat{T} = \arg \max_{T} P(T \mid W) \tag{5.4}$$

The term $\hat{T}$ denotes the recognized tag sequence. We will rewrite formula 5.4

$$\hat{T} = \arg \max_{T} P(T \mid W) = \arg \max_{T} \frac{P(T)P(W \mid T)}{P(W)} \tag{5.5}$$

The word sequence W is only one and we can thus ignore the $P(W)$. We end up with the well known formula of HMM tagging.

$$\hat{T} = \arg \max_{T} P(T)P(W \mid T). \tag{5.6}$$

The problems which are left is how to estimate probabilities $P(T)$ and $P(W \mid T)$ and how to find the tagging function $g(w)$.

5.3.1 Maximum Likelihood Training

We have shown in chapter 2.2 that there is an algorithm which will estimate $P(T)$ and $P(W \mid T)$ by requiring no hand-tagged text. This has been tried with different initialization schemes [38] of the forward backward algorithm. Unfortunately, regardless how much data was used, the automatic determination of the model parameter vector Φ always led to worse tagging results then the approach we describe in this section.

We will describe a method which will estimate the probabilities $P(T)$ and $P(W \mid T)$. The HMM tagger, we are designing will use the following formulas

$$\begin{aligned} P(T) &= P(t_1, t_2, \dots, t_n) \\ &= P(t_1)P(t_2 \mid t_1)P(t_3 \mid t_1, t_2) \dots P(t_n \mid t_1, t_2, \dots, t_{n-1}) \\ &= \prod_{i=1}^{n} P(t_i \mid t_1, \dots, t_{i-1}) \approx \prod_{i=1}^{n} P(t_i \mid t_{i-1}, t_{i-2}) \end{aligned} \tag{5.7}$$

and by using the same approximation for the probability $P(W|T)$ we get

$$P(W|T) \approx \prod_{i=1}^{n} P(w_i \mid t_i, t_{i-1}). \tag{5.8}$$

First we need to fit these probabilities into the HMM theory as described in chapter 2.2. The output observation alphabet O corresponds to the dictionary of all words V_W. The observed sequence $X = X_1, X_2, \dots, X_n$ corresponds to the words $W = W_1, W_2, \dots, W_n$.

The hidden states have to correspond to the individual tags t_i, but at the same time we have to follow the assumptions 2.12 and 2.13. This correspondence can be achieved by defining the state space Ω as

$$\Omega = \{1, 2, \dots, V_T^2\}, \tag{5.9}$$

where each state corresponds to some pair of tags $\langle t_x, t_y \rangle$. The correspondace is thus achieved as a_{ij} is the probability of taking a transition from state i to state j, i.e.,

$$a_{ij} = P\Big(s_{time} = j = \langle t_x, t_y \rangle \mid s_{time-1} = i = \langle t_y, t_z \rangle\Big) = P(t_x \mid t_y, t_z) \tag{5.10}$$

The same approach is used for expressing the $b_i(k)$ output probability of emitting symbol w_k when state i is entered.

$$b_i(k) = P\Big(w_k \mid s_{time} = i = \langle t_x, t_y \rangle\Big) = P(w_i \mid t_x, t_y) \tag{5.11}$$

Maximum likelihood estimation (MLE) is a straightforward training method if we have some large corpora of tagged text available. The tagged text comes from the human annotator who hand-crafted the correct alignment (W, T). According to the MLE, the probability $P(t_i \mid t_{i-1}, t_{i-2})$ is computed as

$$P(t_i \mid t_{i-1}, t_{i-2}) = \frac{N(t_{i-2}, t_{i-1}, t_i)}{N(t_{i-2}, t_{i-1})} \tag{5.12}$$

and a similar estimate for the probability $P(w_i \mid t_i, t_{i-1})$

$$P(w_i \mid t_i, t_{i-1}) = \frac{N(t_{i-1}, t_i, w_i)}{N(t_{i-1}, t_i)} \tag{5.13}$$

where the term $N(t_{i-2}, t_{i-1}, t_i)$ states how many times the sequence t_{i-2}, t_{i-1}, t_i has appeared in the alignment and, similarly, $N(t_{i-1}, t_i, w_i)$ means how many times the word w_i appears with a tag t_i and the word w_{i-1} with tag t_{i-1} in the alignment.

We can immediately see the main drawback of the MLE approach. Estimates 5.12 and 5.13 will assign a probability of zero to any sequence tags t_x, t_y, t_z that did not occur in the training data. The same problem occurs for unseen words. We can have several hundreds of tags in the tag set, so the chance to see all the possible combination is close to zero. What is even worse, we cannot even hope that we will see all the words from our dictionary V_W in the training.

We have to introduce a method which will allow us some work around of the unseen events, otherwise the Viterbi search algorithm 2.2.3, which we hope to use, will never find an alignment with probability better than zero for some sentences.

5.3.2 Linear Smoothing

We have introduced the problem of unseen events in the training data. Linear smoothing is a simple and effective answer to this problem. Instead of using the relative frequencies from the MLE estimates 5.12 and 5.13 directly, we define a smoothed probability $P(t_i \mid t_{i-1}, t_{i-2})$

$$\begin{aligned} P(t_i \mid t_{i-1}, t_{i-2})_{smooth} &= \lambda_3 P_{MLE}(t_i \mid t_{i-1}, t_{i-2}) + \lambda_2 P_{MLE}(t_i \mid t_{i-1}) + \\ &+ \lambda_1 P_{MLE}(t_i) + \lambda_0 \frac{1}{|V_T|} \end{aligned} \tag{5.14}$$

where $\sum_{i=0}^{3} \lambda_i = 1$ and $0 < \lambda_j \leq 1$. For the probability $P(W|T)$ we define the smoothed alternative as

$$P(w_i \mid t_i, t_{i-1})_{smooth} = \gamma_3 P_{MLE}(w_i \mid t_i, t_{i-1}) + \gamma_2 P_{MLE}(w_i \mid t_i) + \gamma_0 \frac{1}{|V_W|} \tag{5.15}$$

where $\gamma_0 + \gamma_2 + \gamma_3 = 1$ and $0 < \gamma_j \leq 1$.

The unigram term $\gamma_1 P_{MLE}(w_i)$ can be omitted from formula 5.15 as the output probability for the HMM depends on the current state. When word w_i happens to be unseen in the training, term $\gamma_0 \frac{1}{|V_W|}$ will not allow the zero probability problem to occur in the test data.

Our goal is to find the optimum parameters λ_j and γ_j. To achieve that, we will not use the whole hand annotated data for computing probabilities P_{MLE}. The part which has not been used is called held-out data and will serve us to find the set of smoothing coefficients λ and γ which maximize the probability of emitting the held-out data by our interpolate model.

We will demonstrate two different approaches how to find the optimum smoothing coefficients for the set of λ_i.

We have shown that the term $P(t_i \mid t_{i-1}, t_{i-2})$ can be regarded as an HMM. We will use this fact by incorporating the λ coefficients to it as shown on figure 5-1.

By examining the figure we see four null transitions (see 2.2.5) outgoing from the leftmost state $s_\lambda(t_1, t_2)$ into states $s_0(t_1, t_2)$, $s_1(t_1, t_2)$, $s_2(t_1, t_2)$ and $s_3(t_1, t_2)$. These transitions are taken with probabilities λ_0, λ_1, λ_2 and λ_3 respectively. Out of each of the four states lead $|V_T|$ transitions. By taking any of these transitions and entering the corresponding state $s_\lambda(t_2, tg_i)$ we observe tg_i. The transitions from states $s_0(t_1, t_2)$, $s_1(t_1, t_2)$, $s_2(t_1, t_2)$ and $s_3(t_1, t_2)$ are taken with probabilities $\frac{1}{|V_T|}$, $P_{MLE}(tg_i)$, $P_{MLE}(tg_i \mid t_2)$ and $P_{MLE}(tg_i \mid t_1, t_2)$, respectively.

The figure 5-1 shows us just a small part of the overall HMM network. We must realize that the P_{MLE} probabilities are known and the only thing we need to estimate are the λ_i probabilities. The other simplifying fact is that due to the formula 5.14 the λ_i coefficients will have the same value regardless of the actual combination t_1, t_2, tg_i.

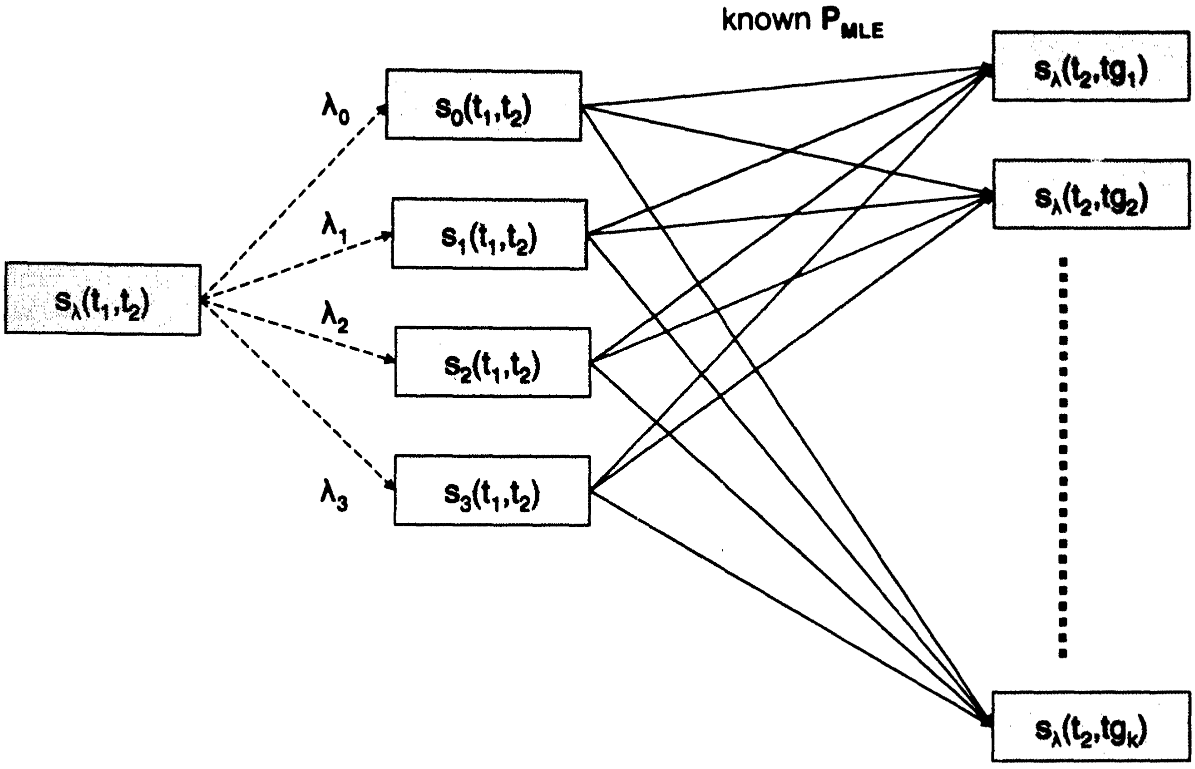

Figure 5-1: Linear smoothing section of an HMM corresponding to transition probabilities in the tagger

The forward backward algorithm from section 2.2.4 is thus feasible for computing the smoothing parameters.

The alternative way to compute the set of λ_i parameters has been introduced in [48] and this is also the algorithm which the implementation of the HMM tagger uses for the experiments in this thesis.

We define logprob LP as

$$LP = -\frac{1}{N_H}\sum_{i=2}^{N_H} log(P(t_i \mid t_{i-1}, t_{i-2})_{smooth}) \tag{5.16}$$

where N_H is the length of the held–out data. LP can be viewed as the average of the appearance of the quantity $log(P(t_i \mid t_{i-1}, t_{i-2})_{smooth})$ in the heldout data. The idea is to minimize the term LP by computing the corresponding scaling parameters λ. Derived from the EM algorithm in [48], the iterative algorithm for computing the set of $\lambda_0 \ldots \lambda_K$ looks as follows:

1. Initialization step. Set $\lambda_j^0 = \frac{1}{K+1}$ for $j = 0, \ldots, K$

2. Expected counts computation.

$$C(\lambda_j^\tau) = \sum_{i=2}^{N_H} \frac{\lambda_j^\tau P_{MLE}(t_i \mid t_{i-1}, t_{i-2})}{P(t_i \mid t_{i-1}, t_{i-2})_{smooth}^\tau} \tag{5.17}$$

3. Iterative update for λ_j.

$$\lambda_j^{\tau+1} = \frac{C(\lambda_j^\tau)}{\sum_{j=0}^{K} C(\lambda_j^\tau)} \tag{5.18}$$

4. If $|\lambda_j^\tau - \lambda_j^{\tau+1}| < \epsilon$ print the set of λ_j^t else $\tau = \tau + 1$ and continue with step 2.

By computing the smoothing parameters values we have solved the problem of zero probability. We have one set of λ_j which guarantees non-zero probability even for a sequence which contains an unseen tag.

5.3.3 Bucketing

It is the perfect moment to realize that the values λ_j (once computed) are fixed and independent of the individual counts $N(t_{x-2}, t_{x-1}, t_x)$ and $N(t_{x-2}, t_{x-1})$. But it is obvious that $P_{MLE}(t_x \mid t_{x-1}, t_{x-2})$ will be more reliable if the estimate of P_{MLE} was based on a larger count $N(t_{x-2}, t_{x-1})$. This can be achieved by introducing a method called bucketing.

Our goal is to have multiple sets of λ_j^b, where b denotes the corresponding bucket $B^b(i,j)$. Each bucket is defined by an interval which it covers. The size of the interval is computed in such a way that each bucket contains approximately the same amount of trigrams from the training data. Given a history h we compute an index $v(h)$.

$$v(h) = \frac{N(h)}{|t : N(h,t) > 0|} \tag{5.19}$$

The corresponding bucket $B^b(i,j)$ for the index $v(h)$ has to satisfy $i < v(h) \leq j$. The table 5.1 shows an example of λ_j^b coefficients as computed for the Czech HMM tagger.

Linear smoothing is not the only way of smoothing. An extensive overview of smoothing methods used in language modeling can be found in [32].

	interval	λ_3	λ_2	λ_1	λ_0
Bucket 1 (least reliable histories)	$\langle 0; 1.000 \rangle$	0.0367	0.8077	0.1548	0.0006
Bucket 2	$\langle 1.000; 1.481 \rangle$	0.1825	0.6750	0.1421	0.0001
Bucket 3	$\langle 1.481; 1.840 \rangle$	0.2555	0.6161	0.1279	0.0002
Bucket 4	$\langle 1.840; 2.160 \rangle$	0.3257	0.5531	0.1206	0.0004
Bucket 35 (most reliable histories)	$\langle 51.84; 79.00 \rangle$	0.8634	0.1255	0.0109	0.0000

Table 5.1: Example smoothing coefficients for the probability $P(t_i \mid t_{i-1}, t_{i-2})_{smooth}$

5.4 Tagging of Inflective Languages

Inflective languages pose a specific problem in tagging due to two phenomena: highly inflective nature (causing sparse data problem in any statistically based system such as language model in chapter 4), and free word order (causing fixed-context systems, such as n-gram HMMs, to be even less adequate than for English). The average tagset contains about 1,000 - 2,000 distinct tags; the size of the set of possible and plausible tags can reach several thousands. There have been attempts at solving this problem for some of the highly inflective European languages, such as [20], [21] for Slovenian, or [28], [30] for Czech and [26] for five Central and Eastern European languages.

So far no system has reached - in the absolute terms - a performance comparable to English tagging (such as [46]), which stands above 97%.

5.4.1 Tagging Czech

Thanks to the Prague Dependency Treebank [2] project we can use about 1.8 million hand annotated tokens of Czech for training and testing. The HMM tagger uses the Czech morphological processor [27] to disambiguate only among those tags which are morphologically plausible.

The meaning of the Czech tags we are using is explained in table 5.2. The detailed explanation of the individual positions can be found in the appendix A.1.

No.	Name	Description
1	POS	Part of Speech
2	SUBPOS	Detailed Part of Speech
3	GENDER	Gender
4	NUMBER	Number
5	CASE	Case
6	POSSGENDER	Possessor's Gender
7	POSSNUMBER	Possessor's Number
8	PERSON	Person
9	TENSE	Tense
10	GRADE	Degree of comparison
11	NEGATION	Negation
12	VOICE	Voice
13	RESERVE1	Unused
14	RESERVE2	Unused
15	VAR	Variant, Style, Register, Special Usage

Table 5.2: Czech Morphology and the Positional Tags

5.4.2 Tagging Experiment for Czech

It is our target to design a tagger which will allow us to improve the error rates for the task of speech recognition. The language where we will demonstrate this is Czech and so we are interested in the performance of the Czech tagger itself. The best taggers for Czech (which use the same tagset) are reported to have the accuracy bellow ninety five percent [30].

To demonstrate the difficulties of tagging Czech let us investigate the following example. To follow the definitions of section 5.2 we have a sequence W for which the annotater created the following alignment (W, T).

(*Pak*, Db-------------) (*zasedal*, VpYS---XR-AA---)
(*dětský*, AAIS1----1A----) (*tribunál*, NNIS1-----A----).[5]

This sentence was passed to the morphological processor. The output contains for every word all the tags which are morphologically plausible.

(*Pak*, Db-------------, NNNP2-----A----)

[5] Then the children tribunal held the session.

(*zasedal*, VpYS---XR-AA---)
(*dětský*, AAFP1----1A---6, AAFP4----1A---6, AAFP5----1A---6,
AAFS2----1A---6, AAFS3----1A---6, AAFS6----1A---6, AAIP1----1A---6,
AAIP4----1A---6, AAIP5----1A---6, AAIS1----1A----, AAIS4----1A----,
AAIS5----1A----, AAMP1----1A---6, AAMP4----1A---6, AAMP5----1A---6,
AAMS1----1A----, AAMS5----1A----, AANP1----1A---6, AANP4----1A---6,
AANP5----1A---6, AANS1----1A---6, AANS4----1A---6, AANS5----1A---6)
(*tribunál*, NNIS1-----A----, NNIS4-----A----).

The first thing we realize is that the adjective *dětský* is highly ambiguous. We are not sure if it is singular or plural (fourth position), we don't even know if the gender is neuter, feminine or masculine(third position). The case (fifth position) can be nominative, genitive, dative, accusative or vocative. Some of the variants, as for example AAFS3----1A---6, can be used in spoken Czech only (position fifteen). But we must remind the reader that the morphological processor does not use any context information. What we see here are all plausible tags for the given word.

When examining the whole noun phrase *dětský tribunál* any Czech speaker will make the assumption that the gender and case shall be the same for both the adjective and the noun in this noun phrase.

Let us examinate what has happened in the next step, when the HMM tagger performed the Viterbi search for the best corresponding tag sequence.

(*Pak*, Db-------------) (*zasedal*, VpYS---XR-AA---) (*dětský*, AAIS4----1A----)
(*tribunál*, NNIS4-----A----).

The alignment created by the statistical tagger follows the natural feeling that the gender and case should be same for the noun and adjective in the noun phrase. However the alignment is not correct as the case of the noun phrase *dětský tribunál* is not accusative but nominative. This example characterizes one of the most common errors done by the HMM-tagger which is the substitution of accusative and nominative.

	Accuracy smoothing w/o bucketing	Accuracy (bucketing)
Exp 1	95.23%	95.34%
Exp 2	94.95%	95.13%
Exp 3	95.04%	95.19%
Exp 4	94.77%	95.04%
Exp 5	94.86%	95.11%
Average	94.97%	95.16%

Table 5.3: Evaluation of the HMM tagger on the Prague Dependency Treebank, 5-fold cross validation

The HMM tagger described in this chapter has achieved results shown in table 5.3. It has produced only the best tag sequence for every sentence (although the N-best decoding is also possible) therefore accuracy is reported only. Five-fold cross-validation has been performed on a total data size 1489983 of tokens (heldout data excluded), divided up to five datasets of roughly the same size. The source of the data is the Prague Dependency Treebank [2], where the distribution [1] of the tagger can be also found.

5.4.3 Further Improvements to the Czech HMM Tagging

We have strictly followed the principles of the HMM framework till now. The tagger was allowed to work with the left context only and so we succeeded with preparing it for the speech recognition experiment in section 4.5. However, there are applications in the field of computational linguistics [19] where the limitation to the left context is not needed (such as machine translation, parsing, or offline transcription). In these cases any method leading to better accuracy is acceptable.

In [29] the author of this thesis together with his colleagues introduced the serial combination of a rule–based component and a statistical HMM tagger. We shall note that the rule–based component has more than a context free power, which makes it impossible to use in the speech recognizer based on the HMM approach. The task for the manual rule component (which follows immediately after the morphological processor) is to keep the recall very close to 100%, with the task of improving precision as much as possible. The data flow in the serial combination can be described as

	Accuracy (bucketing)	Accuracy combined	Relative improvence
Exp 1	95.34%	95.53%	4.08%
Exp 2	95.13%	95.36%	4.72%
Exp 3	95.19%	95.41%	4.57%
Exp 4	95.04%	95.28%	4.84%
Exp 5	95.11%	95.34%	4.70%
Average	95.16%	95.38%	4.58%

Table 5.4: Evaluation of the combined tagger on the Prague Dependency Treebank, 5-fold cross validation

follows:

1. The morphological analyzer is run on the test data set. Every input token receives a list of possible tags based on an extensive Czech morphological dictionary.

2. The manual rule component is run on the output of the morphology. The rules eliminate some tags which cannot form grammatical sentences in Czech.

3. The HMM tagger is run on the output of the rule component, using only the remaining tags at every input token. The output is one best only; i.e., the tagger outputs exactly one tag per input token.

This combination (using exactly the same HMM tagger as described in this chapter) obtained 4.58% relative error reduction and become the best tagging tool for the Czech language. These improvements beat even the pure statistical classifier combination [30], which obtained 3% relative improvement only. The detailed results of the serial combination of a rule-based component and a statistical HMM tagger are to be found in table 5.4.

Chapter 6

Speech Recognition Experiment Using the Combination of a Hidden Tag Model and a Traditional Word Based N-Gram Model

6.1 Acoustic Data

Our acoustic corpus consists of 26 hours of clean speech of broadcast radio and TV news. Weather forecast, traffic announcements and sport news were excluded from the corpus. The following radio and TV stations are included in the corpus: ČT1, Nova, Prima, Radiožurnál, Praha, Vltava and Frekvence 1.

The channel has been sampled at 22.05 kHz with 16-bit resolution. 22 hours were used for acoustic modeling, the remaining four hours were used as the test set. The corpus was collected at the University of West Bohemia [42], which allows to directly compare the results of the proposed language model with the top scoring model in [33].

For the purpose of our experiment we have divided the available data (which were not included in the training) into two parts. Heldout data consists of 400 utterances and will be used for finding the best scaling factors. Test data contains 2500 utterances.

6.2 Acoustic Features

The acoustic features are Mel-Frequency Cepstral coefficients [43]. Each acoustic feature vector consists of twelve cepstral coefficients plus energy and their delta and delta–delta coefficients. Cepstral mean subtraction was applied to all feature vectors on a per utterance basis [56].

6.3 Lattice Rescoring

Obviously, it is very expensive to implement a large-vocabulary n-gram search (where $n > 2$) given the complexity of the search space. It becomes necessary to perform a multiple-pass search strategy, in which the first-pass search uses less detailed language model (bigrams in our case) to generate the word lattice, and then a second pass detailed search can use complex models on a much smaller search space.

The lattice is an oriented acyclic graph representing the output of the speech recognizer and is composed by word hypotheses. Each word hypothesis in the lattice is associated with a score and an explicit time interval (see figure 6-2). The lattice rescoring mechanism is a widely accepted method for the evaluation of the language models and it is supported by the main speech recognition toolkits [52][56].

6.4 Baseline System

In order to see how much improvement the integration of the tagger component will bring us, we decided to implement the best baseline we can achieve using traditional LM techniques. It is still impossible to run a full trigram decoder on word forms for Czech due to its vocabulary size. Thus we took a bigram decoder (using the AT&T tools [40][39]) and created lattices with it. The lattices have been transformed to trigram lattices and rescored with a trigram language model. The trigram model has been trained on a collection of Lidové Noviny (Czech daily newspaper) containing approximately 33 million words and it uses the Katz discounting method. The collection of Lidové Noviny [4] is a part of data collected by the Institute of Czech National Corpus[3].

Our bigram back-off language model used in the decoder and the trigram model used for lattices rescoring has been built with a vocabulary of 62k most frequent tokens. The outof–vocabulary rate of the transcriptions of the test data is 8.17%. We utilized [52] to estimate the corresponding back-off parameters of the language model as explained in section 4.4. The oracle accuracy of the held-out data lattices is 87.76%.

6.4.1 Scaling Factors

From our preceding experiments we learned that the correct setting of the scaling factors makes a significant difference on the WER. The scaling factors compensate the differences of the language and acoustic model. The fundamental equation of speech recognition 3.5 will actually lead to very unsatisfying results in the terms of accuracy. This happens due to the fact that the acoustic model assigns the probability to the acoustic observation a_i every 10 or 15 milliseconds, in other words each utterance will be assigned hundreds of probabilities by the acoustic model. The language model on the other side will assign only eight to ten probabilities $P(W)$ (as this is the average length of the sentence). The other feature we have to compensate for, is the inability of the acoustic model to consider strong correlation of adjacent acoustic observations a_i. The solution offered by [6] is to weight the acoustic model by a number f_{Ac} smaller then one. Other works (such as [53]) introduce another enhancement of the equation 3.5, the “word insertion penalty”.

The word insertion penalty is based on the assumption that the length of the sentence $|W|$ follows an exponential distribution. Using these two schemes we end up

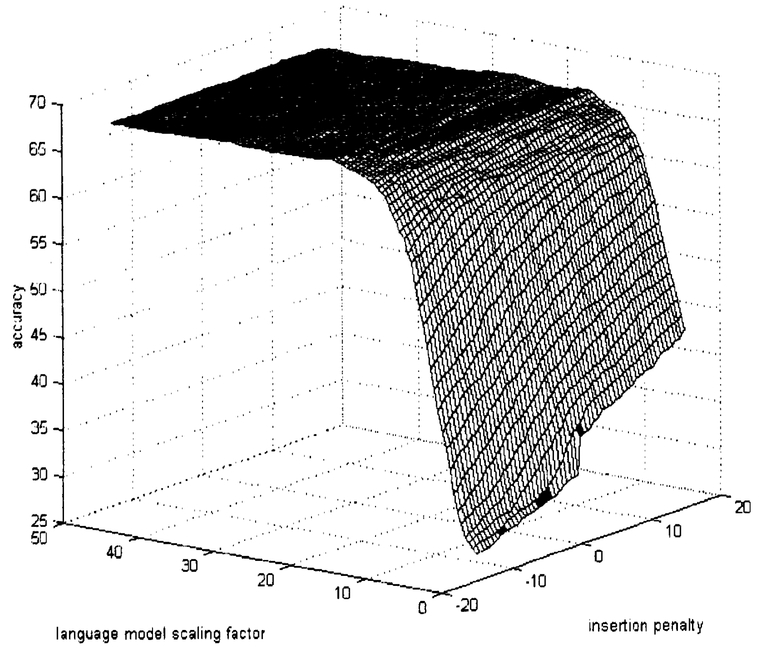

Figure 6-1: Impact of the scaling factor for an n-gram based language model

with

$$\hat{W} = \arg\max_{W}(\log P(W) + f_{Ac}\log P(A|W) + f_{IP}|W|) \tag{6.1}$$

We decided to use the equivalent approach and weight the language model with a factor f_{LM}. Based on the preceding experiments [14] we have also found that introducing the word insertion penalty does not get us any gain in the accuracy as long as we use the optimum scaling factors for the language model (see figure 6-1). On a set of held-out data (400 lattices) we found the optimal scaling factors (see table 6-1) for the baseline trigram LM and the acoustic model. The scaling factor f_{LM} is optimized for achieving the best accuracy on the held-out data with the following formula:

$$\hat{W} = \arg\max_{W}(f_{LM}\log P(W) + \log P(A|W)) \tag{6.2}$$

The baseline trigram system uses the scaling factor 15 as shown in Table 6.1. The Accuracy of the baseline system on the test data is 71.90%.

f_{LM}	**Accuracy**
11	70.35%
12	70.88%
13	71.33%
14	71.51%
15	72.04%
16	71.59%

Table 6.1: Finding the optimum scaling factor on the set of held-out data

6.5 Beating the N-Grams

The formula 4.16 gives us a hint how to combine the tagger component with the trigram language model. For practical reasons we decided to use a slightly different approach similar to the way we tuned our baseline. Our goal is to find the optimum scaling factors f_{LM} and f_{tag} on the same set of the held-out data as used for the baseline tuning. The formula 6.2 now becomes:

$$\hat{W} = \arg\max_{W}(f_{LM} \log P(W) + f_{tag} \log Q^{**} + \log P(A|W)) \tag{6.3}$$

The effect of tuning the parameters f_{LM} and f_{tag} can be seen in figure 6-3.

Our task is to find the optimum scaling parameters f_{tag} and f_{LM} on the set of held-out data in a similar way as we have done for the n-gram baseline. We can see from the figure 6-3 that the introduction of the tagger component Q^{**} leads to the accuracy improvement. The maximum accuracy gain point occurs with the scaling factors $f_{LM} = 10$ and $f_{tag} = 5$. The accuracy of this best setup is 72.97% on the held-out data. The results are summarized in table 6.2.

6.5.1 Discussion of the Results

We managed to beat the trigram baseline by 1.21% absolute. That corresponds to a relative improvement of 4.3% in the WER. We have also succeeded in improving the accuracy for Czech presented in [32], where the author uses the same data set for testing and the same vocabulary. What is even more important, we introduced a

Accuracy	Language model used
70.24%	bigram model
69.31%	trigram model $f_{LM} = 10$
71.90%	trigram model $f_{LM} = 15$ (baseline)
72.73%	best class based model introduced in [32]
73.11%	combination $f_{LM} = 10$, $f_{tag} = 5$
87.69%	oracle accuracy

Table 6.2: Test data experiments

new language model which is a combination of a traditional trigram language model and an HMM tagger. We achieved a promising improvement in accuracy compared to the baseline trigram model.

There still remains place for further improvement. The oracle[1] accuracy 87.9% of the lattices we have used still allows investigating new approaches in language modeling.

One of the advantages of the language model presented in this thesis is that it can be combined with the morpheme based approach as we have introduced in [14][13] and as it was further explored in [32].

[1] The best error which can be achieved using the test data lattices.

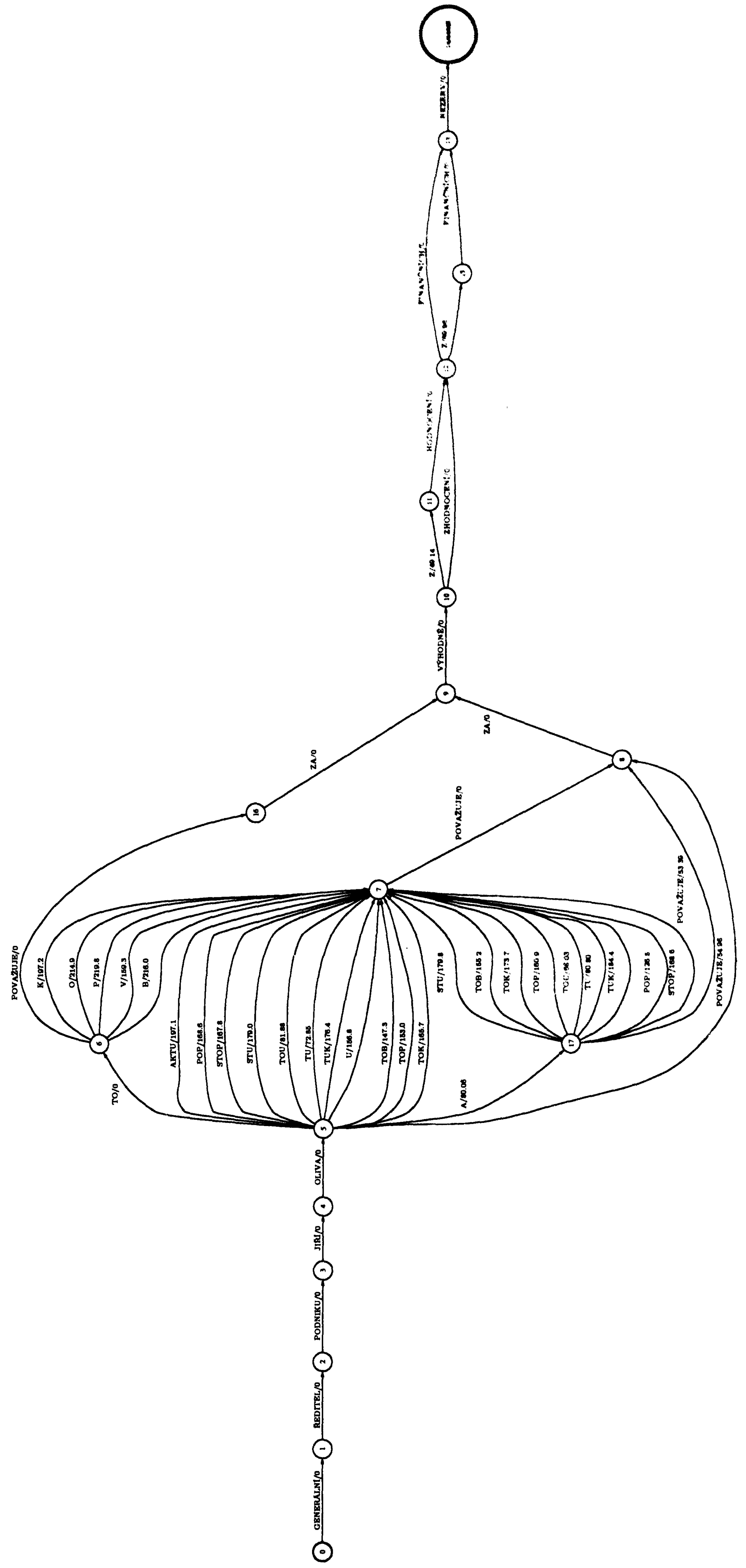

Figure 6-2: Lattice corresponding to a sentence "GENERÁLNÍ ŘEDITEL PODNIKU JIŘÍ OLIVA TO POVAŽUJE ZA VÝHODNÉ ZHODNOCENÍ FINANČNÍCH REZERV."

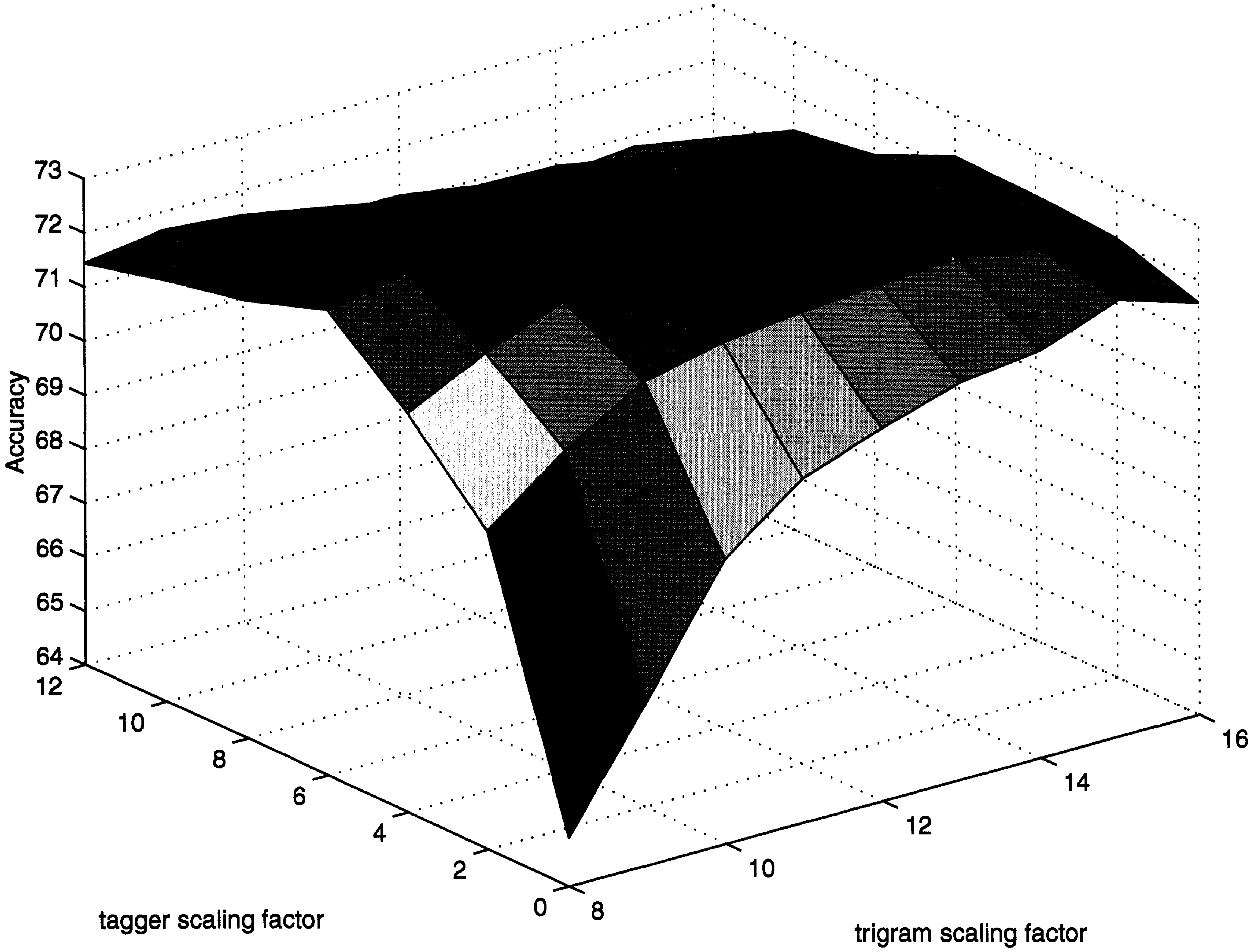

Figure 6-3: Accuracy as a function of f_{LM} and f_{tag}

Chapter 7

Conclusion

We have described the speech recognition system for Czech using the HMM framework. We have paid special attention to the language model component and we have shown some of the difficulties the designer of the LVCSR system has to face when working with inflective languages such as Czech.

We have created a trigram language model which served as a baseline. We have shown the weaknesses of this word based n-gram model and presented a solution to these weaknesses. We have proposed to create a language model using the underlying morphological information. We had the theory that the use of the morphological information will lead to more reliable statistics of the language model.

To incorporate the morphology into the language model properly, it was required to develop a robust statistical tagger for Czech. We have presented an implementation of an HMM tagger for Czech language. This tagger (when combined with a hand–written rule component) is the top disambiguation tool for the Czech language with the error rate less than five percent.

By combining the HMM based tagger with the word trigram model we have obtained significant reduction in the word error rate. We believe that the approach of combining the word n-gram based models with HMM taggers will lead to better accuracy in other inflective languages as well. The advantage of our solution is that we solely use the Viterbi left to right decoding approach and thus it is theoretically possible to use our language model in a single pass decoder approach as required by an LVSCR system with close–to–zero latency.

The speech recognition experiment has shown that there is a lot of space for further research in the area of language modeling. The oracle accuracy is still far from our best result and so we have to ask (together with the reader) which algorithms will allow us to narrow this gap. Is the Holy Grail of language modeling in the more complex methods such as SLM? The author of this thesis believes that the odyssey of achieving better error rates can be in the long run solved by the quantity of the training data. To quote Eric Brill[1]: "More data is more important than better algorithms". By stating that we do not have to stop investigating more complex language modeling techniques immediately (there are situations when we simply do not have enough data), but the brute force approach to statistical language modeling must be considered as an alternative even for highly inflective languages such as Czech.

[1] Eric Brill is a head of the Text Mining, Search and Navigation Group, Microsoft Research

Appendix A

Tables

A.1 Positional Tags: Quick Reference

Value	Description
A	Adjective
C	Numeral
D	Adverb
I	Interjection
J	Conjunction
N	Noun
P	Pronoun
V	Verb
R	Preposition
T	Particle
X	Unknown, Not Determined, Unclassifiable
Z	Punctuation (also used for the *Sentence Boundary* token)

Table A.1: Part of Speech

Value	Description
!	Abbreviation used as an adverb (now obsolete)
#	Sentence boundary (for the *virtual* word ###)
*	Word krát (lit.: *times*) (POS: C, numeral)
,	Conjunction subordinate (incl. aby, kdyby in all forms)

.	Abbreviation used as an adjective (now obsolete)
0	Preposition with attached -ň (pronoun něj, lit. *him*); proň, naň, ... (POS: P, pronoun)
1	Relative possessive pronoun jehož, jejíž, ... (lit. *whose* in subordinate relative clause)
2	Hyphen (always as a separate token)
3	Abbreviation used as a numeral (now obsolete)
4	Relative/interrogative pronoun with adjectival declension of both types (*soft* and *hard*) (jaký, který, čí, ..., lit. *what*, *which*, *whose*, ...)
5	The pronoun *he* in forms requested after any preposition (with prefix n-: něj, něho, ..., lit. *him* in various cases)
6	Reflexive pronoun se in long forms (sebe, sobě, sebou, lit. *myself* / *yourself* / *herself* / *himself* in various cases; se is personless)
7	Reflexive pronouns se (CASE = 4), si (CASE = 3), plus the same two forms with contracted -s: ses, sis (distinguished by PERSON = 2; also number is singular only)
8	Possessive reflexive pronoun svůj (lit. *my* /*your* /*her* /*his* when the possessor is the subject of the sentence)
9	Relative pronoun jenž, již, ... after a preposition (n-: něhož, niž, ..., lit. *who*)
:	Punctuation (except for the virtual sentence boundary word ###, which uses the SUBPOS #)
;	Abbreviation used as a noun (now obsolete)
=	Number written using digits (POS: C, numeral)
?	Numeral kolik (lit. *how many* /*how much*)
@	Unrecognized word form (POS: X, unknown)
A	Adjective, general
B	Verb, present or future form
C	Adjective, nominal (short, participial) form rád, schopen, ...

D	Pronoun, demonstrative (ten, onen, ... lit. *this*, *that*, *that ... over there*, ...)
E	Relative pronoun což (corresponding to English *which* in subordinate clauses referring to a part of the preceding text)
F	Preposition, part of; never appears isolated, always in a phrase (nehledě (na), vzhledem (k), ..., lit. *regardless*, *because of*)
G	Adjective derived from present transgressive form of a verb
H	Personal pronoun, clitical (short) form (mě, mi, ti, mu, ...); these forms are used in the second position in a clause (lit. *me*, *you*, *her*, *him*), even though some of them (mě) might be regularly used anywhere as well
I	Interjections (POS: I)
J	Relative pronoun jenž, již, ... not after a preposition (lit. *who*, *whom*)
K	Relative/interrogative pronoun kdo (lit. *who*), incl. forms with affixes -ž and -s (affixes are distinguished by the category VAR (for -ž) and PERSON (for -s))
L	Pronoun, indefinite všechen, sám (lit. *all*, *alone*)
M	Adjective derived from verbal past transgressive form
N	Noun (general)
O	Pronoun svůj, nesvůj, tentam alone (lit. *own self*, *not-in-mood*, *gone*)
P	Personal pronoun já, ty, on (lit. *I*, *you*, *he*) (incl. forms with the enclitic -s, e.g. tys, lit. *you're*); gender position is used for third person to distinguish on/ona/ono (lit. *he/she/it*), and number for all three persons
Q	Pronoun relative/interrogative co, copak, cožpak (lit. *what*, *isn't-it-true-that*)
R	Preposition (general, without vocalization)
S	Pronoun possessive můj, tvůj, jeho (lit. *my*, *your*, *his*); gender position used for third person to distinguish jeho, její, jeho (lit. *his, her, its*), and number for all three pronouns
T	Particle (POS: T, particle)

U	Adjective possessive (with the masculine ending - ův as well as feminine -in)
V	Preposition (with vocalization -e or -u): (ve, pode, ku, ..., lit. *in*, *under*, *to*)
W	Pronoun negative (nic, nikdo, nijaký, žádný, ..., lit. *nothing*, *nobody*, *not-worth-mentioning*, *no/none*)
X	(temporary) Word form recognized, but tag is missing in dictionary due to delays in (asynchronous) dictionary creation
Y	Pronoun relative/interrogative co as an enclitic (after a preposition) (oč, nač, zač, lit. *about what* , *on/onto what* , *after/for what*)
Z	Pronoun indefinite (nějaký, některý, číkoli, cosi, ..., lit. *some* , *some* , *anybody's* , *something*)
^	Conjunction (connecting main clauses, not subordinate)
a	Numeral, indefinite (mnoho, málo, tolik, několik, kdovíkolik, ..., lit. *much/many* , *little/few* , *that much/many* , *some (number of)* , *who-knows-how-much/many*)
b	Adverb (without a possibility to form negation and degrees of comparison, e.g. pozadu, naplocho, ..., lit. *behind* , *flatly*); i.e. both the NEGATION as well as the GRADE attributes in the same tag are marked by - (Not applicable)
c	Conditional (of the verb být (lit. *to be*) only) (by, bych, bys, bychom, byste, lit. *would*)
d	Numeral, generic with adjectival declension (dvojí, desaterý, ..., lit. *two-kinds/...* , *ten-...*)
e	Verb, transgressive present (endings -e/-ě, -íc, -íce)
f	Verb, infinitive
g	Adverb (forming negation (NEGATION set to A/N) and degrees of comparison GRADE set to 1/2/3 (comparative/superlative), e.g. velký, za\-jí\-ma\-vý, ..., lit. *big* , *interesting*

h	Numeral, generic; only jedny and nejedny (lit. *one-kind/sort-of* , *not-only-one-kind/sort-of*)
i	Verb, imperative form
j	Numeral, generic greater than or equal to 4 used as a syntactic noun (čtvero, desatero, . . . , lit. *four-kinds/sorts-of* , *ten-. . .*)
k	Numeral, generic greater than or equal to 4 used as a syntactic adjective, short form (čtvery, . . . , lit. *four-kinds/sorts-of*)
l	Numeral, cardinal jeden, dva, tři, čtyři, půl, . . . (lit. *one* , *two* , *three* , *four* , *half*); also sto and tisíc (lit. *hundred* , *thousand*) if noun declension is not used
m	Verb, past transgressive; also archaic present transgressive of perfective verbs (ex.: udělav, lit. *(he-)having-done* ; arch. also udělaje (VAR = 4), lit. *(he-)having-done*)
n	Numeral, cardinal greater than or equal to 5
o	Numeral, multiplicative indefinite (-krát, lit. (*times*): mnohokrát, tolikrát, . . . , lit. *many times* , *that many times*)
p	Verb, past participle, active (including forms with the enclitic -s, lit. *’re* (*are*))
q	Verb, past participle, active, with the enclitic -ť, lit. (perhaps) *-could-you-imagine-that?* or *but-because-* (both archaic)
r	Numeral, ordinal (adjective declension without degrees of comparison)
s	Verb, past participle, passive (including forms with the enclitic -s, lit. *’re* (*are*))
t	Verb, present or future tense, with the enclitic -ť, lit. (perhaps) *-could-you-imagine-that?* or *but-because-* (both archaic)
u	Numeral, interrogative kolikrát, lit. *how many times?*
v	Numeral, multiplicative, definite (-krát, lit. *times* : pětkrát, . . . , lit. *five times*)

w	Numeral, indefinite, adjectival declension (nejeden, tolikátý, ..., lit. *not-only-one* , *so-many-times-repeated*)
x	Abbreviation, part of speech unknown/indeterminable (now obsolete)
y	Numeral, fraction ending at -ina (POS: C, numeral); used as a noun (pětina, lit. *one-fifth*)
z	Numeral, interrogative kolikátý, lit. *what* (*at-what-position-place-in-a-sequence*)
}	Numeral, written using Roman numerals (XIV)
~	Abbreviation used as a verb (now obsolete)

Table A.2: Detailed Part of Speech

Value	Description
-	Not applicable
F	Feminine
H	Feminine or Neuter
I	Masculine inanimate
M	Masculine animate
N	Neuter
Q	Feminine (with singular only) or Neuter (with plural only); used only with participles and nominal forms of adjectives
T	Masculine inanimate or Feminine (plural only); used only with participles and nominal forms of adjectives
X	Any of the basic four genders
Y	Masculine (either animate or inanimate)
Z	Not fenimine (i.e., Masculine animate/inanimate or Neuter); only for (some) pronoun forms and certain numerals

Table A.3: Gender

Value	Description
-	Not applicable
D	Dual
P	Plural
S	Singular
W	Singular for feminine gender, plural with neuter; can only appear in participle or nominal adjective form with gender value Q
X	Any

Table A.4: Number

Value	Description
-	Not applicable
1	Nominative
2	Genitive
3	Dative
4	Accusative
5	Vocative
6	Locative
7	Instrumental
X	Any

Table A.5: Case

Value	**Description**
-	Not applicable
F	Feminine possessor
M	Masculine animate possessor (adjectives only)
X	Any gender
Z	Not feminine (both masculine or neuter)

Table A.6: Possgender

Value	**Description**
-	Not applicable
P	Plural (possessor)
S	Singular (possessor)

Table A.7: Possnumber

Value	**Description**
-	Not applicable
1	1st person
2	2nd person
3	3rd person
X	Any person

Table A.8: Person

Value	**Description**
-	Not applicable
F	Future
H	Past or Present
P	Present
R	Past
X	Any (Past, Present, or Future)

Table A.9: Tense

Value	**Description**
-	Not applicable
1	Positive
2	Comparative
3	Superlative

Table A.10: Grade

Value	Description
-	Not applicable
A	Affirmative (not negated)
N	Negated

Table A.11: Negation

Value	Description
-	Not applicable
A	Active
P	Passive

Table A.12: Voice

Value	Description
-	Not applicable

Table A.13: Reservel

Value	Description
-	Not applicable

Table A.14: Reserve2

Value	Description
-	Not applicable (basic variant, standard contemporary style; also used for standard forms allowed for use in writing by the Czech Standard Orthography Rules despite being marked there as colloquial)
1	Variant, second most used (*less frequent*), still standard
2	Variant, rarely used, bookish, or archaic
3	Very archaic, also archaic + colloquial
4	Very archaic or bookish, but standard at the time
5	Colloquial, but (almost) tolerated even in public
6	Colloquial (standard in spoken Czech)
7	Colloquial (standard in spoken Czech), less frequent variant
8	Abbreviations
9	Special uses, e.g. personal pronouns after prepositions etc.

Table A.15: Var

Appendix B

Examples of Recognized Sentences

This appendix contains the sample output of the speech recognizer using the best performing language model as described in section 6.5. Sentences which were misrecognized are included only. W and $\hat{W}$ denotes the correct sentence and the recognized utterance respectively.

W: OHLASY SHRNUJE NAŠE BRATISLAVSKÁ ZPRAVODAJKA RENATA HAVRANOVÁ.

$\hat{W}$: OHLASY SHRNUJE NAŠE BRATISLAVSKÁ ZPRAVODAJKA VRÁNOVÁ.

W: SHROMÁŽDĚNÍ DOPORUČILO TOTIŽ SVÉMU VÝBORU MINISTRŮ ABY ZAHÁJIL OKAMŽITOU PROCEDURU POZASTAVENÍ ČLENSTVÍ RUSKA KVŮLI ČEČENSKU.

$\hat{W}$: SHROMÁŽDĚNÍ DOPORUČILO TOTIŽ SVÉMU VÝBORU MINISTRŮ ABY ZAHÁJÍ OKAMŽITOU PROCEDURU POZASTAVENÍ ČLENSTVÍ RUSKA KVŮLI ČEČENSKU.

W: PODLE IVANOVA ČINÍ ALE VĚTŠINA EVROPSKÝCH POSLANCŮ ČASTÁ BEZPRECEDENTNÍ ROZHODNUTÍ NA ZÁKLADĚ JEDNOSTRANNÝCH INFORMACÍ POCHÁZEJÍCÍCH OD ČEČENSKÝCH TERORISTŮ.

$\hat{W}$: PODLE IVANOV A ČINÍ ALE VĚTŠINA EVROPSKÝCH POSLANCŮ ČEST A BEZPRECEDENTNÍ ROZHODNUTÍ NA ZÁKLADĚ JEDNOSTRANNÝCH INFORMACÍ POCHÁZEJÍCÍCH OD ČEČENSKÝCH TERORISTŮ.

W: RUSKÁ PARLAMENTNÍ DELEGACE KTERÁ VČERA OPUSTILA JEDNÁNÍ SHROMÁŽDĚNÍ RADY PO TÉ CO BYLA ZBAVENA HLASOVACÍHO PRÁVA SE MÁ VRÁTIT DO MOSKVY DNES VEČER.

$\hat{W}$: RUSKÁ PARLAMENTNÍ DELEGACE KTERÉ VČERA OPUSTILA JEDNÁNÍ SHROMÁŽDĚNÍ RADY POTÉ CO BYL ZBAVEN HLASOVACÍM PRÁVEM SE MÁ VRÁTIT DO MOSKVY DNES VEČER.

W: SPISOVATEL JOSEF ŠKVORECKÝ PŘILETĚL Z KANADY DO PRAHY ABY SE ZÚČASTNIL AKCE NAZVANÉ NONSTOP ČTENÍ.

$\hat{W}$: SPISOVATEL JOSEF ŠKVORECKÝ PŘILETĚL Z KANADY DO PRAHY ABY SE ZÚČASTNÍ AKCE NAZVANÉ NONSTOP ČTENÍ.

W: DVAASEDMDESÁTIHODINOVÝ MARATON PŘEDČÍTÁNÍ UKÁZEK Z JEHO LITERÁRNÍ TVORBY ZAČNE V KOSTELE SVATÉHO SALVÁTORA V NEDĚLI V ŠESTNÁCT HODIN.

$\hat{W}$: NÁSILÍ SETIN HODINOVÝ MARATÓN PŘI SČÍTÁNÍ UKÁZEK Z JEHO LITERÁRNÍ TVORBY JICH ZAČNE V KOSTELE SVATÉHO SALVÁTORA V NEDĚLI V ŠESTNÁCT HODIN.

W: DOZVĚDĚT JESTLI BUDEME PRO CESTY DO KANADY I NADÁLE POTŘEBOVAT VÍZA.

$\hat{W}$: Z VĚDĚT JESTLI BUDEME PRO CESTY DO KANADY I NADÁLE POTŘEBOVAT VÍZA.

W: PRÁVĚ DNES SE TOTIŽ CHYSTÁ MINISTERSTVO ZAHRANIČÍ ZVEŘEJNIT PO DALŠÍCH JEDNÁNÍCH NEJNOVĚJŠÍ STANOVISKO KANADY.

$\hat{W}$: PRÁVĚ DNES SE TOTIŽ CHYSTÁ MINISTERSTVO ZAHRANIČÍ ZVEŘEJNIT O DALŠÍCH JEDNÁNÍCH NEJNOVĚJŠÍ STANOVISKO KANADY.

W: TO PŘITOM ŠÉF DIPLOMACIE JAN KAVAN ZNÁ UŽ DVA DNY A ZATÍM HO TAJIL.

$\hat{W}$: PŘITOM ŠÉF DIPLOMACIE JAN KAVAN ZNÁ UŽ DVA DNY A ZATÍM UTAJENÍ.

W: ROBERT MIKOLÁŠ SHRNUJE JAK SE CELÁ ZÁLEŽITOST S VÍZY DO TÉTO SEVEROAMERICKÉ ZEMĚ VYVÍJELA.

$\hat{W}$: ROBERT MIKOLÁŠ SHRNUJE JAK SE CELÁ ZÁLEŽITOST Z VÝZVY DO TÉTO SEVEROAMERICKÉ ZIMNÍ JEHO.

W: TŘINÁCTÉHO BŘEZNA DEVATENÁCT SET DEVADESÁT ŠEST DOŠLO MEZI KANADOU A ČESKOU REPUBLIKOU K VÝMĚNĚ NÓT RUŠÍCÍCH VÍZOVOU POVINNOST MEZI OBĚMA ZEMĚMI A TATO DOHODA VSTOUPILA V PLATNOST O NĚKOLIK DNÍ POZDĚJI PRVNÍHO RESPEKTIVE Z KANADSKÉ STRANY DRUHÉHO DUBNA.

$\hat{W}$: TŘINÁCTÉHO BŘEZNA DEVATENÁCT SET DEVADESÁT ŠEST DOŠLO MEZI KANADOU A ČESKOU REPUBLIKOU K VÝMĚNĚ NOC V KOŠICÍCH VÝZVOU POVINNOST MEZI OBĚMA ZEMĚMI A TATO DOHODA VSTOUPILA V PLATNOST O NĚKOLIK DNÍ POZDĚJI PRVNÍHO RESPEKTUJE Z KANADSKÉ STRANY DRUHÉHO DUBNA.

W: VRAŤME SE ALE DO OBDOBÍ ROZVOJE STYKU MEZI KANADOU A ČESKOU REPUBLIKOU JAK UŽ JSEM ŘEKL VÍZOVÁ POVINNOST BYLA ZRUŠENA V ROCE DEVADESÁT ŠEST.

$\hat{W}$: VRAŤME SE ALE DO OBDOBÍ ROZVOJ STYKŮ MEZI KANADOU A ČESKOU REPUBLIKOU V USA ŘEKL VÝZVA POVINNOST BYLO ZRUŠENO ROCE DEVADESÁT ŠEST.

W: DŮVODEM BYL VZESTUP ŽÁDOSTÍ ČESKÝCH OBČANŮ PŘEDEVŠÍM ROMŮ O AZYL V KANADĚ KTERÝ PRÁVĚ V ROCE DEVADESÁT SEDM DRAMATICKY STOUPL.

$\hat{W}$: DŮVODEM BYL VZESTUP ŽÁDOSTI ČESKÝCH OBČANŮ PŘEDEVŠÍM ROMŮ O AZYL V KANADĚ KTERÝ PRÁVĚ V ROCE DEVADESÁT SEDM DRAMATICKY STOUPLA.

W: KDYŽ SI PRO PŘÍKLAD SROVNÁME ROK DEVATENÁCT SET DEVADESÁT PĚT KDY KANADU POŽÁDALO O AZYL JEN DEVĚTADVACET OBČANŮ

ČESKÉ REPUBLIKY V ROCE DEVADESÁT SEDM TO UŽ BYLO DVANÁCT SET OSMDESÁT PĚT ŽADATELŮ TEDY ZA DEVĚT MĚSÍCŮ TOHO ROKU.

$\hat{W}$: KDYŽ SI NAPŘÍKLAD SROVNÁME ROK DOTACE DEVADESÁT PĚT KDY KANADU POŽÁDALO O AZYL JEN DEVĚT SET OBČANŮ ČESKÉ REPUBLIKY V ROCE DEVADESÁT SEDM TO BYLO TO UŽ BYLO DVANÁCT OSMDESÁT PĚT ŽADATELŮ TEDY ZA DEVĚT MĚSÍCŮ TOHOTO ROKU.

W: KANADA SE STALA CÍLEM MNOHA LIDÍ I PROTO ŽE TAMNÍ ÚŘADY HRADILY ŽADATELŮM AŽ DO KONEČNÉHO ROZHODNUTÍ VŠECHNY NÁKLADY VČETNĚ UBYTOVÁNÍ STRAVY A ZDRAVOTNÍHO POJIŠTĚNÍ.

$\hat{W}$: KANADA SE STALA CÍLEM MNOHA LIDÍ PROTOŽE TAMNÍ ÚŘADY ZTRATILI ŽADATELŮM AŽ DO KONEČNÉHO ROZHODNUTÍ VŠECHNY NÁKLADY VČETNĚ UBYTOVÁNÍ STRAVY A ZDRAVOTNÍHO POJIŠTĚNÍ.

W: TATO SKUTEČNOST TAK VEDLA ČESKÉ PŘEDSTAVITELE K TOMU ŽE POŽÁDALI KANADSKÉ ČINITELE ABY PŘEZKOUMALI SVÉ ROZHODNUTÍ ZEJMÉNA PAK PREZIDENT VÁCLAV HAVEL KTERÝ SE BĚHEM SVÉ LOŇSKÉ NÁVŠTĚVY KANADY DOHODL S PREMIÉREM JEANEM CRÉTIENEM NA PŘEZKOUMÁNÍ DŮVODŮ VEDOUCÍCH K OBNOVENÍ VÍZOVÉ POVINNOSTI PRO OBČANY ČESKÉ REPUBLIKY.

$\hat{W}$: TATO SKUTEČNOST VEDLA ČESKÉ PŘEDSTAVITELE K TOMU ŽE POŽÁDALI KANADSKÉ ČINITELE ABY PŘEZKOUMAT I SVÉ ROZHODNUTÍ ZEJMÉNA PREZIDENT VÁCLAV HAVEL KTERÝ SE BĚHEM SVÉ LOŇSKÉ NÁVŠTĚVY KANADY DOHODU S PREMIÉREM JEANEM TEĎ JEN NA PŘEZKOUMÁNÍ DŮVODŮ VEDOUCÍCH K VEDENÍ VÍZOVÉ POVINNOSTI PRO OBČANY ČESKÉ REPUBLIKY.

Bibliography

[1] http://quest.ms.mff.cuni.cz/pdt/Morphology_and_Tagging/Tagging/MM_tagger/.

[2] http://shadow.ms.mff.cuni.cz/pdt/.

[3] http://ucnk.ff.cuni.cz/.

[4] http://www.lidovenoviny.cz/.

[5] http://www.w3.org/TR/jsgf/.

[6] L. Bahl, R. Bakis, F. Jelinek, and R. Mercer. Language-model/acoustic channel balance mechanism. *IBM Technical Disclosure Bulletin*, pages 3464–3465, 1980.

[7] L. Bahl, P. S. Gopalakrishnan, and R.L. Mercer. Search issues in large vocabulary speech recognition. In *Proceedings of the 1993 IEEE Workshop on Automatic Speech Recognition*, Snowbird, 1993.

[8] L. Bahl, F. Jelinek, and R. Mercer. A Maximum Likelihood Approach to Continuous Speech Recognition. *IEEE Journal of Pattern Analysis and Machine Intelligence*, 1983.

[9] L. Bahl and R.L. Mercer. Part of speech assignment by a statistical algorithm. In *IEEE International Symposium on Information Theory*, Ronneby, Sweden, 1976.

[10] T. Beran, V. Bergl, R. Hampl, P. Krbec, Jan Šedivý, B. Tydlitát, and J. Vopička. Embedded ViaVoice. In *Proceedings of TSD 2004*, Brno, 2004.

[11] E. Brill, D. Harris, S. Lowe, X. Luo, P.S. Rao, E. Ristad, and S. Roukos. A Hidden Tag Model for Language, LM workshop. Technical report, Johns Hopkins University, Baltimore, 1995.

[12] L. Burnard. *Users Reference Guide for the British National Corpus*, 1995.

[13] W. Byrne, J. Hajič, P. Ircing, F. Jelinek, S. Khudanpur, P. Krbec, and J. Psutka. On Large Vocabulary Continuous Speech Recognition of Highly Inflectional Language - Czech. In *Proceedings of Eurospeech 2001*, Aalborg, 2001.

[14] W. Byrne, J. Hajič, P. Ircing, P. Krbec, and J. Psutka. Morpheme Based Language Models for Speech Recognition of Czech. In *Proceedings of TSD 2000*, Brno, 2000.

[15] C. Chelba. *Exploiting Syntactic Structure for Natural Language Modeling*. PhD thesis, Johns Hopkins University, 2000.

[16] C. Chelba and P. Xu. Richer Syntactic Dependencies for Structured Language Modeling. In *IEEE Workshop on Automatic Speech Recognition and Understanding*, Madonna di Campiglio, Italy, 2001.

[17] N. Chomsky. *Syntactic Structures*. Mouton, The Hague, 1957.

[18] K. Church. A stochastic parts program noun phrase parser for unrestricted text. In *Proceedings of ICASSP*, Glasgow, 1989.

[19] Jan Cuřín, Martin Čmejrek, and Jiří Havelka. Czech-English Dependency-based Machine Translation. In *Proceedings of the 10th Conference of The European Chapter of the Association for Computational Linguistics*, pages 83–90, Budapest, Hungary, 2003.

[20] W. Daelemans, Jakub Zavrel, Peter Berck, and Steven Gillis. MBT: A memory-based part of speech tagger generator. In *Proceedings of the WVLC 4*, 1996.

[21] Tomaz Erjavec, Saso Dzeroskix, and Jakub Zavrel. Morphosyntactic Tagging of Slovene: Evaluating PoS Taggers and Tagsets. Technical report, Dept. for Intelligent Systems, Jozef Stefan Institute, Ljubljana, Slovenia, 1999.

[22] M. J. F. Gales. Maximum likelihood linear transformations for hmm-based speech recognition. Technical report, CUED, 1997.

[23] Vaibhava Goel. *Minimum Bayes-Risk Automatic Speech Recognition.* PhD thesis, Johns Hopkins University, 2001.

[24] A. J. Good. The Population Frequencies of Species and the Estimation of Population Parameters. In *Biometrika*, 1953.

[25] A. Gunawardana, H.W. Hon, and L. Jiang. Word-Based Acoustic Confidence Measures for Large-Vocabulary Speech Recognition. In *Int. Conf. on Spoken Language Processing.*

[26] J. Hajič. Morphological tagging: Data vs. dictionaries. In *Proceedings of the NAACL '00*, Seattle, WA, 2000.

[27] J. Hajič. *Disambiguation of Rich Inflection (Computational Morphology of Czech).* Karolinum Press, Charles University, Prague, 2001.

[28] J. Hajič and B. Vidová Hladká. Tagging inflective languages: Prediction of morphological categories for a rich, structured tagset. In *Proceedings of the Conference COLING - ACL '98*, Mountreal, Canada, 1998.

[29] J. Hajič, P. Krbec, P. Kveton, K. Oliva, and V. Petkevič. Serial Combination of Rules and Statistics: A Case Study in Czech Tagging. In *Proceedings of the ACL '01*, pages 260–267, Toulouse, France, 2001.

[30] B. Hladká. *Czech language tagging.* PhD thesis, Charles University, Institute of Formal and Applied Linguistics, 2000.

[31] Xuedong Huang, Alex Acero, and Hsiao-Wuen Hon. *Spoken Language Processing: A Guide to Theory, Algorithm and System Development.* Prentice Hall PTR, 2001.

[32] P. Ircing. *Large Vocabulary Continuous Speech Recognition of Highly Iflectional Language (Czech).* PhD thesis, University of West Bohemia, Pilsen, 2003.

[33] P. Ircing and J. Psutka. Comparison of Word-based and Class-based Language Models for Speech Recognition of the Czech Weather Forecast. In *Proceedings of ICSPAT 2000*, Dallas, 2000.

[34] F. Jelinek. *Statistical Methods for Speech Recognition.* The MIT Press, Cambridge, 1997.

[35] S. Katz. Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 35(3):400–401, 1987.

[36] P. Kenny, R. Hollan, Gupta VN, M. Lennig, Mermelstein P., and O'Shaughnessy D. A* admissible heuristics for rapid lexical access. In *IEEE Trans. Speech and Audio Processing*, 1993.

[37] Pavel Krbec, Petr Podveský, and Jan Hajič. Combination of a Hidden Tag Model and a Traditional N-gram Model: A Case Study in Czech Speech Recognition. In *EUROSPEECH 2003 Proceedings (8th European Conference on Speech Communication and Technology)*, volume 3, pages 2289–2291. ISCA, September 1–4 2003.

[38] B. Merialdo. Tagging English text with a probabilistic model. *Computational Linguistics*, 20(2):72–155, June 1984.

[39] M. Mohri. Finite-State Transducers in Language and Speech Processing. *Computational Linguistics*, 23(2):269–311, 1997.

[40] M. Mohri, F. Pereira, and M. Riley. Weighted Finite-State Transducers in Speech Recognition. In *Proceedings of ASR2000, International Workshop on Automatic Speech Recognition: Challenges for the Next Millennium*, Paris, 2000.

[41] M. Novak, R. Hampl, P. Krbec, V. Bergl, and J. Sedivy. Two-pass Strategy for Large List Recognition on Embedded Speech Recognition Platforms. *Proceedings of ICASSP 2003*, 2003.

[42] J. Psutka, V. Radová, L. Müller, J. Matoušek, P. Ircing, and D. Graff. Large Broadcast News and Read Speech Corpora of Spoken Czech. In *Proceedings of Eurospeech 2001*, Aalborg, 2001.

[43] J.V. Psutka. Využití perceptuální lineární prediktivní analýzy pro rozpoznávání souvislé řeči v telefonní kvalitě. Master's thesis, ZČU-FAV, 2001.

[44] Thomas F. Quatieri. *Discrete-Time Speech Signal Processing: Principles and Practice.* Prentice Hall PTR, 2001.

[45] Lawrence R. Rabiner and Ronald W. Schafer. *Digital Processing of Speech Signals.* Pearson Education, 1978.

[46] A. Ratnaparkhi. A maximum entropy model for part-of-speech tagging. In *Proceedings of EMNLP 1*, pages 133–142, 1996.

[47] D. R. Reddy. An approach to computer speech recognition by direct analysis of the speech wave. Technical report, Stanford University, 1966.

[48] R. Rosenfeld. *Description of the algorithm for finding optimal linear interpolation coefficients* λ_i, 2001.

[49] M. Siu and H. Gish. Evaluation of word confidence for speech recognition systems. *Computer Speech and Language*, pages 299–319, 1999.

[50] R. Sproat. Introduction to Computational Linguistics, 2003.

[51] Stendhal. *Le Rouge et le Noir.* 1830.

[52] A. Stolcke. SRILM - an Extensible Language Modeling Toolkit. In *Proceedings of ICSLP 2002*, Denver, 2002.

[53] D. Vergyri. *Integration of Multiple Knowledge Sources in Speech Recognition Using Minimum Error Training.* PhD thesis, Johns Hopkins University, 2000.

[54] A.J. Viterbi. Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm. *IEEE Trans. on Information Theory*, 1967.

[55] E. Whittaker and P. Woodland. Language Modelling for Russian and English Using Words and Classes. *Computer Speech and Language*, 17:87–104, 2003.

[56] S. Young, D. Kershaw, J. Odell, D, Ollason, V. Valtchev, and P. Woodland. *The HTK Book.* Entropic, Cambridge, 2000.

[57] Phil Zimmermann. PGP-INTRO(7), PGP 2.6.2. man pages.