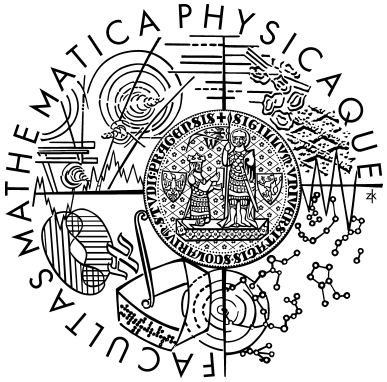
Speech Recognition of Czech Using Finite-State Machines



Petr Podveský

Institute of Formal and Applied Linguistics

Dissertation Thesis

Author: Petr Podveský

Advisor: Doc. RNDr. Jan Hajič, Dr.
Ústav formální a aplikované lingvistiky, MFF UK

Department: Ústav formální a aplikované lingvistiky, MFF UK,
Malostranské náměstí 25, 118 00 Praha 1

Opponents: NAME. **********,
Katedra kybernetiky,
Fakulta aplikovaných věd Západočeské univerzity v Plzni,
Univerzitní 8, 306 14 Plzeň

Name. **********,
Department,
Address

Abstract

Speech recognition has become a thriving field with many real-life applications. Voice dialing in cell phones, voice control in embedded devices, speech-driven interactive manuals and many other utilities rely on solid speech recognition software. We believe that research in speech recognition can boost performance of many applications related to the area.

The thesis concentrates on automatic large-vocabulary continuous-speech recognition of Czech. Czech differs from English in a few aspects. We focus on these differences and propose new language-depended techniques. Namely rich morphology is investigated and its impact on speech recognition is studied.

Out-of-vocabulary (OOV) words are identified as one of the major sources deteriorating recognition performace. New language modeling techniques are proposed to alleviate the problem of OOV words.

The proposed language models are tested in speech recognition systems on diverse speech corpora. The obtained results validate the original approach to language modeling. Significant overall speech recognition improvement is observed.

Declaration

I hereby declare that this thesis is my own work and where it draws on the work of others it is properly cited in the text.

Acknowledgments

First of all I would like to thank my parents for their continuous support during my studies. They encouraged me to complete my education.

I thank my wife for her love and for making me happy.

My supervisor Jan Hajič guided my research. I am also grateful to him for arranging a one-year stay at the Center for Language and Speech Processing, Johns Hopkins University, USA. It was a great experience. I learned a lot on speech recognition during the visit.

This research would not be possible without financial support of the Faculty of Mathematics and Physics and the Center for Computational Linguistics where I was employed as a research assistant.

Finally, I would like to express my gratitude to my colleagues Nino Peterek, Pavel Ircing, Pavel Krbec and David Mrva for numerous discussions, Honza Cuřín, Martin Čmejrek, Pavel Květoň, Roman Ondruška, Pavel Pecina, Jiří Mírovský, Jirka Semecký and Jura Havelka for keeping a pleasant working environment in the office and in the department corridors.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Communication in natural language with electronic devices and software applications has been an ultimate goal for speech scientist for many decades. Not until recently became speech recognition systems part of everyday life.

Voice dialing has already been implemented in standard cell phones, so instead of pressing tiny buttons on a small device, a user is invited to simply say whom he or she wants to call. Usually the user has to record names or voice commands first. Later on, when he says a name or a command the best match from the prerecorded list is found and the number associated with the name is dialed or the command associated with the sound is executed. All major cell phone producers include voice dialers into some of their products.

Navigation systems are becoming an inherent part of luxury cars. In these systems, driver specifies his destination and the navigation system displays the optimal route and may even give a guidance during the trip. A speech system comes in handy when selecting the destination. There is a tremendous number of cities, towns, villages, streets and avenues the driver may choose from. Selecting the location from a never-ending list by pressing keys is surely an annoying burden. Simply saying the address is then clearly preferable. During the trip the driver may decide to change the destination or make a detour to visit some astonish landmarks. Clearly, modifying the route

in spoken words disturbs the driver less then deciphering information displayed on embedded car display. Such a system has been developed by IBM and Honda Motor Company. Honda now offers the voice-driven navigation system as the standard equipment in some models, for example Acura RL is equipped with this system. More technical details can be found at `http://www.research.ibm.com`

NASA has also found speech processing very helpful. Astronauts at orbit stations have many diverse duties which include, for example, doing scientific experiment. Since the variety of tasks is very wide, astronauts are provided with electronic or printed manuals. It turned out that browsing the manuals while carring out assignments is impractical. Therefore a new voice-driven dialog system was developed to communicate with the electronic guidelines. A user can browse the manual in spontaneous speech and gets appropriate response. This system is nicely introduced in [*Rayner* et al. (2005)].

Another application of speech recognition is a medical dictation system. For doctors who have to type long medical reports, it is much faster to dictate a diagnosed disease description and suggested treatment instead of wasting their time by typewriting it, not mentioning that they are usually very bad typists. Such dictation systems are being developed and tested by medical stuff at hospitals. Unfortunately, current speech recognition systems are not 100% accurate, so human correction of the generated speech transcription is inevitable. However such corrections can be done by less educated people, thus saving time of doctors and consequently cutting cost of whole medical treatment.

Speaking dialog systems can be also utilized at companies with large clientele. Consider a client calling a big institution such as bank, telephone or insurance company for a particular piece of information. The phone call has to be redirected to the right department. The current systems force the client to press keys on their phones to navigate through a branching tree of possible divisions while listening to prerecorded, often irritating, instructions. If a wrong button is accidentally pressed, the annoying procedure has to be repeated from the beginning. When a dialog system is exploited, the client can specify his or her needs spontaneously, and the system transfers the

call accordingly. This kind of application has been usefully used at AT&T [*Wright*, *Gorin*, and *Abella* (1998)].

In automatic ticket reservation task, when booking concert tickets, airplane tickets, use of a dialog system is desirable as well. The demanding job of telephone operators whose only responsibility is to fill out a database according to a short conversation can be replaced by an automatic system. A client then speaks to a machine instead of a human being.

The task of audio indexing has drawn attention recently. There are large speech corpora available which contain valuable information for many people of different background. For instance, content of telephone conversations might be of interest or finding a relevant audio segment in a business meeting recordings. So the task is to find the relevant audio segment given key words or phrase. Obviously, manually transcribing whole speech corpus and then running an information retrieval system is very time-consuming and sometimes infeasible. Therefore, an automatic procedure which would select the relevant audio portions is needed. See [*Chelba* and *Acero* (2005)] for a viable indexing method.

Not all broad-casted TV programs nor movies are being subtitled nowadays. The deaf would certainly welcome an automatic subtitling system. This task is especially challenging due many factors which deteriorate recognition performance. Severe background noise, loud background music, people talking at the same time, unrestricted topic of conversations, all make speech recognition difficult to work at high accuracy rates. Unfortunately, high accuracy is essential to practical usage of such a system. Some experiments on automatic transcribing broadcasted ice-hockey match and further analysis of the problem on Czech data was discussed in [*Hajič* and *Psutka* (2003)].

The above mentioned applications indicate that speech recognition is a vital field and that speech recognition is worth further research. Some of the applications can handle erroneous recognition particularly those which do information retrieval on the recognition output. Others are sensitive to speech recognition accuracy such as medical dictation systems. Nevertheless in both cases boosting recognition performance

increases overall system performance. Therefore, we believe that effort put in speech recognition research will bring even wider use of speech technologies.

1.2 Czech

The chief interest of this work is speech recognition of Czech language. Since Czech differs from English in many aspects, we believe that developing language-specific modifications can bring significant improvement which would not be possible with standard English-like techniques.

The contrast between Czech and English from the point of view of speech recognition lies in word order and morphology. While English has very strict rules on the ordering of words in a sentence, Czech allows many word permutations. Intonation and stress is crucial to understanding of the correct meaning in Czech. Morphology of English as opposed to Czech is rather limited. Czech words get inflected into many forms. Careful handling of these forms is therefore desired in speech recognition of Czech.

We will further discuss issues of Czech in section 5.

1.3 Noisy Channel

After profound research which lasted several decades, the fundamental approach to large-vocabulary continuous speech recognition has settled down to the Bayesian approach. We have adopted this modeling technique throughout our work too.

The task of speech recognition is to transcribe speech signal to text. That is, given an acoustic evidence $O = o_1 \dots o_m$, we are after the word sequence $W = w_1 \dots w_n$ which was uttered. Acoustic evidence O is a sequence of feature vectors or quantized values obtained from acoustic signal by so-called *front-end* preprocessing. The feature vectors should contain as much of relevant information for speech recognition as possible.

Put formally, we are searching for the word sequence $\widehat{W}$ which maximizes the

conditional probability $P(W|O)$ given the acoustic evince O.

$$\widehat{W} = \arg\max_{W} P(W|O) \tag{1.1}$$

Using the Bayes rule, this tremendous task can be decomposed into manageable components.

$$P(W|O) = \frac{P(O|W) \cdot P(W)}{P(O)} \tag{1.2}$$

We aim at finding the word sequence $\widehat{W}$ that maximizes $P(W|O)$. Since $P(O)$ does not influence the value of $\widehat{W}$, the denominator can be discarded and equation 1.2 can be rewritten as

$$\widehat{W} = \arg\max_{W} P(O|W) \cdot P(W) \tag{1.3}$$

Formula 1.3 breaks down the problem into four basic components, the front-end preprocessing which generates acoustic observation O from signal, the acoustic model $P(O|W)$, the language model P(W) and the search involved in decoding of the word sequence $\widehat{W}$. [*Jelinek* (1997)]

1.4 Signal Preprocessing

In the first phase of speech recognition process, speech signal is digitalized. Since the raw data are not suitable as an input for a recognition system, feature vectors are extracted using various signal processing techniques. Feature vectors should be of low dimension to admit further modeling and should well represent speech in the sense of keeping relevant information. In addition, they should be computationally feasible, and should be affected as little as possible by environmental factors such as background noise, recording device, speaker's age, etc.

Among most popular features are Mel-Frequency Cepstral Coefficients (MFCC), Perceptual Linear Predictive (PLP) coefficients and Linear Predictive Cepstral (LPC) coefficients.

To record acoustic signal, it has to be sampled and quantized. The resulting sequence of numbers is stored in a defined format like *.wav* or *.ua*. Sampling frequency is usually 8KHz for telephone conversation, common dialogs are often sampled at 16KHz or 22KHz. The quantization is done with 8 or 16 bits per sample. When the signal is recorded, it is ready for feature extraction, so-called *parametrization*. Short time spectrum is computed on slices of length 25ms, Hamming window is applied to get the slices. Slices overlap by 15 ms.

In case of MFCC, the short time spectrum is obtained and then the log is taken and the features are warped according perceptually motivated Mel filter bank. Finally an inverse discrete cosine transformation is taken. Adding singal energy, time derivatives and second order time derivatives completes the MFCC feature representation.

PLP features are based on similar ideas as MFCC. They are also computed from short time spectrum. In contrast to MFCC the use more psychophysically motivated transformation and instead of Mel frequency filter bank they use Bark scale. How to combine advantages of MFCC and PLP features was studied in [*Hönig* et al. (2005)]. A more general introduction written in Czech can be found in [*Psutka* (1995)].

1.5 Acoustic Model

Acoustic model expresses how much an acoustic sequence matches a given word in terms of probability $P(O|W)$. Hidden Markov Models (HMMs)[*Jelinek* (1997)] are predominant learning models in large-vocabulary speech recognition. However other models such as neural networks, support vector machines and other large margin discriminative techniques have been justified on smaller vocabularies, their use on large scale has to be further studied.

HMM are very attractive for their capability of modeling variable-length input sequences and their relatively easy training. An HMM can be viewed as a finite state automaton $A =< S, i, F, A, B >$, where S is a set of states, i is the initial state, F is a set of final states, A is transition probability distribution. $A(i, j)$ assigns probability to the arc leading from state i to state j. A must obey summing rule $\sum_{j \in S} A(i, j) = 1$.

$B(i, o)$ represents output distribution of observation o in state i. Output probability is usually a mixture of Gaussians $f(o)$.

$$f(o) = \frac{1}{(2\pi)^{N/2}|\Sigma|^{1/2}} exp(-\frac{1}{2}(o-\mu)^\top \Sigma^{-1}(o-\mu))$$

$f(o)$ is a multivariate Gaussian distribution with mean μ and covariance matrix Σ. N is the dimension of observation vectors. For fast computation, covariance matrix Σ is approximated by a diagonal matrix, off-diagonal values are set to 0. An example of a simple HMM is depicted on figure 1-1.

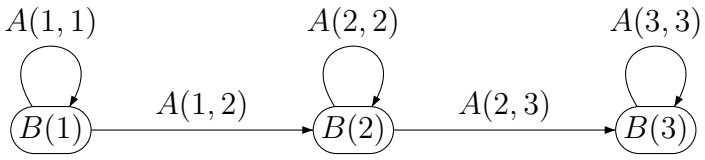


Figure 1-1: HMM example. $A(i, j)$ are transition probabilities from i to j. $B(i)$ represent output probability distribution for state i.

Given an HMM model for a particular word, we may ask a question what is the probability of generating observation sequence O. Sequence O can be generated by any sequence of states starting from state i and ending at F. Probability of O given a particular state sequence s is just a multiplication of transition probabilities $A(i, j)$ and output probabilities $B(i, o)$ along the path.

$$P(s, O|HMM) = \prod_i A(i, i+1)B(i+1, o_{i+1})$$

The total probability $P(O|HMM)$ of generating O is then sum over all paths which generate O. Eventhough the number of paths is exponential in the number of states, thanks to dynamic programming $P(O|HMM)$ can be computed in polynomial time.

$$P(O|HMM) = \sum_s P(s, O|HMM)$$

However for large HMMs, we need faster computation of $P(O|HMM)$. A reasonable

approximation is to take just the most likely path of the model.

$$P(O|HMM) \approx \max_{s} P(s, O|HMM)$$

Such approximation is referred to as *Viterbi*, the hidden assumption is that probability of most likely path contributes to the total probability greatly and therefore we can substitute the Viterbi estimate for the total probability.

A great advantage of HMMs is that they can be easily trained. A variant of expectation-maximization algorithm so-called *Baum-Welsh training* can be used. This estimation is based on maximum likelihood principle.

1.6 Language Model

In recognition of large-vocabulary spontaneous speech, language modeling plays a substantial role. A language model suggests which words are likely to appear one after another and assigns probability to any word sequence. This probability estimate is important for further combining with other components of the speech recognition system. Furthermore, based on the context, it can disambiguate between acoustically similar words or homophones, for example *check* versus *Czech* which can be hardly distinguished from acoustic evidence. Ideally, language modeling encompasses all linguistic knowledge of the language starting from morphology, through syntax up to semantics and pragmatics.

Language modeling is not only essential to speech recognition but also to other research areas concerned with human interaction. Recognition of hand-written texts or automatic machine translation are fields in which language models are employed.

1.7 Decoder

To obtain the word sequence which best matches acoustic observations, a huge space of word sequences must be searched. The Viterbi search mentioned above works

according to recursion

$$P(o_1^{i+1}, s_j) = \max_k P(o_1^i, s_k) A(k, j) B(j, o_{i+1})$$

Probability estimate of the whole sequence of observations o_1^{i+1} ending in state s_j is decomposed into shorter estimate $P(o_1^i, s_k)$, transition weight $A(k, j)$ and output weight $B(j, o_{i+1})$. The idea is that for each state at time i we keep track only of the best predecessor state with its probability estimate. So, for each state at time i we go through all states and search for the best predecessor at time $(i-1)$ and compute $P(o_1^{i+1}, s_j)$. When we are done with the last observation o_T we go through all the states considered at time T and find the one with highest $P(o_T, s_k)$. Starting from the best final state we follow the stored link to the best predecessor, from the the predecessor we follow the stored link to its predecessor an so on. Thus the best sequence of states of the HMM generating observations o_1^T is deciphered. The overall time complexity is $O(TN^2)$ where T is the number of observations and N is the number of HMM states.

This implementation of Viterbi search does not work in real time for large-vocabulary recognition, therefore some speed-up modifications must be introduced. One optimization is to keep only K best states in time i, alternatively we can keep only the states whose probability is competitive with the the best state at time i. The level of competitiveness is given by some threshold. So states whose score

$$P(o_i, s_{best}) - P(o_i, s_j) > threshold$$

are discarded. These modifications are referred to as *Beam search*.

Other graph search techniques have been investigated. A^* *search* well-known from the field of artificial intelligence can be employed or stack decoding as was described in [*Jelinek* (1997)] can be used.

For futher computation acceleration, techniques like fast match are employed. Input observations are quantized, so Gaussians are fed only with discrete values. Therefore, Gaussians evaluation can be precomputed, and thus estimation of prob-

ability of an observation at a given state reduces to a table look up. In addition, states of an HMM (see fig. 1-1) can be merged to only single state. Probability of an observation generated by this single state is defined as maximum over estimates of the three original states. Fast match is intended for rough state elimination, full search is run after fast-match, see [*Huang*, *Acero*, and *Hon* (2001)].

1.8 Text Organization

The text is organized as follows. Chapter 1 introduces the task of speech recognition, gives an overview of the field and motivates the research. In chapter 2 language modeling is described including smoothing, pruning and finite state representation. Specifics of Czech from the point of view of speech recognition are characterized in chapter 5. Chapter 6 specifies the acoustic data and text corpora exploited in experiments. Chapter 7 presents the main contributions of the thesis. The drawn conclusion are summarized in chapter 8.

Chapter 2

Language Modeling

A language model is a crucial part of any large-vocabulary continuous speech recognition system. It aims at predicting the next word given the previous utterance and thus guiding a search procedure throughout a huge number of hypothesis. Formally, a language model can be viewed as a probability distribution $P(w_i|w_1,\ldots,w_{i-1})$, i.e. probability of a word w_i given the previous words $w_1,\ldots,w_{i-1}$, which are often referred to as a *history*. To simplify the notation, $w_1,\ldots,w_{i-1}$ is rewritten as w_1^{i-1}.

The conditional word distribution is estimated from a possibly large text corpus. A straightforward method of distribution estimation is to follow the maximum likelihood (ML) principle and to take the relative frequency of a word given its history as the estimate.

$$P(w_i|w_1^{i-1}) = \frac{C(w_1^i)}{C(w_1^{i-1})} \tag{2.1}$$

$C(w_i^j)$ refers to the number of occurrences of w_i^j in a training corpus.

The direct ML estimate, however, turns out to be unfeasible. First, as sentence length increases, the word sequence w_1^i is observed only few times in any text corpus. This fact is in contrary with the ML presupposition which assumes that frequency of modeled events is high. Thus, the ML estimate for bigger i is unreliable. One possible solution to this problem is to group histories into classes according to some

well-defined criterion ϕ, see [*Manning* and *Schütze* (2000)].

$$P(w_i|w_1^{i-1}) = \frac{C(\phi(w_1^i))}{C(\phi(w_1^{i-1}))} \tag{2.2}$$

This approach will be further discussed in the next section.

Second reason why formula 2.1 is impractical is that language model should assign nonzero probability to any word sequence. Indeed, word sequences of zero probability will never be recognized. But formula 2.1 assigns zero probability to all words w_i which have not been observed immediately after w_1^i. This issue is solved by so-called *smoothing* and will be discussed later on.

2.1 N-grams

The traditional approach to language modeling is to take only a few previous words as the history, so-called *n-grams*. Despite their simplicity, they proved to be very efficient and hard to beat. Much effort is needed to significantly outperform this model. Probability of a word in an n-gram scheme is computed as

$$P(w_i|w_1^{i-1}) = \frac{C(w_{i-n+1}^i)}{C(w_{i-n+1}^{i-1})} \tag{2.3}$$

In practice, n is set to a small number, say between 1 and 5. Of course, this model has to be smoothed to incorporate unseen n-grams.

2.2 Smoothing

A language model has to assign nonzero probability to any word sequence, not only to those observed in a training corpus. Assigning zero probability would result in prohibiting such a sentence from being recognized even though it was actually uttered. Smoothing is a way of redistributing probability mass to unseen events. Chen [*Chen* and *Goodman* (1998)] gives an empirical evaluation of many current smoothing techniques. We will briefly mention the most common smoothing methods.

2.2.1 Laplace Smoothing

Laplace smoothing, sometimes called *add-one* smoothing, is one of the oldest and easiest smoothing techniques. Zero events are avoided by adding 1 to each count. The smoothed probability estimate is then

$$P(w_i|w_{i-n+1}^{i-1}) = \frac{C(w_{i-n+1}^{i}) + 1}{\sum_{w_j}(C(w_{i-n+1}^{i-1}w_j) + 1)} = \frac{C(w_{i-n+1}^{i}) + 1}{C(w_{i-n+1}^{i-1}) + V} \tag{2.4}$$

where V denotes the vocabulary size.

2.2.2 Add-Lambda Smoothing

Laplace smoothing leaves too much of probability space to unseen events. To circumvent overestimation of unseen n-grams, Lidstone's law can be applied. Instead of adding one, some smaller value λ is used.

$$P(w_i|w_{i-n+1}^{i-1}) = \frac{C(w_{i-n+1}^{i}) + \lambda}{\sum_{w_j} C(w_{i-n+1}^{i-1}w_j) + \lambda} = \frac{C(w_{i-n+1}^{i}) + \lambda}{C(w_{i-n+1}^{i-1}) + \lambda V} \tag{2.5}$$

λ has to be tuned to optimize given objective function.

2.2.3 Good-Turing Discounting

Good-Turing [*Good* (1953)] discounting is based on the assumption of binomial distribution of events. Good-Turing takes adjusted counts $r^\star$ to compute probability estimate as

$$P(w_1 \dots w_n) = \frac{r^\star}{N} \tag{2.6}$$

where $r^\star$ is computed as

$$r^\star = (r+1)\frac{E(n_{r+1})}{E(n_r)}$$

n-gram count is denoted as r, n_r refers to the number of n-grams which occurred r times. For example, n_1 is the number of n-grams which appeared once, while n_0 is the number of n-grams which haven't appeared in the corpus at all. $E()$ denotes the

expected value and N is the total number of counts. Note that

$$N = \sum_{r=1}^{\infty} r n_r$$

For infrequent n-grams, $E(n_r)$ can be approximated by n_r since many n-grams occur only a few times resulting in high n_r. For very frequent n-grams, it may happen that $n_{r-1} = 0$ even if $n_r > 0$. [*Gale* and *Sampson* (1995)] have proposed a simple and effective algorithm for smoothing of n_r. That counts are properly normalized in 2.6 can be readily checked.

$$N = \sum_{r=1}^{\infty} r n_r = \sum_{r=0}^{\infty} (r+1) n_{r+1} = \sum_{r=0}^{\infty} r^{\star} n_r$$

Probability mass left to unseen n-grams, n_1/N, is distributed over them uniformly.

2.2.4 Katz Smoothing

Katz [*Katz* (1987)] extends Good-Turing estimates by treating large counts as reliable estimates. Only counts smaller than a predefined threshold k are discounted and thus saving some probability mass for unseen n-grams.

$$r^{\star} = \frac{(r+1)\frac{n_{r+1}}{n_r} - r\frac{(k+1)n_{k+1}}{n_1}}{1 - \frac{(k+1)n_{k+1}}{n_1}} \qquad \text{for } 1 \leq r \leq k \tag{2.7}$$

Furthermore, [*Katz* (1987)] suggests to use statistics of shorter n-grams for unseen events. For example, if a trigram $w_1 w_2 w_3$ has zero count, conditional likelihood is estimated from the bigram $w_2 w_3$. Conditional probability P_{BO} is computed as

$$P_{BO}(w_i|w_{i-n+1}^{i-1}) = \begin{cases} P_{ML}(w_i|w_{i-n+1}^{i-1}) & \text{for } r > k \\ P_{GT}(w_i|w_{i-n+1}^{i-1}) & \text{for } 1 \leq r < k \\ \beta(w_{i-n+1}^{i-1}) P_{BO}(w_i|w_{i-n+2}^{i-1}) & \text{for } r = 0 \end{cases} \tag{2.8}$$

P_{ML} refers to the maximum likelihood estimate, P_{GT} stands for the discounted estimate derived from 2.7, the back-off weight $\beta(w_{i-n+1}^{i-1})$ assures that probabilities add

up to one.

2.2.5 Jelinek-Mercer Deleted Interpolation

[*Jelinek* and *Mercer* (1980)] linearly interpolates higher order n-grams with lower order n-grams. Higher order n-grams are sparser and thus their estimates are less reliable. Interpolation with lower order n-grams can yield better models. Interpolated estimate P_{LI} is computed from maximum likelihood estimates P_{ML} as

$$P_{LI}(w_i|w_{i-n+1}^{i-1}) = \lambda_0 \frac{1}{V} + \lambda_1 \frac{c(w_i)}{\sum_j c(w_j)} + \sum_{k=2}^{n} \lambda(w_{i-k+1}^{i-1}) P_{ML}(w_i|w_{i-k+1}^{i-1}) \tag{2.9}$$

Interpolation weights, so-called *lambdas*, must sum up to 1. In general, it is useful to group lambdas into bins according to some tying criterion. [*Bahl*, *Jelinek*, and *Mercer* (1983)] suggests partitioning according to $C(w_{i-n+1}^{i-1})$. This idea is further extended by [*Chen* and *Goodman* (1996)]. Lambdas can be directly calculated by the expectation maximization algorithm [*Dempster*, *Laird*, and *Rubin* (1977)] which is run on so-called *held-out data*. Held-out data should be as similar as possible to the test data and must be different from the corpus on which P_{ML} was calculated. If estimation of lambas is run on train data instead of held-out data, lambda of the longest n-gram would be 1, the other lambdas would be zero.

2.2.6 Whitten-Bell Smoothing

Whitten-Bell smoothing [*Bell*, *Cleaqry*, and *Witten* (1990)], as well as Jelinek-Mercer smoothing, interpolates maximum-likelihood estimates of n-grams of all orders. The probability of Whitten-Bell smoothing P_{WB} can be recursively computed as

$$P_{WB}(w_i|w_{i-n+1}^{i-1}) = \lambda(w_{i-n+1}^{i-1}) P_{ML}(w_i|w_{i-n+1}^{i-1}) + (1 - \lambda(w_{i-n+1}^{i-1})) P_{WB}(w_i|w_{i-n+2}^{i-1}) \tag{2.10}$$

It differs from Jelinek-Mercer in computation of the interpolation weights. It is based on the number of unique words that follow the history denoted as $N_{1+}(w_{i-n+1}^{i-1}\bullet)$

$$N_{1+}(w_{i-n+1}^{i-1}\bullet) = |w_i : c(w_{i-n+1}^{i-1}w_i) > 0| \tag{2.11}$$

The smoothing coefficient of Whitten-Bell is defined as

$$1 - \lambda(w_{i-n+1}^{i-1}) = \frac{N_{1+}(w_{i-n+1}^{i-1}\bullet)}{N_{1+}(w_{i-n+1}^{i-1}\bullet) + \sum_{w_i} c(w_{i-n+1}^{i})} \tag{2.12}$$

2.2.7 Absolute Discounting

In absolute discounting, counts of high-order n-grams are discounted by a fixed value D. The saved probability mass is then uniformly redistributed over unseen n-grams. Kneser-Ney smoothing [*Ney*, *Essen*, and *Kneser* (1994)] is an implementation of absolute discounting suggesting to set

$$D = \frac{n_1}{n_1 + 2n_2}$$

where n_1 and n_2 denote the total number of n-grams with exactly one or two counts, respectively. Lambdas are set in such a way that probabilities sum up to 1.

$$P_{KN}(w_i|w_{i-n+1}^{i-1}) = \frac{\max\{C(w_{i-n+1}^{i}) - D, 0\}}{\sum_{w_i} C(w_{i-n+1}^{i})} + (1 - \lambda(w_{i-n+1}^{i-1}))P_{KN}(w_i|w_{i-n+2}^{i-1}) \tag{2.13}$$

2.2.8 Modified Kneser-Kney Smoothing

[*Chen* and *Goodman* (1998)] extend Kneser-Kney smoothing by making constant D dependent on n-gram count.

$$P_{MKN}(w_i|w_{i-n+1}^{i-1}) = \frac{\max\{C(w_{i-n+1}^{i}) - D(C(w_{i-n+1}^{i})), 0\}}{\sum_{w_i} C(w_{i-n+1}^{i})} + (1 - \lambda(w_{i-n+1}^{i-1}))P_{MKN}(w_i|w_{i-n+2}^{i-1}) \tag{2.14}$$

$$D(C) = \begin{cases} 0 & \text{if} \quad C = 0 \\ 1 - 2Y\frac{n_2}{n_1} & \text{if} \quad C = 1 \\ 2 - 3Y\frac{n_3}{n_2} & \text{if} \quad C = 2 \\ 3 - 4Y\frac{n_4}{n_3} & \text{if} \quad C \geq 3 \end{cases} \tag{2.15}$$

$$Y = n_1 \frac{n_1}{2n_2}$$

where n_i denotes the total number of n-grams occurring exactly i times. Authors experimentally justified that this method yields superior performance over above mentioned methods.

2.3 Language Model Evaluation

In speech recognition, we are mainly interested in the *word error rate (WER)*. However, other objective functions might be desired in application which the speech recognition system is only a part of. In spoken document classification systems, for instance, only key words are substantial, therefore accuracy of key words is measured instead.

WER is defined as the number of mis-recognized words divided by length of the corresponding transcriptions. Let $w = w_1, \ldots, w_n$ denote the sequence of words which have been uttered and let $\widehat{w} = \widehat{w}_1, \ldots, \widehat{w}_m$ denote the recognized sequence. An optimal alignment can be computed via dynamic programming. The optimal alignment is an alignment with the lowest number of insertions i, deletions d and substitutions s. WER is then calculated as

$$\text{WER} = \frac{i + s + d}{n}$$

in the optimal alignment. WER reports how bad we are at decoding an utterance. A measure which takes a positive perspective, *accuracy (ACC)*, is computed as

$$ACC = 1 - \text{WER}$$

WER is an ultimate measure of the performance of a speech recognition system as a whole.

A speech recognition system can be designed to work in several stages. In the first stage a rough recognition is carried out, while in later stages fine-tuned complex models are employed. The output of the first stage is usually a big set of hypotheses which are the subject of further processing. The generated set of hypotheses can be either represented as a list of sentences or as a more effective structure so-called *lattice*. A lattice a directed acyclic graph which have one starting node and one final node, arcs are label with words. Each path through a lattice starting in the starting node and ending in the final node corresponds to a hypothesis. To evaluate the first stage of the recognition process, *oracle accuracy* is often calculated. The oracle accuracy is defined as the accuracy of the hypothesis which has the highest accuracy among all hypotheses stored in the lattices. Note that neither LM scores nor acoustic model scores are taken in account.

$$oracle\_acc(lattice) = \max\{acc(hyp_i)|hyp_i \in lattice\}$$

For fast language model comparison and tuning, *perplexity* is widely used as the objective function. Perplexity comes from the field of information theory [*Cover* and *Thomas* (1991)] and is closely related to the notion of *entropy*. Let X be a discrete random variable with underlying probability distribution p and let X be the sample space from which the data are drawn. Entropy of $H(X)$ is defined as

$$H(X) = -\sum_{\mathsf{x}\in\mathsf{X}} p(X = \mathsf{x}) \log_2 P(X = \mathsf{x})$$

Roughly speaking, entropy measures uncertainty of a random variable. It attains minimum at deterministic distributions. By a deterministic distribution is meant a distribution which has one sure event x, $p(X = \mathsf{x}) = 1$, and other events $\mathsf{y_i}$, $p(X = \mathsf{y_i}) = 0$, impossible. Indeed, there is no uncertainty of outcome in such distributions. Distributions which maximize entropy are uniform distributions. All events drawn

from a uniform distribution are equally likely therefore it's very difficult to predict which event will be drawn next.

Cross-entropy, $H_C(p; p_M)$, of the true probability distribution p and model probability distribution p_M measures the average uncertainty if p_M is used instead of the true probability p. Cross-entropy, $H_C(p; p_M)$ is computed as

$$H_C(p; p_M) = -\sum_{\mathsf{x}} p(\mathsf{x}) \log_2 p_M(\mathsf{x})$$

In context of language modeling, cross-entropy is evaluated as

$$H_C(p_M) = -\sum_{i=1}^{N} \frac{1}{N} \log_2 p_M(w_i | w_1^{i-1})$$

where i runs through the text corpus of length N. A more intuitive measure, *perplexity (PP)*, is often reported in the literature. A language model of perplexity x is equivalent to a uniform language model on a vocabulary of size x. In other words, for a language models of perplexity x, the average number of equiprobable words which a decoder has to choose from is x.

$$PP = 2^{H_C(p_M)}$$

Perplexity is a measure based on a text corpus only and therefore neglects acoustic confusability of words. Thus optimizing perplexity may not lead to reduction in terms of word error rate. Correlation of word error rate with accuracy has been investigated by many authors. [*Chen*, *Beeferman*, and *Rosenfeld* (1998)] [*Jelinek* (1990)]

2.4 Finite-State Representation

An n-gram language model can be nicely represented in a finite-state framework. Such representation is advantageous for future LM combination with the lexicon and acoustic models. Use of finite-state machinery in natural language processing is described in detail in [*Mohri*, *Pereira*, and *Riley* (2002)].

A weighted finite-state automaton is defined as a tuple $A = (\Sigma, Q, E, i, F, \lambda, \rho)$ where Σ is an alphabet, which includes the empty symbol ϵ, Q is a set of states, i denotes the initial state with the initial weight λ, F is the set of final states and ρ is a weight function, which assigns a weight to each final state. E refers to a set of arcs. An arc $e = (p, n, s, w)$ starts in the state p and ends in the state n, accepts symbol s with weight w. In addition, an arithmetics on weights has to specified. Two operation are needed. One to combine weights along path $\otimes$, another to combine paths together $\oplus$. Formally, these operation must have the property of semi-ring. The most common semi-rings in natural language processing are the so-called *tropical* and *probabilistic* semi-ring. In the tropical semi-ring, weights represent negative log probabilities. Weights along a path are summed ($\otimes = +$), and path weights are recombined by taking the minimum ($\oplus = min$), which corresponds to the Viterbi decoding. In the probabilistic semi-ring, weights represent actual probabilities, which are multiplied along a path ($\otimes = \cdot$), paths are recombined by taking the sum ($\oplus = +$). Figure 2-1

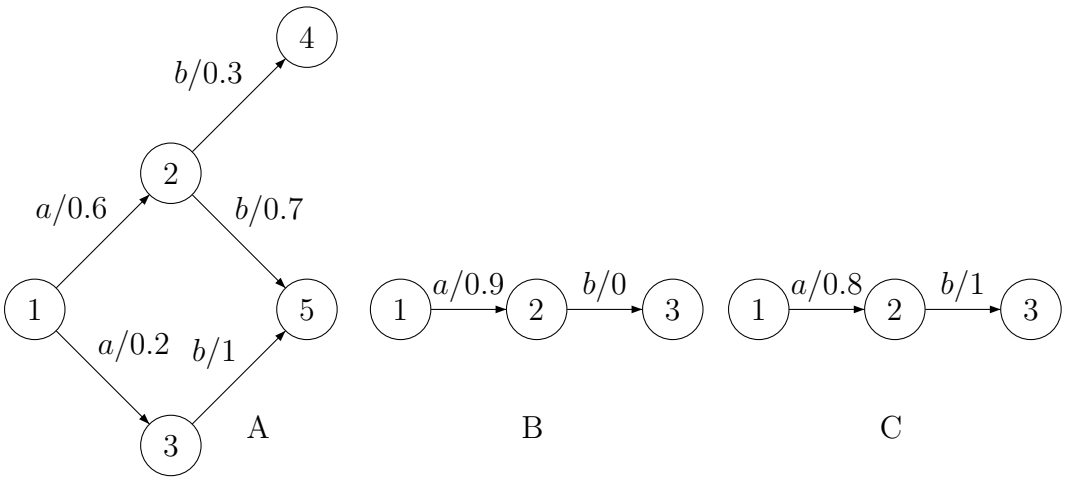


Figure 2-1: An example of finite state automata. A is equivalent with B under tropical semi-ring. Under probabilistic semi-ring A is equivalent with C.

displays an example of a weighted automaton A. If tropical semi-ring is considered, automaton B is equivalent with A. Under probabilistic semi-ring, automaton C is equivalent with A.

A finite-state representation of an n-gram model is straightforward. An example of a trigram language model representation is displayed in Figure 2.4. The alphabet

contains the vocabulary and the empty word ϵ. Weights are negative log probabilities. Nodes represent history. By taking an arc, the history can be extended by one word, keep the same length or shortened by one word. Consider the node representing history w_1. There might be an arc leading to the node representing history w_1, w_2. Such an arc would be then labeled with the word w_2 and associated weight $-\log(P(w_2|w_1)$. In this case the history is prolonged. From the node w_1, w_2 it is possible to back off to the node w_2 by taking the arc labeled with ϵ with weight $\beta(w_1, w_2)$, which is the back-off factor of formula 2.8. Arcs which do not extend history length lead among nodes which represent history of length $n-1$. There is an arc leading from node w_1, w_2 to node w_2, w_2 labeled with with word w_2 and probability $-\log(P(w_2|w_1, w_2)$. There are no arcs among back-off states which would keep the history length.

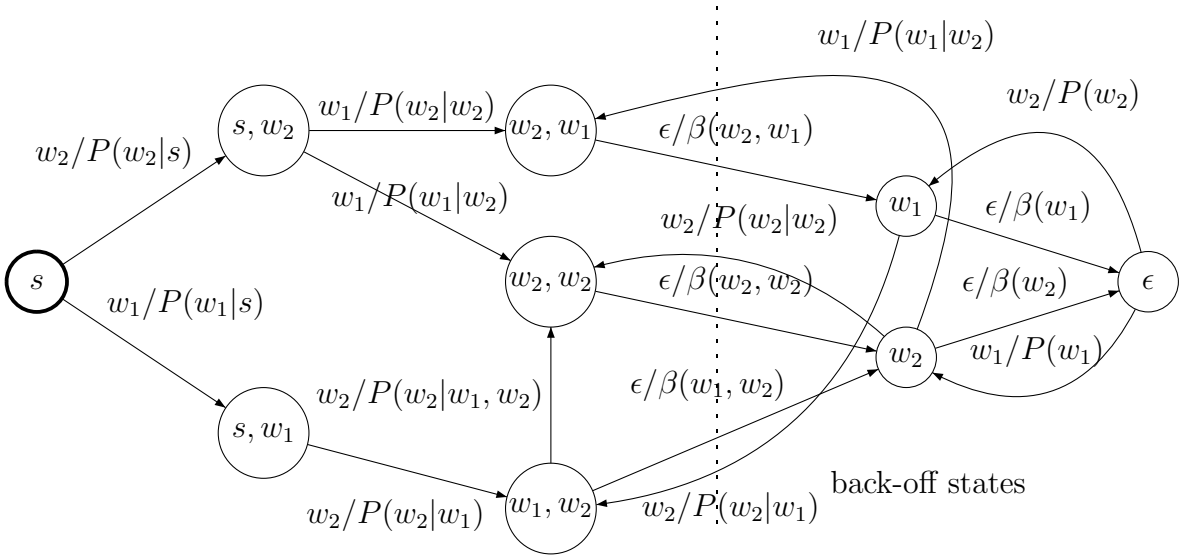


Figure 2-2: A fragment of a trigram language model, s represents the sentence start. Node labels denote implicit representation of history. To simplify the picture, weights are probabilities instead of minus log probabilities.

Not only language models, but also pronunciation dictionaries can be represented in the finite state framework. Weighted finite state transducers (WFST) are the automata which can capture mappings of one regular language on another. WFST $T = (\Sigma, \Delta, Q, E, i, F, \lambda, \rho)$ is defined as 8-tuple, Σ is the input alphabet, Δ is the output alphabet, Q is the set of states, i is the initial state, F is the set of final

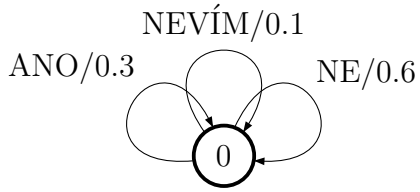


Figure 2-3: An example of a simple unigram language model.

states. λ is a function which assign some weight to the initial state, ρ is a function which assigns some weight to the final states and can also output symbols of Δ. An arc $e = (p, n, i, o, w)$ of E leads from state p to state n, inputs symbol $i \in \Sigma$, outputs symbol $o \in \Delta$ with weight w. As in the case of WFA, a semi-ring has to be specified.

An example of a lexicon represented as a WFST is depicted on Figure 2-4.

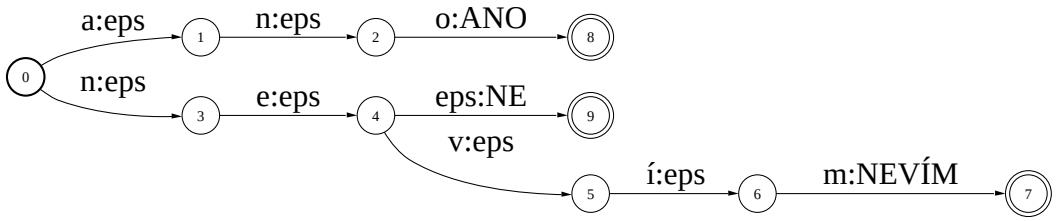


Figure 2-4: Example of a pronunciation lexicon represented as a finite state transducer.

2.4.1 Automata Combination

One of the main advantages of finite state representation is that models of different levels can be combined. Operation which combines FSTs is called composition and can be deemed an extension of classical FSA intersection algorithm. Consider two WFST A and B. Let A map word u on v with weight c_1, and B map word v on w with weight c_1. The composition $C = A \circ B$ of A an B maps word u directly on v with weight $c_1 \otimes c_2$.

A pronunciation lexicon L can be thus composed with language model G. Figure 2.4.1 displays a simple example of such composition. We will refer to the obtained transducer as LG.

LG can be further composed with more fine-grained automata such as a mapping from phones to triphones or a mapping from triphones to HMM states. There are

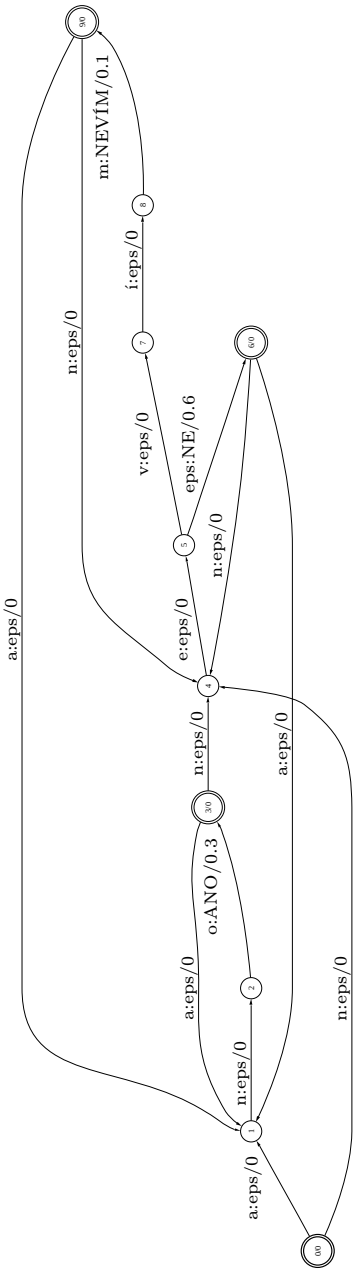


Figure 2-5: Optimized composition of lexicon displayed on Fig 2-4 and language model depicted on Figure 2-3.

techniques to optimize these transducers, e.g. determinization, minimization, weight pushing an so on, aiming at reducing automaton size and shifting the weights, see [*Mohri*, *Pereira*, and *Riley* (1996)].

2.5 Language Model Pruning

As corpora suitable for language modeling grow large, memory consumption of LM parameters increases beyond practical limits. To exploit all the available data for estimation of a LM of feasible size, a pruning method is needed.

2.5.1 Cutoff Method

The cutoff method is the simplest method for reducing LM size. It is based on n-gram frequency. For each n a threshold is set. If an n-gram has lower frequency than the predefined threshold, it is discarded. This method follows the intuition that infrequent n-grams are the least likely to be observed in the test data, in addition to the fact that infrequent n-grams are not reliably estimated.

2.5.2 Weighted Difference

Weighted difference is a pruning method proposed by K. Seymore [*Seymore* and *Rosenfeld* (1996)]. This pruning technique aims at removing arcs from the model in such a way that the LM probability estimate would be as little affected as possible. If the probability of an n-gram arc is very close to the back-off $(n-1)$-gram estimate, there is no need to store the n-gram arc. The word sequence can be produced through the back-off state with negligible influence on performance.

The pruning algorithm proceeds in two steps. Arcs are sorted by so-called weighted difference factor

$$wdf = K \cdot (\log(\text{original prob}) - \log(\text{back-off prob}))$$

where K denotes the Good-Turing discounted n-gram count. Arcs which have wdf below certain threshold are discarded. The threshold is set in such a way that the reduced LM meets the memory requirements.

Seymore reported slightly better performance in terms of word error rate as compared to the cut-off method.

2.5.3 Entropy-Based Pruning

Stolcke [*Stolcke* (1998)] suggested a more principal method for language model pruning. Let p denote the original LM and p' the pruned LM. His approach searches for the p' such that $D(p||p')$ is minimized. $D(\cdot||\cdot)$ refers to the Kullback-Leibler distance, which is defined as

$$D(p||p') = -\sum_{w_i,h_j} (p(w_i,h_j)[\log\ p'(w_i|h_j) - \log\ p(w_i|h_j)])$$

As in the Weighted Difference Method, pruning works in a greedy fashion. First, arcs are sorted according to the criterion. Then a threshold is set and arcs which fall below the threshold are pruned out.

In addition to arc removal, the Entropy-Based pruning recomputes the back-off weights.

Experimental results showed that there is great overlap between the n-grams selected by Weighted Difference method and the n-grams chosen by the Entropy-Based pruning. In terms of WER, both methods perform about the same.

Chapter 3

Advanced Models

3.1 Neural Networks

Neural networks have been tested as a novel approach to language modeling. Schwenk [*Schwenk* and *Gauvain* (2004)] worked with fully connected multi-layer perceptron, that is a forward network without any loops, see 3.1. On the input layer, words are represented as boolean vectors. Word w_i from a dictionary of size V is represented as a vector of length V. All cells of the vector are set to 0 except for the i-th cell which set to 1. A more compact representation is obtained by projecting these vectors to P dimensional space, where P is chosen much smaller than V. The projection is a simple multiplication of the input vector by a projection matrix. Size of a projection matrix is $V \times P$. So for history of length k we have $k \cdot P$ input values (instead of $k \cdot P$) and k projection matrices.

Value of hidden state d_j is computed as tanh applied to linear combination of its inputs, that is

$$d_j = \tanh(\mathbf{M_j} \cdot \mathbf{c} + \mathbf{b_j})$$

where $\mathbf{M_j}$ is a combination matrix related to the state j, $\mathbf{b_j}$ is a shift vector related to the state j and $\mathbf{c}$ is vector of the output values of the previous layer (note that we work with a fully connected multilayer network.)

On the output layer the softmax is taken.

$$P(w_i|history) = \frac{\exp(d_i)}{\sum_{j \in outputstates} \exp(d_j)}$$

Thus the probability of word w_i is estimated. The number of the output states is of the dictionary size V. So the network computes the conditional probability estimate of each word in a single run.

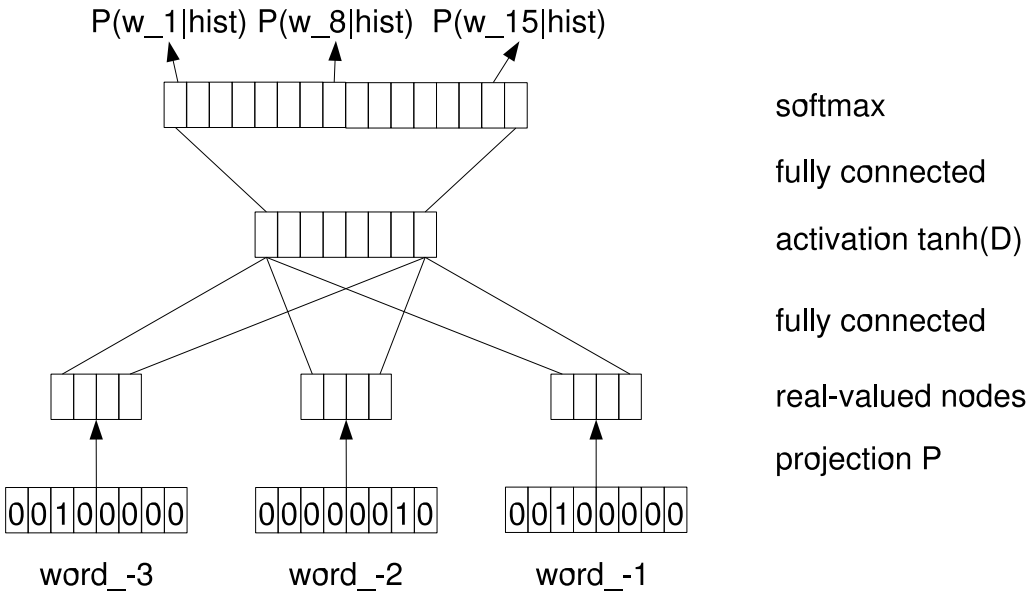


Figure 3-1: Neural network as a language model. A forward multilayer fully-connected network.

To train the network, correct output values for given inputs must be provided. A training corpus comes in handy. For each n-gram w_i^{i+k+1}, history w_i^{i+k} is given to the network input, the correct output is set to be 1 for the node representing w_{i+k+1}, and 0 for the other output nodes. Neural networks can be trained by the *back-propagation algorithm*, which is a gradient method. More sophisticated methods can be used for training as well.

To make the whole approach computationally feasible, optimizations have to be introduced. The size of the output layer can be shrunk from V to some moderate value referred to as a *shortlist*. The authors shrunk the output layer to 2000 nodes,

which covered 89% of probability estimates requested during testing. For the words whose probability is not estimated via the network, standard back-off model was used. So the probability of word w_j was computed as

$$P(w_j|h_j) = \begin{cases} P_{NN}(w_j|h_j)P_S(h_j) & \text{if } w_j \in \text{shortlist} \\ P_B(w_j|h_j) & \text{else} \end{cases}$$
$$P_S(h_j) = \sum_{w \in shortlist} P_B(w|h_j)$$

$P_{NN}()$ refers to the neural net probability estimate, $P_B()$ is the standard back-off estimate. $P_S()$ a normalization factor which ensures that P() is a proper probability distribution. In fact neural network redistributes probability mass assigned by the back-off model to the words from the shortlist.

The authors have tested this approach on NIST R03 evaluation data, which are transcriptions of conversational telephone speech in English, see `http://www.nist.gov/speech` for detailed data description. The best result they achieved in terms of accuracy was a reduction from 22% to 21.5%, relative perplexity reduction was about 9%.

Neural networks seem to be appealing from the theoretical point of view, nevertheless computational issues restrict their modeling power. Only a moderate number of words can have its probability estimate computed. In addition, current implementations of neural network models are capable of handling histories of only about the same size as standard n-grams. To summarize, we think that this model has to be further improved to be worth employing in real-life applications.

3.2 Latent Semantic Analysis

Latent semantic analysis (LSA) was motivated by information retrieval approach to word modeling. Words are viewed as points in continuous space of large dimension. On a continuous space distances are naturally defined, standard transformations can be performed and advanced techniques from other research fields can be adopted.

LSA requires that training corpus is segmented into documents. A document,

understood as a set of sentences, should be semantically homogeneous, convey a single idea. Assume we have N such documents, and an inventory of M words which occurred in the documents. LSA builds a huge matrix A of dimension $M \times N$. A cell $a_{i,j}$ represents how word w_i relates to document c_j. Many functions have been studied in the literature, one the most appropriate values for $a_{i,j}$ is

$$a_{i,j} = (1 - \epsilon_i)\frac{\kappa_{i,j}}{\lambda_j}$$

$\kappa_{i,j}$ is the number of times word w_i occurs in document c_j, λ_j is the total number of words in the document c_j, and ϵ_i denotes the normalized entropy of word w_j over the collection of documents

$$\epsilon_i = -\frac{1}{logN}\sum_{j=1}^{N}\frac{\kappa_{i,j}}{\tau_i}\log\frac{\kappa_{i,j}}{\tau_i}$$

where $\tau_i = \sum_j \kappa_{i,j}$.

$a_{i,j}$ can be understood as a weighted relative frequency of word w_i in document c_j. The weight measures how important the word w_i is. If word w_i is very descriptive, is observed in only in a few documents with high frequency, the entropy ϵ_i is small, and thus the weight factor $(1 - \epsilon_i)$ is close to 1, so the $a_{i,j}$ is high. However, if the word is frequently observed in all documents, its normalized entropy would be close to 1, so the weight $(1 - \epsilon_i)$ would be close to zero. To give an example, function words like prepositions, conjunctions would get low $a_{i,j}$ even if they are frequent. Topic-specific words, say 'algebra' or 'logic', should get high score if they are sufficiently observed.

In the next step, matrix A is decomposed into three matrices. The singular value decomposition (SVD) is used.

$$A = USV^T$$

Matrix U is of dimension $M \times R$, V^T is of dimension $R \times N$. Both U and V^T are ortho-normal $U^TU = V^TV = I$. S is diagonal matrix $(R \times R)$, off-diagonal cells are zero, diagonal values are eigen values of matrix A sorted in descending order. R is rank of matrix A.

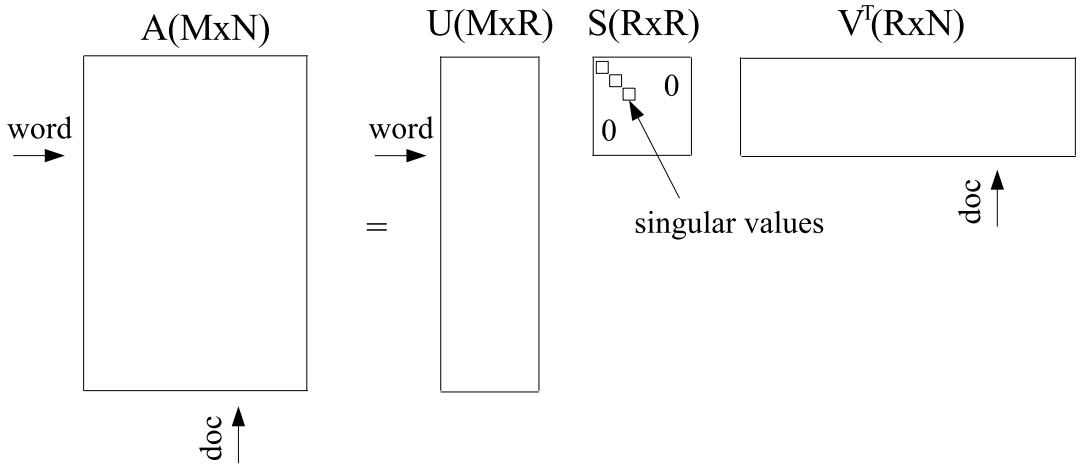


Figure 3-2: Decomposition of word-document co-occurrence matrix A.

In the original matrix A, words were represented as horizontal vectors w_i, the documents were represented as vertical vectors c_j. Having done SVD, the representation of words w_i and documents c_j is transformed

$$\begin{aligned} w_i &\rightarrow \bar{u_i} = u_i S \\ c_j &\rightarrow \bar{v_j} = v_j S \end{aligned}$$

u_i vectors are row vectors of U, v_j are column vectors of V^T. The dimension of new vectors is R. SVD can be optimized such that not all singular values would be present in S. Only the biggest eigen values are kept, the small ones are discard. This optimization drastically reduces size (R) of the new space. Representation of words and documents is thus more compact. The step of selecting only biggest eigen values is referred to as *dimensionality reduction*.

Having a unified representation for words and documents, we can define distance of word from a document. Cosine distance has been appreciated in information retrieval.

$$K(w_i, c_j) = \frac{u_i S v_j^T}{\|u_i S^{1/2}\| \|v_j S^{1/2}\|}$$

$K(w_i, c_j)$ is close 1 when word is relevant for document c_j. If there is no correspondence between word w_i and document c_j, $K(w_i, c_j)$ should be close to zero.

In terms of language modeling, a document is just the word history. It may happen that a particular word history was not observed as a document in training data. Distance from unobserved documents is not defined. Therefore we need to add the document representing the history into matrix V^T. Fortunately, there is an easy way to expand it on the fly. So the distance of word from even unobserved document can be computed.

Cosine distance can be included into standard language models. [*Bellegarda* (2005)] suggested to use

$$P(w_i|w_1^i) = \frac{P_{n-gram}(w_i|w_1^i)K(w_i, c_i)}{P_{n-gram}(w_k|w_1^i)K(w_k, c_i)}$$

where $P_{n-gram}(\cdot)$ refers to a standard n-gram model, c_i represents a document composed of history of word W_i.

LSA is very attractive from the theoretical point of view. The prediction made by LSA can be based on long word history. As a bag-of-words model, it neglects syntax since the word order in the history is not reflected in the model. Nevertheless, it attempts to model words with similar meaning to the history. In practice, no dramatic WER reduction has been observed, see [*Gildea* and *Hofmann* (1999)].

3.3 Structured Language Model

Structured language model (SLM) builds on linguistic understanding of language, it embraces morphology and syntax in a unified statistical framework. SLM can be understood as a generalization of n-gram model. The key difference from the n-gram model is that prediction is not necessarily driven by immediately preceding words but in addition so-called *heads* of preceding phrases are considered. SLM constructs parse

trees upon which prediction of the next word is made. To give a simple illustration of the SLM motivation consider sentence *The cat I told you about died.* Bigram language model would estimate probability of *died* given word *about* whereas SLM would consider word *cat* as the history since it is the head of the preceding phrase.

Put formally, joint probability of word sequence W and parse tree T is computed as

$$\begin{aligned} P(W,T) &= \prod_{k=1}^{n+1}[P(w_k|W_{k-1}T_{k-1}) \cdot P(t_k|W_{k-1}T_{k-1}, w_k) \cdot \\ &\quad \prod_{i=1}^{N_k} P(p_i^k|W_{k-1}T_{k-1}, w_k, t_k, p_1^k, \ldots, p_{i-1}^k)] \end{aligned}$$

Conditional word probability is then

$$\begin{aligned} P_{SLM}(w_{k+1}|W_k) &= \sum_{T_k \in S_k} P(w_{k+1}|W_k T_k) \cdot \rho(W_k T_k) \\ \rho(W_k T_k) &= P(W_k T_k) / \sum_{T_k \in S_k} P(W_k T_k) \end{aligned}$$

where $\rho(W_k T_k)$ assures proper normalization.

All train and test sentences used by SLM are augmented by start symbol ($w_0 = \langle start \rangle$) and end symbol ($w_{n+1} = \langle end \rangle$.) $W_{k-1}T_{k-1}$ refers to word-parse (k-1) prefix, w_k is the word being predicted, t_k is the tag being predicted, N_k is the number of operations of parser, referred to as *constructor*, has to make at position k, $p_1^k, \ldots, p_{N_k}^k$ are the operations done by *constructor* at position k.

SLM decomposes into three basic probability distributions

$$P(w_k|W_{k-1}T_{k-1}) = P(w_k|h_0, h_{-1})$$

$$P(t_k|w_k, W_{k-1}T_{k-1}) = P(t_k|w_k, h_0, h_{-1})$$

$$P(p_i^k|W_k T_k) = P(p_i^k|h_0, h_{-1})$$

where h_0, h_{-1} are preceding head words generated by *constructor*. Each of these distributions is estimated by deleted interpolation. The model first generates word w_k given preceding head words. This step is referred to as *word-predictor* move. Note that preceding head words may coincide with immediately preceding words. Then tag t_k is guessed given current word w_k and preceding head words. This step is referred to as *tagger* move. Finally partial parse trees are extended by *constructor* move.

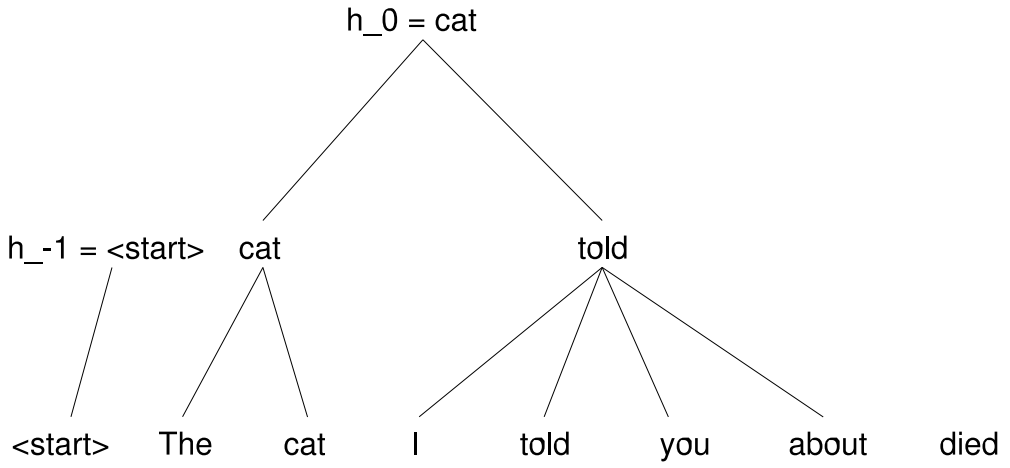


Figure 3-3: Structured language model

Several issues have to be resolved to make SLM approach viable. The most daunting part is the constructor. Number of parses grows exponentially with sentence length. To cope with such tremendous number of trees, authors suggested using multi-stack search algorithm, a concept similar to beam search. Due to memory limitations pruning of unpromising trees must be employed too.

Grammar used for *constructor* is a context-free grammar. The grammar has to be binarized, trees get a bit deeper, but the branching factor is limited to at most two children per node. Headwords and nonterminal label are assigned to inner tree nodes. Some experiments on binarization and head-word percolations are described in [*Xu*, *Chelba*, and *Jelinek* (2001)].

It has been demonstrated that SLM by itself is competitive with standard n-gram. However when the models get interpolated the result outperforms individual models.

$$P(w|hist) = \lambda \cdot P_{SLM}(w|hist) + (1 - \lambda) \cdot P_{n-gram}(w|hist)$$

SLM was used for rescoring n-best lists from first-pass recognition of WSJ DARPA'93 HUB1 data. The baseline model was 3-gram estimated on 50M words of WSJ, vocabulary size was 20K words. The best achieved WER reduction was 10% relative from 87% accuracy.

SLM is a promissing well-motivated approach. It can be potentially used in the first-pass recognition as its prediction of next word is based only on the preceding words and built structure. The major drawback is its complexity. Running SLM on long sentences is still problematic, more optimization tricks are needed.

3.4 Discriminative Syntactic Language Model

Collins [*Collins*, *Saraclar*, and *Roark* (2005)] suggested to use syntax for language modeling in a discriminative framework. The model is used for re-scoring sentences generated by first-pass. First, each sentence from the n-best list is parsed. Then, based on the parse and sentence words a new score is computed. The sentence of highest score is labeled as the best hypothesis. The new best sentence is selected as follows

$$W^* = \arg\max_W (\beta \log P_{LM}(W) + \log P_{AM}(A|W) + \langle \bar{\alpha}, \Phi(A, W) \rangle)$$

where $\log P_{LM}(W)$ is the baseline n-gram language model, β is language model scaling factor, $\log P_{AM}(A|W)$ is acoustic score, $\langle \bar{\alpha}, \Phi(A, W) \rangle$ refers to the new discriminative model. Notation $\langle \cdot, \cdot \rangle$ is used for the dot product. $\bar{\alpha}$ is a real-valued vector of weights, which are learned from a corpus. $\Phi(A, W)$ represents feature vector. A feature can be for example indication whether a particular grammar rule is present in parse tree, or whether a particular bigram is present in sentence. To be more specific, feature

vector at position say 10 is set to one if sentence parse includes rule $VP \rightarrow VB\ NP$, if such a rule does not appear in the parse the feature value is set to zero. This model reduces to n-gram model if all the features indicate presence of words only. To simplify notation, formula 3.4 can be rewritten as

$$W^* = \arg\max_W \langle \bar{\alpha}, \Phi(A, W) \rangle$$

where score from the decoder can be set as the first value of feature vector

$$\Phi_1(A, W) = \beta \log P_{LM}(W) + \log P_{AM}(A|W)$$

and its weight $\bar{\alpha_1} = 1$.

To fully specify this model, features have to be enumerated and the training algorithm for estimation of $\bar{\alpha}$ must be described. Lets concentrate on the latter. $\bar{\alpha}$ can be estimated via so-called *perceptron algorithm*. Perceptron algorithm iteratively goes through the training data. In each iteration (epoch) sentence by sentence is evaluated. For each sentence (acoustic signal) A_i an n-best list is generated, denoted as GEN(A_i). Among the sentences from GEN(A_i), two candidates deserve extra attention, candidate with lowest WER, Y_i, which we aim for and candidate which is suggested as best by the current model $\hat{W}_i = \arg\max_{W \in \mathrm{GEN}(A_i)} \langle \alpha, \Phi(A_i, W) \rangle$. If the two candidate sentences differ $Y_i \neq \hat{W}_i$, weight vector α is updated. If the two candidate sentences are equal, no update is made. Such behavior is often called ultraconservative.

The algorithm is run until convergence is reached, or till changes in vector α are small. Updates of α can be either done after each training sentence or after each epoch. In the end, the final value of $\bar{\alpha}$ is computed as average over all intermediate values of α as updated during training. Lets denote α_{ti} the value of α after i-th sentence in t-th epoch, $\bar{\alpha}$ is computed as

$$\bar{\alpha} = \frac{1}{TD} \sum_{i,t} \alpha_{it}$$

Algorithm 1 Perceptron

$\alpha = 0$
for $t = 1 \dots T$ **do** ▷ *T number of iterations*
  for $i = 1 \dots D$ **do** ▷ *D number of training sentences*
    $\widehat{W_i} = \arg\max_{W_i \in \mathrm{GEN}(A_i)} \langle \alpha, \Phi(A_i, W_i) \rangle$ ▷ *Best sentence according to model*
    $Y_i = \arg\min_{W_i \in \mathrm{GEN}(A_i)} \mathrm{WER}(W_i)$ ▷ *Best sentence according to transcriptions*
    if $\widehat{W_i} \neq Y_i$ **then**
      $\alpha = \alpha - \Phi(A_i, \widehat{W_i}) + \Phi(A_i, Y_i)$
    end if
  end for
end for

Averaging of parameters yields better performance. This fact has been experimentally verified in several papers.

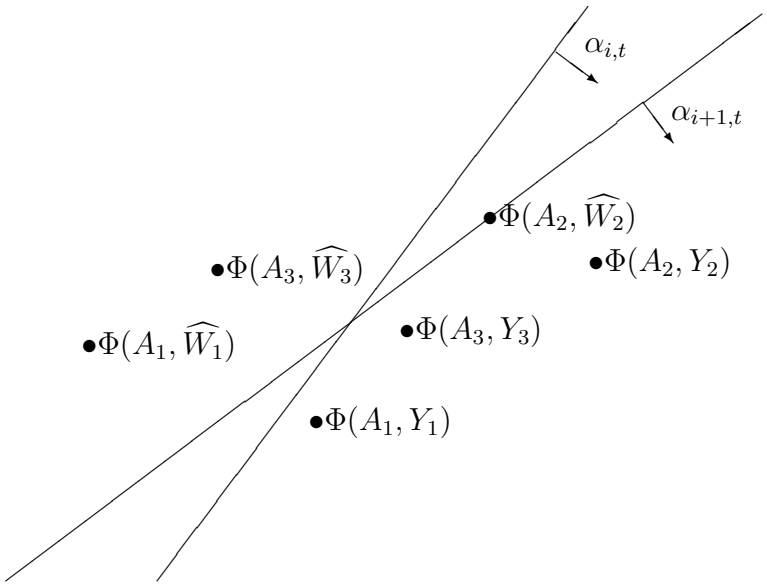


Figure 3-4: Perceptron algorithm. $\Phi(\cdot, \cdot)$ represent feature vectors extracted from sentences. Hyperplane $\alpha_{i,t}$ is updated to $\alpha_{i+1,t}$.

All sorts of features can be put in $\Phi(A, W)$. Authors of [*Collins*, *Saraclar*, and *Roark* (2005)] have worked with word features, in particular trigrams (w_{i-2}, w_{i-1}, w_1).

In addition to trigrams, they used mixed n-grams of words and part-of-speech tags

$$(t_{i-2}, t_{i-1}, t_i), (t_{i-1}, t_i), (t_i), (t_i, w_i) \tag{3.1}$$

$$(t_{i-2}, t_{i-1}, w_i) \tag{3.2}$$

$$(t_{i-2}, w_{i-2}, t_{i-1}, w_{i-1}, t_i, w_i), (t_{i-2}, t_{i-1}, w_{i-1}, t_i, w_i), (t_{i-1}, w_{i-1}, t_i, w_i), (t_{i-1}, t_i, w_i) \tag{3.3}$$

t, w refer to tag and word, respectively at position identified by index. Next, features from parse trees were included. These features include n-grams comprised of nonterminal nodes plus words. For example *(VP,VB,NP,paint,house)* indicates that in the parse tree, there is a rule $VP \rightarrow VB\ NP$ such that word *paint* is head of *VB* subtree, *house* is head of *NP* subtree.

Evaluation on Switchboard data with baseline WER of 37.1% showed interesting results. Discriminatively trained model with only word features yielded WER reduction to 36.4%. Addition of mixed POS of type 3.1 to word features further reduced WER to 36.1%. Including other mixed POS and word features did not improve the resutls. The whole model trained with all features including syntax features yielded WER of 36.0%, which is only 0.1% improvement over POS features.

We can conclude that re-scoring using perceptron algorithm brings significant improvement. However, only words and POS features proved to be beneficial. Syntactic features within perceptron training framework in current implementation seem to be futile.

3.5 Lattice Parsing

Parsing n-best lattices could in theory give better results over parsing n-best lists. A lattice typically contains thousands of sentences whereas n-best lists have n candidates, n is set to a only a few hundreds, sometimes more depending on time complexity of rescoring algorithm. n-best list selection is based on an easy-to-evaluate score such as total acoustic and language model score which is obtained from the recognition

phase. This seems to be quite restrictive. When parsing lattices every sentence is potentially considered by the parser. A more elaborate measure is used not only for rescoring but also for hypothesis selection.

[*Hall* and *Johnson* (2004)] has tried to parse lattices. He adopted coarse-to-fine approach. First stage serves as a rough search. Later stages use complex fine-tuned models to select the best sentence.

In the first stage probabilistic bootom-up parser is used. Its main function is to generate edges together with their category labales. To generate edges parser works over word chunks and builds trees according to a given PCFG. As word chunks grow larger, edges span larger word intervals. Since number of edges which could be added is gigantic (note that number of parses for a sentence is exponential) some pruning must be introduced. Improbable edges are not added into parse agenda. In addition to threshold pruning, inside-outside pruning evaluates likelihood of whole trees. After inside-outside pruning stage only the edges which are present in a probable parse are preserved. In the next stage, elaborate parser computes probability of each parse selected in previous stages. Authors used a lexicalized syntactic parser (Charniak Parser) for the last stage parsing.

Results in speech recognition show that is approach is competitive with n-best list rescoring. However no dramatic improvement was observed. The biggest problem with rescoring lattice is that the number of parses of lattices is enormous. Therefore heavy pruning must be employed. This pruning must done with simplified models to be computationally feasible and thus all potential benefit of working on whole lattices vanishes.

Chapter 4

Thesis Goal

The goal of the thesis is to improve speech recognition of Czech in terms of WER or oracle WER.

The research should investigate possible utilization of morphology in language modeling. Designed language-specific modifications may be not applicable to English, however, the ideas should carry over to other inflective languages.

Chapter 5

Czech Language

In this section, we briefly outline the main characteristics of the Czech language from the point of view of automatic large-vocabulary speech recognition. We also stress the difference between Czech and English and demonstrate the contrast on a few examples.

5.1 Inflection

Czech together with Polish, Slovak, Russian, French and many other languages belongs to the family of so-called *inflective* languages. An overview of topology of languages is given in [*Sgall* (1998)]. Czech, especially, is distinguished by very large degree of inflectionality. Auto-semantic words are inflected into many forms. A noun, for example, is inflected into 14 forms, some of which may be homographic. Table 5.1 displays all forms of Czech word *kniha*, *a book* in English.

Not only nouns but also adjectives, pronouns, numerals and verbs are inflected in Czech. Verbs in particular have very rich morphology. To demonstrate the richness of morphology of Czech verbs, consider, for instance, the verb *pracovat* (*to work*). It has more than 100 different word forms including archaic and colloquial forms. Leaving colloquial and archaic forms aside, verb *pracovat* has more than 60 different forms.

kniha (a book)	singular	plural
Nominative	kniha	knihy
Genitive	knihy	knih
Dative	knize	knihám
Accusative	knihu	knihy
Vocative	kniho	knihy
Locative	knize	knihách
Instrumental	knihou	knihami

Table 5.1: Word forms of Czech noun *kniha* (*a book*).

5.2 Word Order

Besides rich morphology, relative free word order is typical of Czech. Words can be ordered in many ways and still form syntactically correct sentence. Semantics of the sentence, however, may change depending not only on the word order but also on the prosody. More details on this topic can be found in [*Hajičová*, *Partee*, and *Sgall* (1998)]. To give an example of great variety of possible word orderings, consider the Czech sentence *Jan je stále veselý*, which can be translated to English as *John is always cheerful.* All permutations of this particular sentence generate valid Czech sentences. To stress the large variety of word orderings, sentences are enumerated in Tab. 5.2.

Jan je stále veselý.	*Veselý Jan je stále.*
Jan je veselý stále.	*Veselý Jan stále je.*
Jan stále je veselý.	*Veselý stále Jan je.*
Jan stále veselý je.	*Veselý stále je Jan.*
Jan veselý je stále.	*Veselý je stále Jan.*
Jan veselý stále je.	*Veselý je Jan stále.*
Je Jan stále veselý.	*Stále veselý je Jan.*
Je Jan veselý stále.	*Stále veselý Jan je.*
Je stále Jan veselý.	*Stále je Jan veselý.*
Je stále veselý Jan.	*Stále je veselý Jan.*
Je veselý Jan stále.	*Stále Jan je veselý.*
Je veselý stále Jan.	*Stále Jan veselý je.*

Table 5.2: Demonstration of free word order. All permutations of this particular sentence are valid Czech sentences. The English translation is *John is always cheerful.*

5.3 Colloquial Czech

In addition to rich morphology and relatively free word order, colloquial expressions prevail in spoken Czech. Depending on the region, different variations of word forms are popular with native Czech speakers. In Prague, for instance, endings *ej* and *ý* are widespread. In Moravia ending *é* prevails. Table 5.3 shows the colloquial variations of the adjective *obrovský*, which means *huge* in English.

nominative	*obrovský* → *obrovskej*
genitive	*obrovského* → *obrovskýho*
dative	*obrovskému* → *obrovskýmu*
accussative	*obrovského* → *obrovskýho*
vocative	*obrovský* → *obrovskej*
locative	*obrovském* → *obrovským*
	obrovském → *obrovskym*
instrumental	*obrovským* → *obrovskym*

Table 5.3: Colloquial transformation of adjective obrovský (huge). Only singular masculine animate positive affirmative forms are displayed.

Taking into consideration all forms of adjective *obrovský*, there are 27 unique colloquial forms in addition to 53 standard forms.

Not only endings vary in colloquial Czech. Some speakers like to add *v* at the begging of words starting with *o*. Adjective *obrovský* is then transformed to *vobrovský* as well as the other inflections of this word. Even superlative form *nejobrovštější* is transformed to *nejvobrovštější*. This type of colloquial change effectively doubles the number of forms which start with the letter *o*.

Chapter 6

Data Description

This chapter describes both acoustic and text data exploited throughout the thesis. It also renders acoustic models and used front-ends.

6.1 Lidové Noviny

Lidovké Noviny (LN) is a 5-year collection of Czech daily newspaper Lidové Noviny, spanning the period of 1991-1995. LN newspaper covers actual domestic and world-wide events including politics, economy, culture, entertainment and sport.

We preprocessed the collection to obtain a clean text suitable for language model estimation. Numerals were transcribed into text, sentence boundaries were marked, tables were excluded. It contains about 33 mil. tokens, 650k types and 2.3 mil. sentences. This corpus is available at [*Hajič* et al. (2001)]. LN was collected by the Institute of Czech National Corpus.

6.2 CZBN

Acoustic speech corpus, *Czech Broadcast News (CZBN)*, consists of 26 hours of Czech radio (Český Rozhlas, channels 1, 2 and 3) and TV (Czech TV and Prima TV) broadcast news. Sport news, weather forecasts and traffic announcement were excluded from the corpus. 22 hours serves as the training set.

The signal was sampled at 22.05 kHz with 16-bit resolution. HTK toolkit [*Young* (1999)] was utilized for acoustic feature extraction and acoustic model training. Mel-Frequency Cepstral Coefficients were used as acoustic features. Each vector consists of 12 cepstral coefficients plus energy and corresponding delta and delta-delta coefficients. Cepstral mean subtraction was applied to all feature vectors on a per utterance basis. Triphone acoustic models are based on continuous density HMMs. Probability distributions are mixtures of 12 Gaussians. States of the models are tied using broad acoustic classes yielding 5438 states and 18352 different triphones. The speech corpus was recorded and parametrized at the University of West Bohemia, Czech Republic and is available for the public at [*Radová* et al. (2004)].

6.3 MALACH

The *Malach* corpus contains testimonies of survivors of the World War II Holocaust. Personal stories of more 52,000 Holocaust witnesses have been recorded in 32 languages. Testimonies are spontaneously spoken with assistance of a trained moderator. The Czech portion of this huge database has been used in experiments of this thesis. The Czech portion exhibits many difficulties encountered when recognizing Czech speech. Colloquial forms, ungrammatical forms and foreign words are present in the recordings. A detailed description of the Czech portion of the Malach corpus can be found in [*Psutka* et al. (2003)].

For acoustic model training, 336 testimonies were exploited totaling 84 hours of speech. Test set is formed by 10 testimonies of 23 hours. Audio stream was sampled at 44KHz, 16bit resolution. PLP features were used to parametrize the signal at a rate of 100 frames per second. Cross-word tied-mixture acoustic models had 6000 states and 94k Gaussian mixtures. Models were trained at the University of Western Bohemia.

Chapter 7

Fighting OOV

Word forms are the most common recognition units used in current speech recognition systems. Word forms are also the most natural choice since the desired output of most speech recognition systems is a word sequence. Nevertheless, other recognition units might yield superior performance. This chapter discusses several alternative approaches to choosing recognition units and proposes other techniques to reduce impact of out-of-vocabulary (OOV) words.

7.1 Previous Work

Having in mind excessive vocabulary grow of inflective languages, it might be beneficial to use sub-word units. Use of sub-word units leads to higher text coverage and consequently might yield lower word error rates. Several decomposition algorithms have been suggested, both rule-based and automatic. An example of rule-based approach is to decompose words into stems and endings. An automatic approach to decomposition of words into sub-word units is to take an objective function, e.g. perplexity, and derive the sub-word units which maximize the predefined criterion. Both approaches are briefly overviewed below. Note that these methods are applied in the first phase of the recognition process.

7.1.1 Stems and Endings

A morphology-driven method suggested by [*Byrne* et al. (2001)] and elaborated in [*Ircing* (2004)] is to split words into stems and endings, so-called *morphemes*,

$$w_i = stem_i + ending_i$$

Morphemes make up a new vocabulary. To employ this new vocabulary in a recognition system, each word in the text corpus is substituted by its morphemes. So, the corpus looks like $stem_1$, $ending_1$, $stem_2$, $ending_2$, ... Based on this morpheme corpus, a language model is estimated. In case of trigram language model, probability of a morpheme given its history is computed as

$$P(m_i|m_{i-2}, m_{i-1}) = \begin{cases} P(stem_i|stem_i, ending_{i-1}) & \text{if } i\text{-th morph is a stem} \\ P(ending_i|ending_{i-1}, stem_i) & \text{if } i\text{-th morph is an ending} \end{cases}$$

The decoder output is, of course, a sequence of stems and endings. To reconstruct the word sequence, stems and endings are simply concatenated. To distinguish stems from endings, marker symbols are added to each morpheme in such a way that only pairs ($stem$, $ending$) are concatenated to a word form.

Anyhow appealing the motivation of this approach might seem, it has a major drawback. Stems are predicted from both stems and endings although endings of previous words are almost of no use in the estimation of the current stem. Furthermore, prediction of a stem in a trigram model is only based on information of the previous word (its stem and ending), no information of the second previous word is present which is in a sharp contrast to a word form trigram LM. This deteriorates performance possibly boosted by higher text coverage. Indeed, little improvement was gained by applying this model.

7.1.2 Automatically Derived Units

An alternative to use of linguistically-motivated morphological units is to derive the sub-word units in an automatic fashion.

Whittaker [*Whittaker*, *Thong*, and *Moreno* (2001)] suggested to break words into phone sequences of length l. The procedure iterates over l and selects the units which substantially increase log-likelihood of the training data. An n-gram model based on these units is estimated and decoder run. Sub-words of generated lattices were converted to words using confusion networks. Experiments were done on English.

Siivola [*Siivola* et al. (2003)] experimented with an related measure on Finnish. His cost function consisted of two parts which were to be added together. The first part corresponded to unigram likelihood of the training corpus. The second part handled morph length. The longer morph, the higher cost. Working with morphs lead to dramatical decrease in OOV and subsequently in WER reduction. It is worth noting that initial OOV was very high (20%), so WER reduction was expected.

7.2 Inclusion of Rare Words

In this section we propose a simple but effective method for OOV reduction. We aim at adding rare words into LM while keeping the modeling power of word n-grams.

7.2.1 Idea

Language models used in first pass of current speech recognition systems are usually built in the following way. First, a text corpus is acquired. In case of broadcast news, a newspaper collection or news transcriptions are a good source. Second, most frequent words are picked out to form a dictionary. Dictionary size is typically in tens of thousand words. For English, for example, dictionaries of size of 60k words sufficiently cover common domains. Of course, for recognition of entries listed in the Yellow pages, such limited dictionaries are clearly inappropriate. Third, an n-gram language model is estimated. In case of Katz back-off model, the conditional bigram

word probability is estimated as

$$P_1(w_i|w_{i-1}) = \begin{cases} \widetilde{P}(w_i|w_{i-1}) \text{ if } C(w_{i-1}, w_i) > k \\ BO(w_{i-1}) \cdot \widetilde{P}(w_i) \quad \text{otherwise} \end{cases} \tag{7.1}$$

where $\widetilde{P}$ represents a smoothed probability distribution, $BO()$ stands for the back-off weight, and $C(.)$ denotes the count of its argument. Back-off model can be also nicely viewed as a finite state automaton as depicted in Figure 7-1.

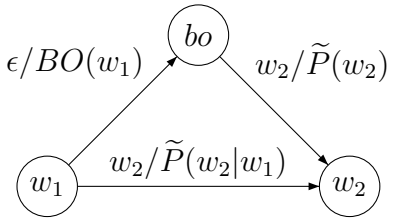


Figure 7-1: A fragment of a bigram back-off model represented as a finite-state automaton.

To alleviate the problem of a high OOV, we suggest to gather supplementary words and add them into the model in the following way.

$$P(w_i|w_{i-1}) = \begin{cases} P_1(w_i|w_{i-1}) & w_i \in D \\ BO(w_{i-1}) \cdot Q(w_i) & w_i \in S \end{cases} \tag{7.2}$$

$P_1()$ refers to the regular back-off model, D denotes the regular dictionary from which the back-off model was estimated, S is the supplementary dictionary which does not overlap with D.

Several sources can be exploited to obtain supplementary dictionaries. Morphology tools can derive words which are close to those observed in corpus. In such a case, $Q(w_i)$ can be set as a constant function and estimated on held-out data to maximize recognition accuracy.

$$Q(w_i) = const \quad \text{for } w_i \text{ generated by morphology} \tag{7.3}$$

Having prior domain knowledge, new words which are expected to appear in audio

recordings might be collected and added into S. Consider an example of transcribing an ice-hockey tournament. Names of new players are desirable in the vocabulary. Another source of S are the words which fell below the selection threshold of D. In large corpora, there are hundreds of thousands words which are omitted from the estimated language model. We suggest to put them into S. As it turned out, unigram probability of these words is very low, so it is suitable to increase their score to make them competitive with other words in D during recognition. $Q(w_i)$ is then computed as

$$Q(w_i) = shift \cdot f(w_i) \tag{7.4}$$

where $f(w_i)$ refers to the relative frequency of w_i in a given corpus, $shift$ denotes a shifting factor which should be tuned on some held-out data.

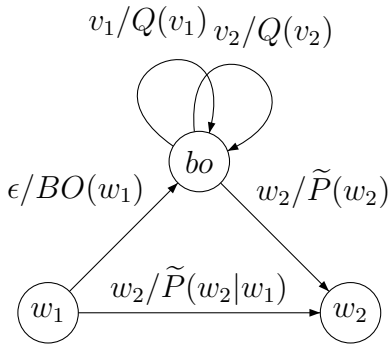


Figure 7-2: A fragment of a bigram back-off model injected by a supplementary dictionary

Note that the probability of a word given its history is no longer proper probability. It does not add up to one. We decided not to normalize the model for two reasons. First, we used a decoder which searches for the best path under Viterbi approximation, so there's no need for normalization. Second, normalization would have involved recomputing all back-off model weights and could also enforce retuning of the language model scaling factor. To rule out any variation which re-tuning of the scaling factor could bring, we decided not to normalize the new model.

In finite-state representation, injection of a new dictionary was implemented as depicted in 7-2. Supplementary words form a loop in the back-off state.

7.2.2 Experiments

We have evaluated our approach on two corpora, Czech Broadcast News and the Czech portion of MALACH data.

CZBN

As a LM training corpus we exploited a collection of newspaper articles from the Lidové Noviny (LN) newspaper. Its vocabulary contains more 650k word forms. OOV rates are displayed in Table 7.1.

Dict. size	OOV
60k	8.27%
80k	6.92%
124k	5.20%
371k	2.23%
658k	1.63%

Table 7.1: OOV rate of transcriptions of the test data. Dictionaries contained most frequent words.

As can be readily observed, moderate-size vocabularies don't sufficiently cover the test data transcriptions. Therefore they are one of the major sources of poor recognition performance.

The baseline language model was estimated from 60k most frequent words. It was a bigram Katz back-off model with Knesser-Ney smoothing pruned by the entropy-based method.

As the supplementary dictionary we took the rest of words from the LN corpus. To learn the impact of injection of infrequent words, we carried out two experiments.

First, we built a uniform loop which was injected into the back-off model. The uniform distribution was tuned on the held-out data. Tuning of this constant is displayed in Table 7.2.

Second, we took relative frequencies multiplied by a shift coefficient as the injected model scores. This shift coefficient was again tuned on held-out data as shown in Table 7.3.

Uniform scale	Acc
12	81.11%
11	81.32%
10	81.60%
9	79.00%

Table 7.2: Tuning of uniform distribution on the held-out set. *Acc* denotes accuracy.

Unigram shift	Acc
no shift	80.48%
e^3	81.46%
e^4	82.09%
e^5	81.25%

Table 7.3: Tuning of the shift coefficient of unigram model on the held-out set.

Then, we took the best parameters and used them for recognition of the test data. Recognition results are depicted in Figure 7.4. The injection of supplementary words helped to increase both recognition accuracy and oracle accuracy. Injection of shifted unigram model brought relative improvement of 13.6%, 3.96% absolute, in terms of accuracy over the 60k baseline model. Uniform injection brought also moderate improvement, despite its simplicity. Indeed, we observed more than 10% relative, 3.05% absolute, improvement in terms of accuracy over the 60k baseline. In terms of oracle accuracy, unigram injection brought more than 30% relative, 4.87% absolute, improvement.

Model	Acc	OAcc
Baseline 60k	70.83%	84.10%
Baseline 80k	72.56%	85.69%
60k + Uniform injection	73.88%	88.90%
60k + Unigram injection	74.79%	88.97%

Table 7.4: Evaluation on 2500 test sentences. *OAcc* stands for the oracle accuracy.

It's worthwhile to mention the model size, since it could be argued that the improvement was achieved by an enormous increase of the model. We decided to measure the model size using two factors. The disk space occupied by the language model and the disk space taken up by the so-called *CLG*. By *CLG* we mean a transducer which maps triphones to words augmented with the model scores. This transducer

Model	CLG size	G size
Baseline 60k	399MB	106MB
60k + Uniform	405MB	115MB
60k + Unigram	405MB	115MB
Baseline 80k	441MB	116MB

Table 7.5: Model size comparison measured in disk space. *G* denotes a language model compiled as a finite-state automaton. *CLG* denotes transducer mapping triphones to words augmented with model scores.

represents the search space investigated during recognition. Table 7.5 summarizes the sizes of the evaluated models.

Injection of supplementary words increased the model size only slightly. To see the difference in the size of the injected models and the traditionally built ones, we constructed a model of 80k most frequent words and pruned with the same threshold as the 60k LM. Not only did this 80k model give worse recognition results, but it also proved to be bigger.

MALACH data

The baseline language model was estimated from transcriptions of the survivors' testimonies. We worked with the standardized version of the transcriptions. To obtain a supplementary vocabulary, we used Czech morphology tools [*Hajič* and *Vidová-Hladká* (1998)]. Out of 41k words we generated 416k words which were the inflected forms of the observed words in the corpus. Note that we posed restrictions on the generation procedure to avoid obsolete, archaic and uncommon expressions. To do so, we ran a Czech tagger on the transcriptions and thus obtained a list of all morphological tags of observed forms. The morphological generation was then confined to this set of tags.

Since there is no corpus to train unigram scores of generated words on, we set the LM score of the generated forms to a constant.

The transcriptions (TR) are not the only source of text data in the MALACH project. [*Psutka* et al. (2004)] searched the Czech National Corpus (CNC) for sentences which are similar to the transcriptions. This additional corpus contains almost

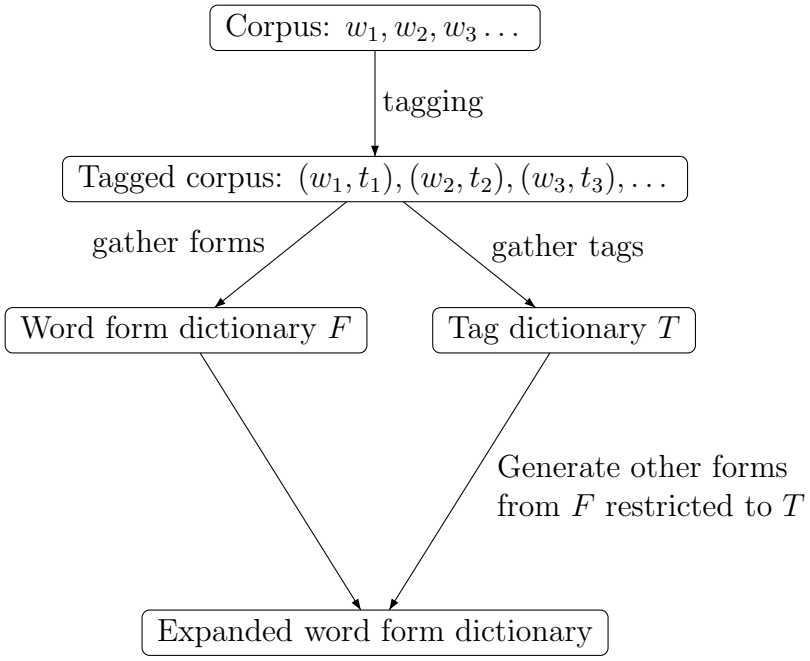


Figure 7-3: Generation of new (unseen) word forms from corpus.

16 million words, 330k types. CNC vocabulary overlaps to a large extent TR vocabulary. This fact is not surprising since the selection criterion was based on a lemma unigram probability. Table 7.6 summarizes OOV rates of several dictionaries.

Dictionary		OOV
Name	Size	
TR41k	41k	5.07 %
TR41k + Morph416k	416k	2.74 %
TR41k + CNC60k	79k	3.04 %
TR41k + CNC100k	114k	2.62 %
TR41k + CNC160k	171k	2.25%
TR41k + CNC329k	337k	1.76 %
All together	630k	1.46 %

Table 7.6: OOV for several dictionaries. *TR*, *CNC* denote the transcriptions, the Czech National Corpus, respectively. *Morph* refers to the dictionary generated by the morphology tools from from TR. Numbers in the dictionary names represent the dictionary size.

We estimated several language models. The baseline models are pruned bigram back-off models with Knesser-Ney smoothing. The baseline accuracy of the model

built solely from transcriptions was 62.65%. We injected constant loop of morphological variants into this model. In terms of text coverage, this action reduced OOV from 5.07% to 2.74%. In terms of recognition accuracy, we observed a relative improvement of 3.5%.

Model	Acc	OAcc
TR41k	62.65%	85.60%
TR41k + Uniform_Morph	63.94%	87.52%
TR41k + CNC_100k	65.53%	88.05%
TR41k + CNC_100k + Inj	66.33%	89.21%
TR41k + CNC_160k	65.81%	88.35%

Table 7.7: Accuracy and oracle accuracy for baseline and injected models. *Uniform_Morph* refers to the constant loop of the morphology-generated words. *CNC_nk* refers to *n* most frequent words of CNC. *Inj* denotes the loop of the rest of words of the CNC corpus and the morphology-generated words.

In the next experiment we took as the baseline LM a linear interpolation of the LM built from transcriptions and a model estimated from the CNC corpus. Into this model, we injected a unigram loop of all the available words. That is the rest of words from the CNC corpus with unigram scores and words provided by morphology which were not already in the model. Table 7.7 summarizes the achieved accuracy and oracle accuracy. Given the fact that the injection only slightly reduced the OOV rate, a small reduction of 2.3% matched our expectations.

model	CLG	G
TR41k	38MB	5.6MB
TR41k + Morph	54MB	11MB
TR41k + CNC_100k	283MB	53MB
TR41k + CNC_100k + Inj	307MB	61MB
TR41k + CNC_160k	312MB	59MB

Table 7.8: Disk usage of tested models. *G* refers to a language model, *CLG* denotes triphone-to-word transducer. *CNC* and *Morph* refer to a LM estimated from transcriptions and the Czech National Corpus, respectively. Morph represents the loop of words generated by morphology. *Inj* is a loop of all words from *CNC* which were not included in *CNC* language model, *Inj* contains also words generated by the morphology.

To learn how the injection affected model size, we measured size of the language model automaton and the optimized triphone-to-word transducer. As in the case

of the LN corpus, injection increased the model size only moderately. Sizes of the models are shown in Table 7.8.

7.3 LG Substitution

7.3.1 Motivation

In this section we propose a strategy to reduce impact of OOV in the second pass. First-pass models have to be heavily pruned to meet memory limitations. In addition, they are built of restricted dictionaries. Therefore, it would be interesting to see whether we could improve on the firs-pass lattices in terms of accuracy and oracle accuracy by employing bigger models in the second pass.

Consider an utterance containing OOV words. Since a decoder can not recognize unknown words, acoustically similar words are output instead. On the word level, the hypothesis looks very unlike the utterance. However, on the phone level, the hypothesis and the utterance might differ only slightly. When looking on the whole phone lattice, the optimal path can be very close to the phone transcription of the utterance.

Consider the utterance *VYSÍLÁME ZPRÁVY*. Since the word *VYSÍLÁME* was not in the lexicon, the best hypothesis produced by decoder was *VYSÍLÁ NESPRÁVNÝ*. Nevertheless, the word sequence *VYSÍLÁ MEZ PRÁVY* was in the generated lattice as well. Note that on the phone level, this hypothesis matches the utterance.

utterance	VYSÍLÁME ZPRÁVY
best first-pass output	VYSÍLÁ NESPRÁVNÝ
k-best first-pass output	VYSÍLÁ MEZ PRÁVY

Table 7.9: Example of output after first pass. Word *VYSÍLÁME* was an OOV word. k-best hypothesis phone sequence matches the utterance phone sequence up to pauses between words.

7.3.2 Algorithm

The proposed idea is to concatenate phones along paths in generated lattices by utilizing a huge dictionary and a language model. The algorithm works in the following way.

1. Remove current LM

   First pass lattices are labeled with words and a language model score. Before we get rid of word labels, we have to remove language model score. Leaving the LM score in the lattice would deteriorate performance, since we are planning to add a new LM. Score of the two LM models together would dominate acoustic score and thus lead to suboptimal solution.

2. Project to phones

   Phone lattices can be obtained from word lattices by utilizing pronunciation lexicons. Words are simply transcribed into all their pronunciations. Another possibility is to take acoustic evidence into account. Our decoder outputs triphone-to-word mappings. So we can use triphone lattices and convert them to phone lattices. We decided to take the latter alternative because it does not enlarge generated lattices and keeps only those phones which were chosen by the decoder.

3. Handle silence models

   The phone lattices contain silence phones, which indicate word boundaries. Consecutive words are delimited by these silence phones. Since we want allow concatenation of phones across word boundaries, we have to get rid of the silence models. This can be done by composition of the phone lattice with the model depicted on Fig 3. It maps the silence phone to the empty symbol. Since pronunciation of each word is followed by a silence model, we have to add an optional silence at every position. Composition with automaton depicted on Figure 3 will do the job.

4. Compose with new LG

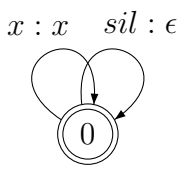


Figure 7-4: An automaton which removes silence phones.

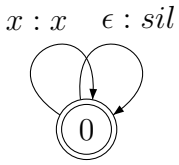


Figure 7-5: An automaton which optionally adds silence phones.

Now, we want to try out all possible concatenations of phones along all paths. The straightforward approach of listing all phone sequences in a lattice and combing with pronunciation lexicon is computationally infeasible. There are too many paths stored in a lattice to be listed. The solution is to create an automaton which would map phones to words (LG) as was introduced in chapter 2.4.1. This automaton encodes not only the mapping but also a language model score. Phone lattice is composed with LG and projected to the output side. Thus a word lattice is obtained.

7.3.3 Results

The proposed method was tested on the CZBN corpus. We experimented with two fist-pass models. Model *60k_bi* was a bigram back-off model estimated on a 60k dictionary. Model *60k_bi_inj* was also a bigram model estimated from 60k words. In addition, it had a unigram loop added into the back-off state containing the rest of words of a 658k dictionary. These models were utilized in the first pass.

To observe whether lexicon substitution can be of any benefit, we built two models. *G_tri_658* was a trigram back-off language model estimated from 658k words. *LG_tri_658* denotes the same language model extended by the corresponding lexicon. First we substituted the language model only (*G_tri_658*). Then we tried to substitute the language model together with lexicon (*LG_tri_658*). In this case, phones were recombined along paths. Table 7.3.3 depicts achieved results.

Consider baseline model *60k_bi*. The accuracy was 70.83%. When rescored by the trigram model, accuracy went up to 71.77%. LG substitution yeilded further accuracy

Model	Acc	OAcc	Rescored by	Acc	OAcc
60k_bi	70.83%	84.10%	G_tri_658	71.77%	84.10%
			LG_tri_658	72.95%	85.09%
60k_bi_inj	74.79%	88.97%	G_tri_658	76.60%	88.93%
			LG_tri_658	76.61%	88.84%

Table 7.10: Results of G and LG substitution. *60k_bi* is built of 60k words, *60k_bi_inj* contains 658k words. *G* means re-scoring by a language model only, *LG* denotes re-scoring by phone recombination along lattice paths and LM substitution. The new LM was estimated from 658k words.

increase to 72.95%. This 4.2% relative improvement over simple LM substitution was due phone recombination. The baseline model consisted of only 60k words, the re-scoring lexicon was by an order of magnitude bigger. So, recombination of phones along lattice paths created words which were not present in the first-pass lattice. Oracle error rate was also improved. The relative improvement was 6.3%.

Lets concentrate on the *60k_bi_inj* model. It already contained all 658k words. Substitution of LM brought some improvement, but further phone recombination was of no use. That phone recombination was not beneficial was expected. All relevant words were chosen in the first pass, the second pass model had no further words to offer.

To conclude this experiment, we can say that phone recombination is beneficial when the models used in the second pass are bigger so that they serve additional relevant words. On the other hand, if the model used in the re-scoring pass contains the same words, phone recombination does not bring any improvement.

7.4 Lemma as Recognition Unit

7.4.1 Motivation

In this section we propose to use lemmas as the recognition units for ASR of Czech. The use of lemmas is advantageous for three reasons. First, a lemmatized dictionary achieves much better text coverage. This obvious property is displayed on Fig. 7.4.1. To achieve 8.20% OOV rate, a 64k word-form dictionary is needed, while the

lemmatized dictionary has only 13k entries. Viewed from another perspective, if we fix dictionary size to 64k, OOV of word-form dictionary is 8.20%, while lemmatized dictionary of the same size yields 3.11% OOV. This is a huge reduction in vocabulary size. OOV was computed on transcriptions of the CZBN corpus. Dictionaries were comprised of N most frequent forms, lemmas of the LN corpus, lemmatized LN corpus, respectively.

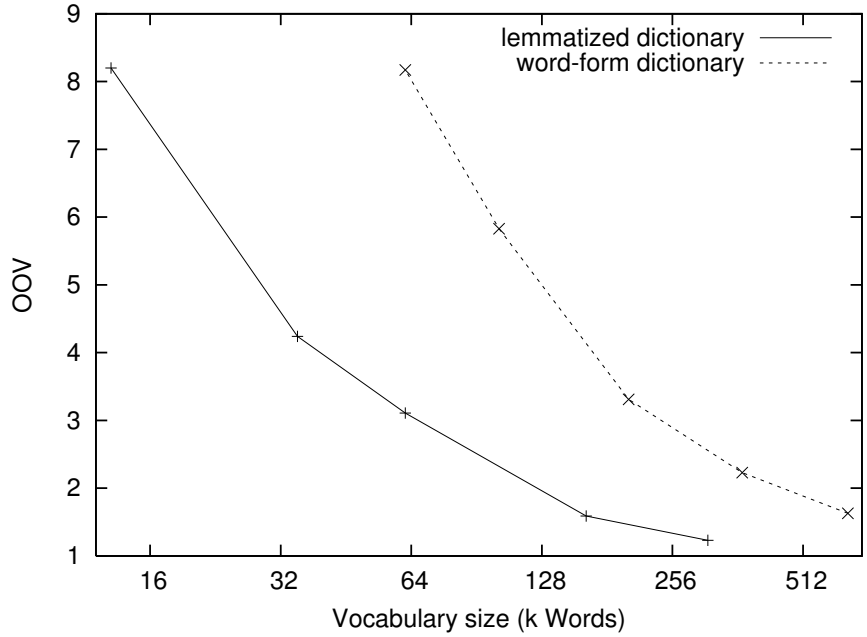


Figure 7-6: Comparison of OOV rates for word form and lemmatized dictionary.

Second, n-gram model based on lemmas is more robust. Frequencies of lemmas are higher than frequencies of word forms, therefore more reliable estimates are obtained.

Third, recognition based on lemmas admits only word forms which exist in language. This fact is in contrast with morpheme based methods where morphemes follow each other without regard to linguistic plausibility. Only in re-scoring phase, invalid morph sequences get pruned out.

7.4.2 Recognition Scheme

The lemma-based recognition process works in two steps:

1. First pass

Assume the vocabulary comprised of the N most frequent lemmas of a training corpus. The pronunciation lexicon lists pronunciations of every lemma. List of pronunciation of a lemma can be either pronunciations of all lemma's forms as defined by morphology, or pronunciations of only those forms which were observed in the corpus. The lexicon obtained by the latter approach is smaller since not all forms of a lemma are actually seen in a real corpus.

Based on the described vocabulary and lexicon, a recognition network (CLG) is estimated and recognition run.

2. Second pass

   In the second pass, LG gets substituted as described in section 7.3. The first pass is intended to provide phone lattice while in the second pass phones of the lattice are recombined into word forms. Language model used in this stage is a huge word form n-gram model.

The suggested scheme can be further generalized. Instead of lemmatizing every word form, only a certain group, e.g. nouns, can be lemmatized. Pronunciation lexicon would then contain pronunciations of non-nouns, and for noun lemma entries a list of pronunciations of all corresponding word forms. More abstractly the process of lemmatization can be viewed as a substitution of a group of word forms by an abstract unit - a class. So, in the first pass, classes are recognized. In the second pass, phones of generated lattices are recombined and thus word lattice obtained.

The natural question is how to evaluate firs-pass lattices. Word error rate can not be computed since we do not have the words yet. Lemma error rate is a bit tricky. Transcriptions can not be lemmatized deterministically therefore the evaluation would be affected by this randomness.

We propose to measure the oracle phone error rate. That is to find the phone sequence which best matches the phone transcription of the utterance and compute the error rate on it.

7.4.3 Experiments

The approach was tested on the CZBN corpus. LN was used to estimate language models. To evaluate the benefits of the lemma-based recognition we built several language models and pronunciations lexicons. All language models were bigram back-off models with Knesser-Ney smoothing. The following three lemma language models were built.

- *LML62* comprised the 62k most frequent lemmas of the LN. Note that there were 309 different lemmas in the corpus. Bigram pruning yielded 2.2M of bigrams.
- *LML35* consists of the 35k most frequent lemmas. Number of bigrams stored in the model was 2.2M.
- *LML13* was estimated from 13k most frequent lemmas. 1.6M bigrams were kept in the model. When we lemmatized top 62k word forms, the number of lemmas was also 13k.

Pronunciation lexicons :

- *LexL371* is a lemma pronunciation dictionary. Pronunciation listed for lemmas in this dictionary are pronunciations of the top 371k word forms. List of pronunciations of a lemma does not contain all pronunciations of all corresponding forms, but only pronunciations of those forms which are among 371k top frequent words.
- *LexL62* is a lemma pronunciation dictionary. It was built in the same way as *LexL371*. Pronunciations listed in dictionary are restricted to pronunciations of the 62k most frequent forms.
- *LexW371* is a word form pronunciation dictionary comprised of pronunciations of top 371k word forms.
- *LexW62* is a word form pronunciation dictionary which contains pronunciations of top 62k word forms.

First Pass				Second Pass			
LM	Lexicon	OPhER	Avg Nodes	LM	Lexicon	WER	OWER
LMW62	LexW62	2.34	4144	LMW371	LexW371	24.57	10.09
LML62	LexL371	1.68	3952	LMW371	LexW371	21.22	7.93
LML35	LexL371	1.99	3399	LMW371	LexW371	22.14	8.85
LML13	LexL371	2.58	4158	LMW371	LexW371	26.25	12.04
LML13	LexL62	2.85	3956	LMW371	LexW371	27.63	13.35

Table 7.11: Results of lemma-based recognition. *OWER* is the oracle word error rate, *OPhER* stands for the oracle phone error rate, and *Avg Nodes* indicates the average number of nodes in a phone lattice

Word error rate of the baseline 62k-word model was 27.72%, the OWER was 12.65%. When re-scored by the 371k model, the WER decreased to 24.57%. This is the value we compare the other models to. Pure lemmatization of the 62k model corresponds to the last model displayed on Table 7.4.3. The WER deteriorated to 27.63%. We conjecture that the deterioration was due to the fact that by merging frequent forms into lemmas we lost information of morphological agreement of succeeding forms. Even the huge pronunciation lexicon LexL371 did not help to match the baseline. An explanation behind this result is that frequent lemmas have most of their forms already listed in *LexL62* so addition of other forms through LexL371 is not that advantageous.

The lemmatized model comprised of 62k lemmas yielded substantial improvement. WER decreased to 21.22% which is 16.6% relative improvement over the baseline. Note that vocabulary size remained the same as in the baseline model. Only the pronunciation model was enlarged.

Lemmatized 35k model also outperformed the baseline. Vocabulary size of this model is half size of the baseline vocabulary. The improvement over the baseline is due to huge pronunciation lexicon. Improvement in terms of oracle word error rate follow the same pattern as word error rate.

To conclude this section, we can say that if we fix vocabulary size, substantial improvement can be gained by extending pronunciation lexicon. Lemma-based language models are coarser than form-based models and by themself perform worse. Nevertheless, when boosting pronunciation lexicon in suggested lemma-based approach,

substantial gains in term of WER and oracle WER can be achieved depending on OOV rates.

Chapter 8

Conclusion

Large-vocabulary continuous speech recognition of Czech has been intensively studied in this thesis. The whole field of speech recognition has been briefly overviewed. Special attention has been paid to language modeling. Some of the most attractive modeling techniques have been described.

OOV words have been identified as one the main culprits of limited speech recognition performance. Several techniques have been suggested to overcome the obstacles posed be OOV words. The proposed methods try to include word forms which would usually not be in the dictionary. The rare words are not considered because they do not appear in corpus often enough or they do not appear at all. Morphology tools come handy to generate the unobserved forms. Three techniques have been proposed.

First, inclusion of rare words proved to be beneficial. On the CZBN data, the improvement over 3% absolute in terms of WER, and over 4% abolute in terms of oracle WER was observed. On more challenging MALACH data, moderate improvement was observed. Furthermore, memory requirements needed for the inclusion increased only slightly.

Second, LG substitution gave interesting insights into rescoring with expanded dictionary. As it turned out, words not used in the first recognition phase can be successfully employed in the rescoring phase. Indeed, dramatic improvement in terms of oracle WER was observed. Nevertheless, if there are no extra words left for second phase, which were not used in the first phase, no improvement should be expected.

Third, lemmas were investigated as recognition units in the first pass. Depending on the data, this approach can either deteriorate or boost performance. If corpus for language model training is small, substantial gains can be achieved.

To summarize, original techniques to speech recognition of Czech have been designed, implemented and tested on real data sets. Achieved recognition improvement experimentally validated correctness of the underlying ideas.

Appendix A

Sample Lattice

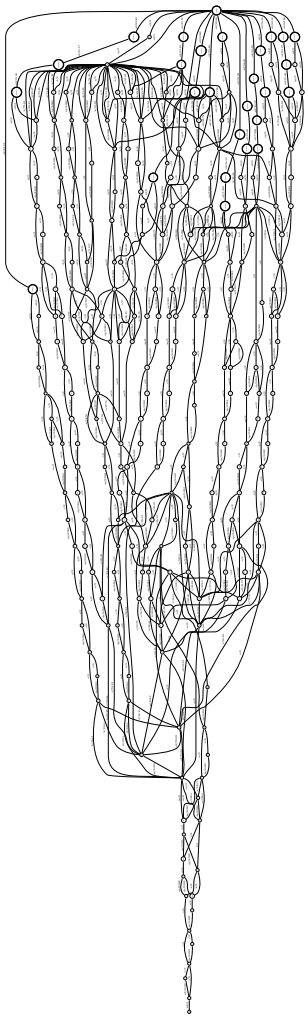


Figure A-1: Phone lattice of uterrance *VYSÍLÁME ZPRÁVY*

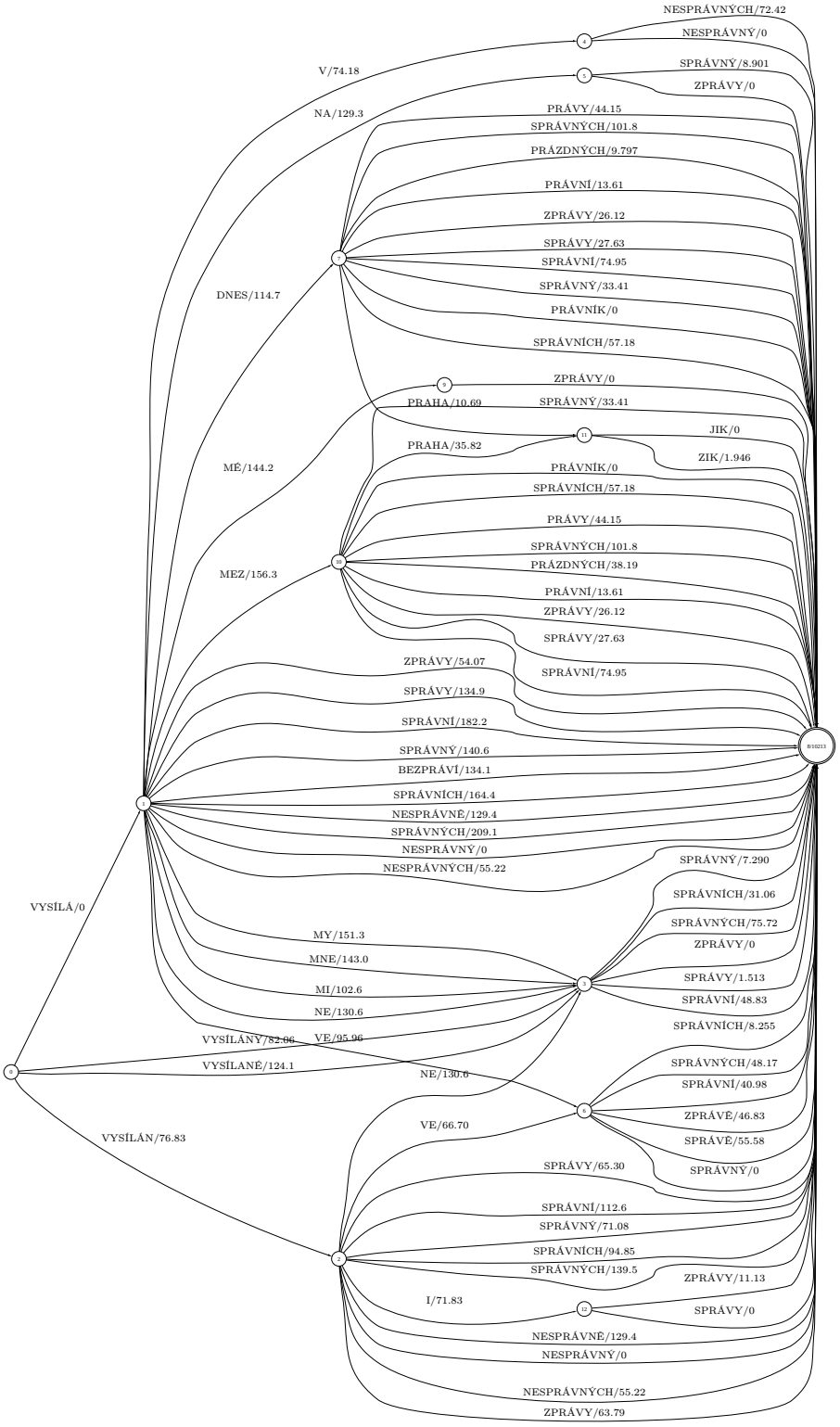


Figure A-2: Word lattice of uterrance *VYSÍLÁME ZPRÁVY*

Appendix B

CZBN Sample Sentences

A list of sample sentences is presented below. The sample sentences are part of the CZBN collection, see chapter 6. Each sentence is displayed in one table, *src* indicates the unique sentence id. Sentence labeled as *True* is a hand-made transcription of acoustic signal. Sentence label as *60k* is oracle best output which was obtained with the baseline 60k bigram model. Sentence labeled as *Expand* is oracle best output generated by expanded model comprised of 658k word forms with uniform injection, see chapter 7.2.2.

src	040719nn0165730.lab
True	STÁLE VYSOKÝ POČET ČESKÝCH ROMŮ ŽÁDAJÍCÍCH O AZYL TO JE PODLE MINISTRA KAVANA TEN PRAVÝ DŮVOD PROČ KANADA NEHODLÁ ZRUŠIT VÍZOVOU
60k	STÁLE VYSOKÝ POČET ČESKÝCH ROMŮ ŽÁDAJÍCÍ H O AZYL TO JE PODLE MINISTRA PRAVÝ DŮVOD PROČ KANADA NEHODLÁ ZRUŠIT VÍZA
Expand	STÁLE VYSOKÝ POČET ČESKÝCH ROMŮ ŽÁDAJÍCÍCH O AZYL TO JE PODLE MINISTRA PRAVÝ DŮVOD PROČ KANADA NEHODLÁ ZRUŠIT VÍZOVOU

src	040719nn0230881.lab
True	MINISTR KAVAN DNES NEVYLOUČIL MOŽNOST ŽE ODPOVĚDÍ BUDE ZAVEDENÍ VÍZ PRO KANAĎANY
60k	MINISTR DNES NEVYLOUČIL MOŽNOST ŽE ODPOVĚDÍ BUDE ZAVEDENÍ VÍZ PRO KANAĎANÉ
Expand	MINISTR DNES NEVYLOUČIL MOŽNOST ŽE ODPOVĚDÍ BUDE ZAVEDENÍ VÍZ PRO KANAĎANY

src	040719nn0401066.lab
True	A ŠÉF VÁM ŘEKNE ZA TŘI ROKY BUDETE MÍT CUKROVKU
60k	A ŠÉF ŘEKNE ZA TŘI ROKY BUDETE MÍT
Expand	A ŠÉF ŘEKNE ZA TŘI ROKY BUDETE MÍT CUKROVKU

src	040806rn0229022.lab
True	KUBÁNSKÝ UPRCHLÍK ELIÁN GONZÁLEZ SE VRÁTÍ KE SVÉMU OTCI UŽ PŘÍŠTÍ TÝDEN
60k	LOŇSKÝ UPRCHLÍK VYJÁDŘIT ZNALEC Z NICH KE SVÉMU OTCI UŽ PŘÍŠTÍ TÝDEN
Expand	BÁŇSKÝ UPRCHLÍKY ELIA GONZÁLEZ SE VRÁTÍ KE SVÉMU OTCI UŽ PŘÍŠTÍ TÝDEN

src	040808rn0078108.lab
True	MEZI ÚČASTNÍKY JSOU TAKÉ BÝVALÝ POLITICKÝ ŠÉF ROZPUŠTĚNÉ KOSOVSKÉ OSVOBOZENECKÉ ARMÁDY HAŠIM TADŽI A UMÍRNĚNÝ KOSOVSKO ALBÁNSKÝ PŘEDÁK IBRAHIM RUGOVA
60k	MEZI ÚČASTNÍKY JSOU TAKÉ BÝVALÝ POLITICKÝ ŠÉF ROZPUŠTĚN JAKO SOF SKÉ OSVOBOZENECKÉ ARMÁDY NAŠIM STAČÍ A UMÍRNĚNÝ VKUSU SKLADBA Z KÝČE DÁM IBRAHIM DROGOVÁ
Expand	MEZI ÚČASTNÍKY JSOU TAKÉ BÝVALÝ POLITICKÝ ŠÉF ROZPUŠTĚNÉ KOSOVSKÉ OSVOBOZENECKÉ ARMÁDY NAŠIM STAČÍ A UMÍRNĚNÝ KOSOVSKÝCH ALBÁNSKÝ PŘEDÁK IBRAHIM DVOUGÓLOVÉ

src	040808rn0088879.lab
True	NA ROZVODNĚNÝCH VÝCHODOSLOVENSKÝCH TOCÍCH JE STÁLE KRITICKÁ SITUACE
60k	NA ROZVOJ JINÝCH VÝCHODOSLOVENSKÉ CÍTÍM STÁLE KRITICKÁ SITUACE
Expand	NA ROZVOD NĚMÝCH VÝCHODOSLOVENSKÝCH TOCÍCH JE STÁLE KRITICKÁ SITUACE

src	040808rn0078108.lab
True	MEZI ÚČASTNÍKY JSOU TAKÉ BÝVALÝ POLITICKÝ ŠÉF ROZPUŠTĚNÉ KOSOVSKÉ OSVOBOZENECKÉ ARMÁDY HAŠIM TADŽI A UMÍRNĚNÝ KOSOVSKO ALBÁNSKÝ PŘEDÁK IBRAHIM RUGOVA
60k	MEZI ÚČASTNÍKY JSOU TAKÉ BÝVALÝ POLITICKÝ ŠÉF ROZPUŠTĚN JAKO SOF SKÉ OSVOBOZENECKÉ ARMÁDY NAŠIM STAČÍ A UMÍRNĚNÝ VKUSU SKLADBA Z KÝČE DÁM IBRAHIM DROGOVÁ
Expand	MEZI ÚČASTNÍKY JSOU TAKÉ BÝVALÝ POLITICKÝ ŠÉF ROZPUŠTĚNÉ KOSOVSKÉ OSVOBOZENECKÉ ARMÁDY NAŠIM STAČÍ A UMÍRNĚNÝ KOSOVSKÝCH ALBÁNSKÝ PŘEDÁK IBRAHIM DVOUGÓLOVÉ

src	040808rn0088879.lab
True	NA ROZVODNĚNÝCH VÝCHODOSLOVENSKÝCH TOCÍCH JE STÁLE KRITICKÁ SITUACE
60k	NA ROZVOJ JINÝCH VÝCHODOSLOVENSKÉ CÍTÍM STÁLE KRITICKÁ SITUACE
Expand	NA ROZVOD NĚMÝCH VÝCHODOSLOVENSKÝCH TOCÍCH JE STÁLE KRITICKÁ SITUACE

src	041923pn0412154.lab
True	SOUD ROZHODL ŽE KUBÁNSKÝ TROSEČNÍK ELIÁN GONZÁLEZ ZŮSTANE AŽ DO KONCE ODVOLACÍHO ŘÍZENÍ VE SPOJENÝCH STÁTECH
60k	SOUD ROZHODL ŽE KUBÁNSKÝ OSVĚDČÍ JEN KONSTELACE ZŮSTANE AŽ DO KONCE ODVOLACÍHO ŘÍZENÍ VE SPOJENÝCH STÁTECH
Expand	SOUD ROZHODL ŽE KUBÁNSKÝ TROSEČNÍK DORIAN GONZÁLEZ ZŮSTANE AŽ DO KONCE ODVOLACÍHO ŘÍZENÍ VE SPOJENÝCH STÁTECH

src	041918rn1207017.lab
True	AKCE CIHLA MÁ VELMI KONKRÉTNÍ CÍL SHROMÁŽDIT PROSTŘEDKY NA REKONSTRUKCI DOMU NA SLAPECH
60k	AKCE JSI MÁ VELMI KONKRÉTNÍ CÍL SHROMÁŽDIT PROSTŘEDKY NA REKONSTRUKCI DOMU NA SLADKÁ
Expand	AKCE SI MÁ VELMI KONKRÉTNÍ CÍL SHROMÁŽDIT PROSTŘEDKY NA REKONSTRUKCI DOMU NA SLAPECH

src	041918rn1100167.lab
True	JAK ŘEKL NAŠEMU REPORTÉROVI JANU MIŠURCOVI PETR BLAŽEK Z DOPRAVNÍCH PODNIKŮ PRAHA V METROPOLI SE O NĚKOLIK SET KORUN POKUTA ZVÝŠÍ OD PRVNÍHO ČERVENCE
60k	JAK ŘEKL NAŠEMU REPORTÉRŮ JIMŽ BY PETR BLAŽEK Z DOPRAVNÍCH PODNIKŮ PRAHA V METROPOLI SE O NĚKOLIK SET KORUN POKUD ZVÝŠÍ OD PRVNÍHO ČERVENCE
Expand	JAK ŘEKL NAŠEMU REPORTÉROVI JENOM ŽUP SCUDY PETR BLAŽEK Z DOPRAVNÍCH PODNIKŮ PRAHA V METROPOLI SE O NĚKOLIK SET KORUN POKUD ZVÝŠÍ OD PRVNÍHO ČERVENCE

src	041918rn0842849.lab
True	DETAILY MÁ NAŠE ZPRAVODAJKA RENATA HAVRANOVÁ
60k	DETAILY MÁ NAŠE ZPRAVODAJKA NOVÁ
Expand	DETAILY MÁ NAŠE ZPRAVODAJKA RENATA HAVRANŮ

src	041823pn0604401.lab
True	ZÍTRA SE OČEKÁVÁ KULMINACE POVODŇOVÉ VLNY V SZOLNOKU A V DALŠÍCH OBLASTECH SMĚREM K SZEGEDU
60k	ZÍTRA SE OČEKÁVÁ UMĚLCE POVODŇOVÉ VLNY V SOUMRAKU A V DALŠÍCH OBLASTECH SMĚREM K SEDĚT U
Expand	ZÍTRA SE OČEKÁVÁ KULMINACE POVODŇOVÉ VLNY V SALÓNU A V DALŠÍCH OBLASTECH SMĚREM K SEDADLŮM

src	041818rn0093479.lab
True	PŘED NAŠÍM ZASTUPITELSTVÍM V HAVANĚ PROTESTOVALI PROTI REZOLUCI TISÍCE KUBÁNCŮ
60k	PŘED NAŠÍM ZASTUPITELSTVÍ V HAVANĚ PROTESTOVALI PROTI REZOLUCI TISÍCE DÁRCŮ
Expand	PŘED NAŠÍM ZASTUPITELSTVÍM V HAVANĚ PROTESTOVALI PROTI REZOLUCI TISÍCE DÁRCŮ

src	041810pn0014495.lab
True	TATO NORMA MÁ DO URČITÉ MÍRY POSÍLIT POSTAVENÍ ZAMĚSTNANCŮ A SLADIT ČESKÉ PRÁVO S LEGISLATIVOU EVROPSKÉ UNIE
60k	TATO NORMA MÁ DO URČITÉ MÍRY POSÍLIT POSTAVENÍ ZAMĚSTNANCŮ A SLADIT ČESKY POSLEDNĚ S VÝZVOU EVROPSKÉ UNIE
Expand	TATO NORMA MÁ DO URČITÉ MÍRY POSÍLIT POSTAVENÍ ZAMĚSTNANCŮ A SLADIT ČESKÝ TRENTU S LEGISLATIVOU EVROPSKÉ UNIE

Appendix C

Malach Sample Sentences

A few sample sentences from the MALACH corpus are displayed below. Transcriptions produced by annotators are marked as *True*. One best automatic transcriptions are labeled as *One*, oracle best transcriptions are labeled as *Oracle*. Setting of the recognition system is described in table 7.6, line 2. Language model is based on transcription plus morphology generated forms.

src	16379_01_00143.lab
True	JESTLI TO ŘEKL NEBO NE ALE JÁ UŽ SEM O NĚM NESLYŠEL NIKDY AŽ SEM HO POTOM PO VÁLCE JEDNOU ZAHLÉDL
One	JESTLI TO ŘEKL NEBO NE ALE JÁ UŽ SEM O NĚM NESLYŠELI KDY AŽ SEM HO POTOM PO VÁLCE JEDNOU ZA ALE
Oracle	JESTLI TO ŘEKL NEBO NE ALE JÁ UŽ SEM O NĚM NESLYŠELI NIKDY AŽ SEM HO POTOM PO VÁLCE JEDNOU ZAHLÉDL

src	16379_06_00020.lab
True	JÁ SEM NEVĚDĚL KDO TO JE PROTOŽE JÁ SEM V NĚMČINĚ SI TO NEUMĚL PŘELOŽIT ŽE ŠPERBR JE JESTŘÁB
One	JÁ SEM NEVĚDĚL KDO TO JE PROTOŽE JÁ SEM V NĚMČINĚ SI TO NEUMĚL PŘELOŽIT JEŽKA DOBRÝ A STAL
Oracle	JÁ SEM NEVĚDĚL KDO TO JE PROTOŽE JÁ SEM V NĚMČINĚ SI TO NEUMĚL PŘELOŽIT ŽE ŠKAREDOVÉ JE JISTA

src	16379_07_00055.lab
True	VPRAVO MŮJ TATÍNEK VLEVO MOJE MAMINKA UPROSTŘED STRÝC BEDŘICH REDLICH
One	VPRAVO MŮJ TATÍNEK VLEVO MOJE MAMINKA UPROSTŘED STRÝC BYL ZŘÍZENÝ
Oracle	VPRAVO MŮJ TATÍNEK VLEVO MOJE MAMINKA UPROSTŘED STRÝC BEDŘICH NE

src	25819_02_00069.lab
True	TEN MĚ ZACHRÁNIL TENKRÁT ŽIVOT PONĚVADŽ BYCH BÝVALA MUSELA HNED DO TRANSPORT
One	TEN NEZACHRÁNIL TENKRÁT ŽILO PRO NÁS BYLO MUSELA NAD DO TRANSPORTU
Oracle	TEN MĚ ZACHRÁNIL TENKRÁT ŽIVOT NABITÉHO MUSELA HNED DO TRANSPORT

src	25819_02_00137.lab
True	JÁ SEM ŘEKLA PROSÍM VÁS JÁ MÁM V TOMHLE PENÍZE NO SAMO DOUFALA SEM ŽE MĚ POMŮŽE JO
One	A CELKEM TO SVAZAMA V TOMHLE PENÍZ ANO SAMOTKU DOUFALA SEM ŽE MĚ TOMU ŽE JO
Oracle	JÁ SEM TAM PROSÍM VÁS JÁ MÁM V TOMHLE PENÍZE NO SAMOTKU DOUFALA SEM ŽE MĚ POMŮŽE JO

src	25819_03_00126.lab
True	A JÁ SEM TAM JÁ ŠLA DOMŮ A NATREFILA SEM TAM MUŽE
One	A JÁ SEM TAM NAŠLA DOMU A NA PROSILA SEM TAM BOHUŽEL
Oracle	A JÁ SEM TAM JÁ ŠLA DOMŮ A NAKRESLILA SEM TAM MUŽE

src	25819_03_00152.lab
Oracle	A JÁ SEM TAM JÁ ŠLA DOMŮ A NAKRESLILA SEM TAM MUŽE
True	CHYTLI MĚ CO TAM DĚLÁM JAK SE ŽE SEM JÁ SEM ŘÍKALA NEJSEM ŽÁDNÁ NĚMKYNĚ JÁ SEM PŘE UTEKLA Z TEREZÍNA
One	CHYTLI MĚ TO TAM DĚLÁM JÁ SEM ŽE SEM TAM JSME KAMENY SEM ŠLA MAMINKY NĚCO PŘES UTEKLA Z TEREZÍNA

src	25819_04_00096.lab
True	A KDYŽ SEM HO VIDĚLA TAK SEM ŘEKLA S TÍM CHCI MÍT DÍTĚ
One	A TY SAMO VIDĚLA TAK SEM SE PROSTĚ BYTIM VIDÍTĚ
Oracle	A TY SEM O VIDĚLA TAK SEM ŘEKLA S TÍM CHCI MÍT DÍTĚ

src	26127_02_00067.lab
True	JÁ NEMĚLA ANI ZDÁNÍ PROTOŽE JÁ SEM BYLA V PRAZE NAROZENÁ VYCHOVANÁ JÁ SEM V ŽIVOTĚ HRÁBĚ NEVIDĚLA NATOŽ ABYCH VĚDĚLA CO SE S NIMI DĚLÁ
One	JÁ NEMĚLA ANI ZDÁNÍ TAKŽE JÁ SEM BYLA V PRAZE NAROZENÁ VYCHOVANÁ JÁ SEM ŽIVOTĚ HRABĚ NEVIDĚLA NATOŽ ABYCH VĚDĚLA CO SE S NIMI DĚLALA
Oracle	JÁ NEMĚLA ANI ZDÁNÍ PROTOŽE SEM BYLA V PRAZE NAROZENÁ VYCHOVANÁ JÁ SEM V ŽIVOTĚ RÁDA NEVIDĚLA NATOŽ ABYCH VĚDĚLA CO SE S NIMI DĚLÁ

src	26127_02_00141.lab
True	POKUSILY JSTE SE NĚKDE SEHNAT NĚJAKÉ JÍDLO POTAJÍ UKRÁST
One	POKUSIL JSTE SE NĚKDE SEHNALA NĚJAKÉ JÍDLO PANÍ BYLA
Oracle	POKUSILY JSTE SE NĚKDE SEHNAT NĚJAKÉ JÍDLO POKOJI BYLA

src	26171_01_00128.lab
True	NO TO BYLO TAKÉ HORROROVÝ SAMOZŘEJMĚ VŠECHNO TEN ROZLOUČENÍ SE ZNÁMÝMI PROSTĚ PŘED TÍM UŽ SE ODEVZDALI TO UŽ NEVÍM ROK PŘED TÍM NEBO KDY VEŠKERÝ ZLATO A STŘÍBRO A VŠECHNO SE HLÁSILO A TO SE ODEVZDÁVALO A POTOM SE SMĚLO VZÍT PADESÁT KILO NA OSOBU SEBOU NO TAK BRALI JSME VOBLÍKÁNÍ
One	NO TO BYLO TAKÉ HOLOHLAVÝ SAMOZŘEJMĚ VŠECHNO ROZLOUČENÍ SE ZNÁMÝMI PROSTĚ PŘED TÍM UŽ SE ODEVZDALI TO UŽ NEVÍM ROK DĚTI NEBO KDY VEŠKERÝ ZLATO A STŘÍBRA VŠECHNO SE HLÁSILA TO SE ODEVZDÁVALO A POTOM SE SMĚLO VZÍT PADESÁT KILO NA OSOBU SE VÁM TAKÉ BRALI JSME OBLÉKLA NÍ
Oracle	NO TO BYLO TAKÉ HOLOHLAVÝ SAMOZŘEJMĚ VŠECHNO ROZLOUČENÍ SE ZNÁMÝMI PROSTĚ PŘED TÍM UŽ SE ODEVZDALI TO UŽ NEVÍM ROK PŘED TÍM NEBO KDY VEŠKERÝ ZLATO A STŘÍBRO A VŠECHNO SE HLÁSILO TO SE ODEVZDÁVALO A POTOM SE SMĚLO VZÍT PADESÁT KILO NA OSOBU SEBOU NO TAK BRALI JSME OBLÉKLA ANI

src	26171_02_00078.lab
True	BYLY ROZTRHANÝ PROSTĚ NĚJAKÝ STARÝ HADRY ALE MĚ TO BYLO ÚPLNĚ JEDNO ŽÁDNÝ PRÁDLO ŽÁDNÝ KALHOTKY POD TO NIC A JAKÝSI DŘEVÁKY JÁ SEM V TOM NEUMĚLA CHODIT BYLO TO VELKÝ KLAPALO TO A TO BYLO VŠECHNO
One	BYLO STRANĚ PROSTĚ A TY STARÝ HADRY ANI TO BYLO ÚPLNĚ JEDNO ŽÁDNÝ PRÁDLO ŽÁDNÝ KALHOTY POPELNICE A JAKÝSI DŘEVÁKY JÁ SEM TO MĚ MĚLA CHODÍ BYLO TO VELKÝ KAPALO TO A TO BYLO VŠECHNO
Oracle	BYL STRANY PROSTĚ TY STARÝ HADRY ALE MĚ TO BYLO ÚPLNĚ JEDNO ŽÁDNÝ PRÁDLO ŽÁDNÝ KALHOTY POD TEN NIC A JAKÝSI DŘEVÁKY JÁ SEM V TOM NEUMĚLA CHODIT BYLO TO VELKÝ KLAPALO TO A TO BYLO VŠECHNO

src	26171_03_00031.lab
True	PROTOŽE JÁ KDYŽ SEM VIDĚLA JÁ PROSTĚ JÁ SEM DO DNESKA TAKOVÁ VŽDYCKY SEM KDYŽ SEM VIDĚLA NĚCO ŽE NĚMCI DĚLALI STRAŠNÝ VĚCI
One	TO JÁ KDYŽ SEM VIDĚLA NÁM PROSTĚ JÁ SEM DO DNESKA TAKOVÁ VŽDYCKY SE VELICE VIDĚLA NĚCO ŽE NĚMCI DĚLALI STRAŠNÝ VĚCI
Oracle	JÁ KDYŽ SEM VIDĚLA JÁ PROSTĚ JÁ SEM DO DNESKA TAKOVÁ VŽDYCKY SEM KTERÝ SEM VIDĚLA NĚCO ŽE NĚMCI DĚLALI STRAŠNÝ VĚCI

src	26171_03_00071.lab
True	A TA PODLAHA POD TÍM SI PŘEDSTAVTE TAKHLE DEJME TOMU ŠTVERCOVOU MÍSTNOST S BETONOVOU PODLAHOU A POD TĚMA SPRCHAMI BYL JAKOBY KRUH A TAM TO BYLO SNÍŽENÝ
One	A TA PODLAHA POD TÍM SI PŘEDSTAVTE TAK TEN DEJME TOMU ČTVERCOVÝ MÍSTNOST A Z BETONOVOU PODLAHO POD TĚMI SPRCHAMI BYL TAKOVÝ KLUK A TAM TO BYLO SNÍŽENI
Oracle	A TA PODLAHA POD TÍM SI PŘEDSTAVTE TAKHLE DEJME TOMU ČTVERCOVÝ MÍSTNOST S BETONOVOU PODLAHO A POD TĚMA SPRCHAMI BYL JAKOBY KRUH A TAM TO BYLO SNÍŽENI

src	26797_03_00207.lab
True	V ARMÁDĚ JO TEN BYL ZAGITOVÁN DO RUDÉ ARMÁDY PROŠEL S NIMI NĚJAKÝ NĚJAKÝ AKCE TEDA ŽE TAKÉ ŽE A POTOM KDYŽ SE TVOŘILA TA NAŠE JEDNOTKA TAK SI ZAŽÁDAL O PŘELOŽENÍ
One	NORMÁLNÍHO NEMUSELA GIDEON RUDÉ ARMÁDY PROŠEL S NIMI NĚJAKÝ NĚJAKÝ AKCE TEDA ŽE TAKÉ ŽE A POTOM KDYŽ SEM POŘÁD TA NAŠE JEDNOTKA TAK SI ZAŽÁDAL O PŘILOŽENI
Oracle	V ARMÁDĚ JO BYL MUSELI DO RUDÉ ARMÁDY PROŠEL S NIMI NĚJAKÝ NĚJAKÝ AKCE TEDA ŽE TAKÉ ŽE A POTOM KDYŽ SE TVOŘILA TA NAŠE JEDNOTKA TAK SI ZAŽÁDAL O PŘELOŽENÍ

src	26797_05_00058.lab
True	TEHDY JEŠTĚ NE TO DĚLALI VŠECHNO JEŠTĚ V CIVILU A TEPRVE POTOM KDYŽ PŘIŠLA TA SKUPINA TĚCH DEVADESÁTI LIDÍ Z TĚCH OLEÁNEK TO BYL VLASTNĚ TEN ZÁKLAD TÉ ARMÁDY A TA SKUPINA VZNIKLA TÍM ZPŮSOBEM ŽE EXISTOVAL HNE JEŠTĚ PŘED VÁ PŘED VYPUKNUTÍM SOVĚTSK NĚMECKO POLSKÉ VÁLKY EXISTOVALA ORGANIZACE UPRCHLÝCH DŮSTOJNÍKŮ A VOJÁKŮ V KRAKOVĚ KTEŘÍ UPRCHLI Z PROTEKTORÁTU ŽE
One	TEHDY JEŠTĚ NE TO JE VŠECHNO JEŠTĚ TŘÍDILO A TEPRVE POTOM KDYŽ PŘIŠLA TA SKUPINA TĚCH DEVADESÁTI LIDI Z TĚCH POLÁNEK TO BYL VLASTNĚ TEN ZÁKLAD TÉ ARMÁDY A TA SKUPINA PŘÍKAZY ZPŮSOBEM ŽE EXISTOVAL JEŠTĚ PŘED VÁLKOU PŘED VYPUKNUTÍM SOVĚTSKOU NĚMECKO POLSKÉ VÁLKY EXISTOVALO ORGANIZACE UPRCHLÍKY DŮSTOJNÍKŮ O VOJÁKŮ V KRAKOVĚ KTEŘÍ UTEKLI Z PROTEKTORÁTU ŽE
Oracle	TEHDY JEŠTĚ NE TO VŠECHNO JEŠTĚ V CIVILU A TEPRVE POTOM KDYŽ PŘIŠLA TA SKUPINA TĚCH DEVADESÁTI LIDÍ Z TĚCH POLÁNEK TO BYL VLASTNĚ TEN ZÁKLAD TÉ ARMÁDY A TA SKUPINA VZNIKLA TÍM ZPŮSOBEM ŽE EXISTOVAL JEŠTĚ PŘED VÁLKOU PŘED VYPUKNUTÍM SLOVENSKO NĚMECKO POLSKÉ VÁLKY EXISTOVALA ORGANIZACE UPRCHLÝCH DŮSTOJNÍKŮ A VOJÁKŮ V KRAKOVĚ KTEŘÍ UPRCHLI Z PROTEKTORÁTU ŽE

src	26797_06_00161.lab
True	TAM JSME BYLI TERČEM A TAM NÁM TO FIČELO KOLEM HLAVY JO TY KULKY PROTOŽE TAM TO BYLA ROVINA TAM SE NEBYLO KAM SKRÝT NORMÁLNĚ TEN MINOMETČÍK ŽE JO MÁ U TOHO MINOMETU NĚJAKÝ OKOP TAKÉ ŽE UDĚLANÝ ŽE TAM TO NEBYLO MOŽNÝ PROSTĚ PROTOŽE ANI TEN OKOP PRO MINOMET NEBYL ŽE NO TAKŽE ASI TAKHLE
One	TAM JSME BYLI TERČEM A TAM NÁM TO SLYŠELO KOLEM HLAVNĚ JO TY KŮRKY KLID A TO BYLO JAKO SE NEBYLO KAM JSTE VYDOVÁDĚNY ULICI ŽE JO NO V TOM ŽIVOTĚ NĚJAKÝ OKOUKL TAKÉ ŽE UDĚLANÝ ŽE TAM TO NEBYLO MOŽNÝ PROSTĚ BYLA JINDE OKO POLOVINU NEBYL NO TAKŽE ASI TAKHLE
Oracle	TAM JSME BYLI TERČEM A TAM NÁM TO SIČOVA KOLEM HLAVY JO TY KULKY TO BYLA TÁBOROVÉHO TAM SE NEBYLO KAM SKRÝT NORMÁLNĚ TEN VEDOUCÍ ŽE JO NO U TOHO MÉHO NĚJAKÝ OKOUKL TAKÉ ŽE UDĚLANÝ ŽE TAM TO NEBYLO MOŽNÝ PROSTĚ TEN NAJDE OKUPOVANÉMU NEBYL ŽE NO TAKŽE ASI TAKHLE

Bibliography

Bahl, Lalit R., Jelinek, Frederick, and Mercer, Robert L.: A Maximum Likelihood Approach to Continuous Speech Recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence 5 1983, 179–190

Bell, Timothy C., Cleaqry, John G., and Witten, Ian H.: Text Compression. Prentice Hall, 1990

Bellegarda, Jerome R.: Latent Semantic Mapping: Dimensionality Reduction via Globally Optimal Continuous Parameter Modeling. In Proceedings of the IEEE ASRU 2005 Workshop. San Juan (Puerto Rico), November 2005

Byrne, W. et al.: On Large Vocabulary Continuous Speech Recognition of Highly Inflectional Language - Czech. In Eurospeech 2001. 2001

Chelba, Ciprian and Acero, Alex: Position Specific Posterior Lattices for Indexing Speech. In Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05). Ann Arbor, Michigan: Association for Computational Linguistics, June 2005 ⟨URL: `http://www.aclweb.org/anthology/P/P05/P05-1055`⟩, 443–450

Chen, Stanley, Beeferman, Douglas, and Rosenfeld, Ronald: Evaluation metrics for language models. In DARPA Broadcast News Transcription and Understanding Workshop. 1998 ⟨URL: `citeseer.ist.psu.edu/chen98evaluation.html`⟩

Chen, Stanley F. and Goodman, Joshua: An Empirical Study of Smoothing Techniques for Language Modeling. In Proceedings of the 34th Annual Meeting of the ACL. 1996, 310-318

Chen, Stanley F. and Goodman, Joshua: An Empirical Study of Smoothing Techniques for Language Modeling. Center for Research in Computing Technology, Harvard University, Cambridge Massachusetts, 1998 – Technical report

Collins, Michael, Saraclar, Murat, and Roark, Brian: Discriminative Syntactic Language Modeling for Speech Recognition. In ACL '05: Proceedings of the 43th Annual Meeting on Association for Computational Linguistics. Association for Computational Linguistics, 2005, 507–514

Cover, T. M. and Thomas, J. A.: Elements of Information Theory. John Wiley & Sons, 1991

Dempster, A., Laird, N., and Rubin, D.: Maximum Likelyhood from Incoplete Data Using the EM Algorithm. Journal of the Royal Society of Statistics 39(1) 1977, 1-38

Gale, William A. and Sampson, Geoffrey: Good-Turing frequency estimation without tears. Journal of Quantitative Linguistics 2 1995, 217-237

Gildea, D. and Hofmann, T.: Topic-based language models using EM. In Proceedings of Eurospeech. Volume 6, 1999 ⟨URL: `citeseer.nj.nec.com/article/gildea99topicbased.html`⟩

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

Hajič, Jan et al.: Prague Dependency Treebank 1.0. Linguistic Data Consortium (LDC), catalog number LDC2001T10, 2001

Hajič, Jan and Psutka, Josef: Jak přepsat hokejový zápas. Rozhovor pro Lidové Noviny, 2003

Hajič, Jan and Vidová-Hladká, Barbora: Tagging Inflective Languages: Prediction of Morphological Categories for a Rich, Structured Tagset. In Proceedings of the Conference COLING ACL '98. Mountreal, Canada, 1998, 483-490

Hajičová, Eva, Partee, Barbara, and Sgall, Petr: Topic-focus articulation, tripartite structures, and semantic content. Amsterdam, Netherlands: Kluwer Academic Publishers, 1998, ISBN 0–7923–5289–0

Hall, Keith and Johnson, Mark: Attention Shifting for Parsing Speech. In Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics. 2004

Huang, Xuedong, Acero, Alex, and Hon, Hsiao-Wuen: Spoken Language Processing. Prentice Hall, 2001, ISBN 0130226165

Hönig, Florian et al.: Revising Perceptual Linear Prediction (PLP). In Proceedings of INTERSPEECH-2005. 2005, 2997–3000

Ircing, Pavel: Large Vocabulary Continuous Speech Recognition of Highly Inflectional Language (Czech). Ph. D thesis, University of West Bohemia in Pilsen Department of Cybernetics, 2004

Jelinek, Frederick: Self-organized language modeling for speech recognition. In **Waibel, Alex and Lee, Kai-Fu, editors:** Readings in Speech Recognition. Morgan-Kaufmann, San Mateo, CA, 1990, 450-506

Jelinek, Frederick: Statistical methods for speech recognition. The MIT Press, 1997, ISBN 0–262–10066–5

Jelinek, Frederick and Mercer, Robert L.: Interpolated estimation of Markov source parametrs from sparce data. In Proceedings of the Workshop of Pattern Recognition in Practice. 1980

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

Kneser, Reinhard and Ney, Hermann: Improved Backing-off for m-gram Language Modeling. In IEEE International Conference on Acoustic, Speech and Signal Processing. 1995, 181-154

Krbec, Pavel: Language Modeling for Speech Recognition of Czech (PhD thesis). Ph. D thesis, Institute of Formal and Applied Linguistics, 2005, 1P05ME786

Manning, D. Chritopher and Schütze, Hinrich: Foundations of Statistical Natural Language Processing. The MIT Press, 2000

Mohri, M., Pereira, F., and Riley, M.: Weighted Automata in Text and Speech Processing. In Proceedings of the 12th biennial European Conference on Artificial Intelligence (ECAI-96),. 1996

Mohri, M., Pereira, F., and Riley, M.: Weighted Finite-State Transducers in Speech Recognition. Computer Speech and Language 16 2002, 69-88

Ney, Hermann, Essen, Ute, and Kneser, Reinhard: On Structuring Probabilistic Dependencies in Stochastic Language Modeling. Computer, Speech and Language 8 1994, 1-38

Psutka, J. et al.: Issues in annotation of the Czech spontaneous speech corpus in the MALACH project. In Proceedings of the 4th International Conference on Language Resources and Evaluation. Lisbon, Portugal, 2004

Psutka, Josef: Komunikace s počítačem mluvenou řečí. Academia, 1995

Psutka, Josef et al.: Large Vocabulary ASR for Spontaneous Czech in the MALACH Project. In EUROSPEECH 2003 Proceedings (8th European Conference on Speech Communication and Technology). Volume 3, ISCA, September, 2003 2003, 1821-1824

Radová, Vlasta et al.: Czech Broadcast News Speech. Linguistic Data Consortium (LDC), catalog number LDC2004S01, 2004

Rayner, Manny et al.: A Voice Enabled Procedure Browser for the International Space Station. In Proceedings of the ACL Interactive Poster and Demonstration Sessions. Ann Arbor, Michigan: Association for Computational Linguistics, June 2005 ⟨URL: `http://www.aclweb.org/anthology/P/P05/P05-3008`⟩, 29–32

Schwenk, Holger and Gauvain, Jean-Luc: Neural Network Language Models for Conversational Speech Recognition. 2004

Seymore, K. and Rosenfeld, R.: Scalable Backoff Language Models. In Proc. ICSLP '96. Volume 1, Philadelphia, PA, 1996 ⟨URL: `citeseer.ist.psu.edu/andr96scalable.html`⟩, 232–235

Sgall, Petr: Prague School Typology. 1998

Siivola, Vesa et al.: Unlimited Vocabulary Speech Recognition Based on Morphs Discovered in an Unsupervised Manner. In Proceedings Eurospeech. Geneva, 2003, 2293–2296

Stolcke, A.: Entropy-based Pruning of Backoff Language Models. In In Proceedings of the ARPA Workshop on Human Language Technology. 1998 ⟨URL: `citeseer.ist.psu.edu/stolcke98entropybased.html`⟩

Whittaker, E. W. D., Thong, Jean Manuel Van, and Moreno, Pedro: Vocabulary Independent Speech Recognition Using Particles. In ASRU 2001. 2001

Wright, J.H., Gorin, A.L., and Abella, A.: Spoken Language Understanding Within Dialogs Using A Graphical Model Of Task Structure. In Proceedings of ICSLP. Sydney, Australia, 1998

Xu, Peng, Chelba, Ciprian, and Jelinek, Frederick: A study on richer syntactic dependencies for structured language modeling. In ACL '02: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics. Morristown, NJ, USA: Association for Computational Linguistics, 2001, 191–198

Young, S.: The HTK Book. Entropic Inc., 1999