Karlsruhe Institute of Technology

Discriminative Maximum Entropy Language Model in the Context of Large-Vocabulary Speech Recognition for Russian

Diploma Thesis of

Evgeniy Shin

At the Department of Informatics
Institute for Anthropomatics

Reviewer:	Prof. Alex Waibel
Second reviewer:	Dr. Sebastian Stüker
Advisor:	Kevin Kilgour

Duration: 01. Januar 2013 – 30. Juni 2013
Version: July 19, 2013

I declare that I have developed and written the enclosed thesis completely by myself, and have not used sources or means without declaration in the text.

Karlsruhe, July 19, 2013

. . . .

(**Evgeniy Shin**)

Abstract

This work is devoted to large-vocabulary speech recognition of Russian language with an accent to language modelling. Problems of high out-of-vocabulary rate and data sparsity are addressed by using sub-words based search vocabulary together with special language modelling techniques. In particular a discriminative model for word endings for Russian language is introduced. This model prevents or decreases confusion between different endings, and, therefore, leads to improvements of the speech recognition system in terms of word error rate. The model is based on the Maximum Entropy modelling framework. Moreover a full-word based n-gram model is applied to address the problem of short contexts, which occurs in case of using sub-words based n-gram model for the underlying recognition system. Both models are applied for rescoring of n-best lists from the baseline system.

Zusammenfassung

Diese Arbeit beschäftigt sich mit der automatische Spracherkennung System für Russische Sprache, mit Fokus zu Sprachmodelierung. Probleme der höhen *out-of-vocabulary* Rate und *data-sparsity* sind adressiert durchAusnutzung der Morphologie der Wörter und besondere Sprachmodelierungsthechniken. Insbesondere ein diskriminatives Model für Wortendungen ist presentiert. Dieses Model dient zur vermeidung beziehungsweise zur minderung der Verwechselbarkeit der Endungen, was führt zu Verbesserungen der Spracherkennung, gemessen mit der Wort Fehler Rate. Das Model basiert auf dem matematischen Gerüst der Maximum Entropie. Außerdem mit einem besonderem N-gram model wird das Problem des kleineren Kontexte angesprochen. Dieses Problem tritt bei der Verwendung der Morpheme als Suchvokabular ein.Beide Modele werden für n-best listen vom unterliegende Spracherkennungssystem angewendet.

Contents

1. Introduction

Nowadays human-machine interaction plays an increasingly important role in our life. The use interfaces that are natural for humans could make working, travelling, shopping and sight-seeing easier, intelligent and more exciting. The most important part of these interfaces is the human speech. It is simple and natural for humans, but for machines speech understanding is not a trivial task. Solving this task would shift human-machine interaction to a new level. Just imagine a web browser, which understands your queries and gives the information you need immediately, or a pocket translator, which simultaneously pronounces the phrase that you have just said but in another language.

According to [Lew09], more than 7000 different languages are spoken in the world. Only 24 languages have at least 50 million first-language speakers. And, there are 340 languages which are spoken by less than 100 speakers.

Such a variety creates such a wonderful variety of languages on the one side, and, on the other side, makes it so difficult for machines to process them.

This work is devoted to large-vocabulary speech recognition of Russian with an emphasis of language modelling. Problems of high out-of-vocabulary rate and data sparsity are addressed by using sub-words based search vocabulary together with special language modelling techniques. In particular a discriminative language model for word endings for Russian language is introduced. This model prevents or decreases confusion between different endings, and, therefore, leads to improvements of the speech recognition system in terms of word error rate. The model is based on the *Maximum Entropy* modelling framework. Moreover a full-word based n-gram model is applied to address the problem of short contexts, which occurs in case of using sub-words based n-gram model for the underlying recognition system. Both models are applied for re-scoring of n-best lists from the baseline system.

This chapter gives a brief introduction to the common structure and function of LVSR and its application to Russian. Chapter 2 describes related work and an approach taken in this work.

Chapter 3 describes the generation process of the *sub-words based search vocabulary* used for this work, discusses problems of its use and solutions.

Next, Chapter 4 speaks about *Maximum Entropy* modelling. It describes the mathematical background of it and introduces an application of *Maximum Entropy endings discrimination* to Russian.

In Chapter 5 the experiments done for this work are described, results are presented and discussed.

And at the last chapter 6 achievements of this work are summarized, consequences and conclusions discussed.

1.1. LVSR

LVSR is featured, as the name says, by a vocabulary size and continuous, i.e. with undefined number of words, word sequence. That means, in contrast to recognition of speech commands, significantly larger vocabularies and selecting word for word from this vocabulary to get a word sequence. LVSR systems try to cover the majority of words in a specific speech domain defined by an application. Vocabularies of such systems can grow to hundreds thousand words. Furthermore continuous speech makes the search space grow exponentially in the number of words in a sequence. To accomplish the task LVSR systems become very complex.

An example of such a system is shown in Figure 1.1. At first, the acoustic features, which are relevant to speech recognition are extracted from the incoming signal. On the basis of these features the decoder applies an acoustic and a language model, to make pruning and search through all possible word sequences.

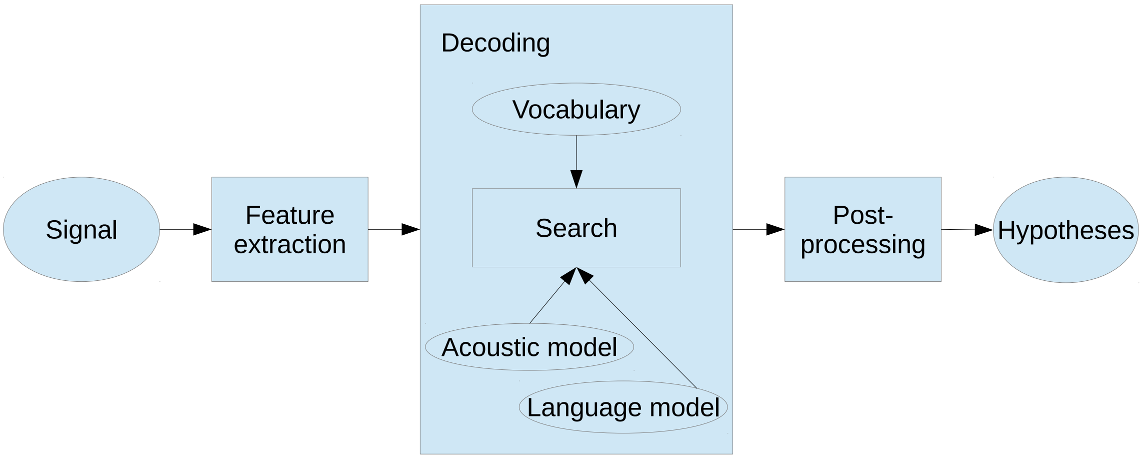


Figure 1.1.: LVSR

Further, in this section only speech recognition systems are discussed, that are based on statistical models. There is another type of systems, which are linguistically motivated and use grammar rules to do the recognition. Statistical systems use the Bayesian framework to find the best hypothesis:

$$h = \underset{W}{\operatorname{argmax}} P(W|X) = \underset{W}{\operatorname{argmax}} \frac{P(X|W)P(W)}{P(X)} = \underset{W}{\operatorname{argmax}} P(X|W)P(W)$$

$P(X|W)$ in this equation gives a measure of acoustic similarity of features X and a model representing the word sequence W, i.e. likelihood of a word sequence given the acoustic signal. As acoustic models Hidden Markov models (HMMs) are very popular. With help of algorithms for HMM probabilities of different sequences of phonetic units, can be computed, according to the trained models. The phonetic units are gained from word sequence with help of pronunciation dictionary.

$P(W)$ is an a-priory probability of the word sequence W. This probability distribution is called the language model. It means that the language model says how probable a given sequence is, according to the training data. In general this prediction depends on the whole word history:

$$P(W) = P(w_1, w_2, \dots, w_k) = \prod_{i=1}^{k} p(w_i|w_1, w_2, \dots, w_{i-1}) = \prod_{i=1}^{k} p(w_i|W_i)$$

where W_i is the word history of the word w_i:

$$W_i = w_1, w_2, \dots, w_{i-1}$$

Since the history space grows exponentially with the length of sequence, it would be not possible to get enough training data. A search through this whole space would be also computationally very expensive. That is why word histories are clustered into equivalent classes:

$$P(W) = P(w_1, w_2, \dots, w_k) = \prod_{i=1}^{k} p(w_i|\Phi(W_{i-1}))$$

Finding appropriate equivalence classifiers Φ and methods to estimate $p(w_i|\Phi(W_{i-1}))$ builds the research field of language modelling[CJ00].

One of the most popular approaches for language modelling is the n-gram model. For $n = 3$ the model is called a trigram model:

$$P(W) = P(w_1, w_2, \dots, w_k) = \prod_{i=1}^{k} p(w_i|w_{i-1}, w_{i-2})$$

n-gram language models have been very popular in speech recognition and machine translation for a long time. These models have a lot of advantages:

- they are easy to estimate
- fully data-driven estimation
- no extra information/parser required
- wide support
- good and robust performance
- advanced backoff techniques

These features make *n-gram* the first choice for language modelling.

The search process is also a very important part of the recognition process. It is not computationally possible to search through the whole search space. That is why pruning and advanced search techniques are needed. Popular search algorithms used in speech recognition are the Viterbi beam search and the A* algorithms.

1.2. Aplication of LVSR to Russian language

LVCSR of Russian language is complicated by two language features:

- high inflectionality
- non-strict word order

High inflectionality is introduced by the language morphology, which leads to high out of vocabulary (OOV) rates in spite of quite large vocabularies [Whi00]. Relations of vocabulary sizes and OOVs are given on Figure 1.2. The unique words, which are not in the pronunciation dictionary cannot be modelled within the system, it means they cannot be never recognized.

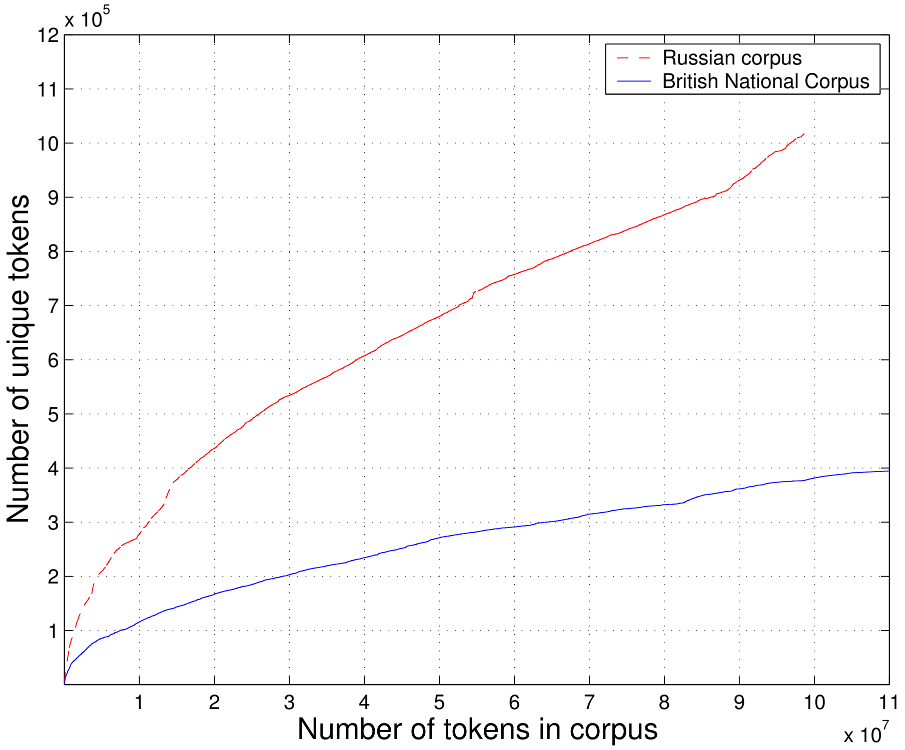


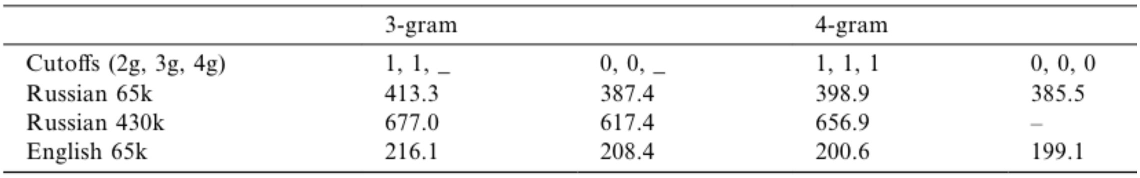
Figure 1.2.: Growth in vocabulary size against corpus size[Whi00].

Moreover, text corpora can be used less effective for language model training, because a lot of n-grams have not been seen often enough in the training. This introduces the problem of data sparsity. This problem becomes even bigger due to the “free" word order. A lot of correct word sequences (permutations) can not be modelled properly as they are not present in the training data.

The growing corpus and data sparsity make them noticeable in the perplexity of the language. The Table 1.3 gives perplexities of word trigram and 4-gram models for Russian and English.

To summarize the upper text, two problems can be defined, which make a standard speech recognizer less effective for Russian:

- OOV words cannot be recognized as they are not in the vocabulary by definition
- Not enough frequently seen n-gramms can not be modelled properly



	3-gram		4-gram	
Cutoffs (2g, 3g, 4g)	1, 1, _	0, 0, _	1, 1, 1	0, 0, 0
Russian 65k	413.3	387.4	398.9	385.5
Russian 430k	677.0	617.4	656.9	–
English 65k	216.1	208.4	200.6	199.1

Figure 1.3.: Perplexities of word trigram and 4-gram models with different cutoffs [WW03].

2. Related work

In 1 two features of Russian language were named: high inflecionality and non-strict word order, which make speech recognition more difficult. In this chapter related approaches chosen by other researches are reviewed. All reviewed works are divided into two categories. In the first category there are works, which deal with high out-of-vocabulary rates (OOVs). The second category is devoted to the problem of data sparsity.

High inflectionality is present not only in Russian but also in many other languages, which belong to Slavic, Uralic and Turkic language groups, e.g. Finnish, Estonian, Ukrainian, Polish, Turkish, Uzbek, etc. Therefore a number of approaches are proposed for these languages.

2.1. Search vocabulary

To address the problem of high OOV rate the most of authors propose to utilize morphological structure of words [Cre+07; EDSN10]. The sub-word units can be derived rules-based, e.g. with Porter stemming algorithm[Por01] or proceed with help of statistic [CL05a], mostly using *Minimum Description Length* principle[Ris78]. Several approaches combine both to achieve better results [Ber09]. For direct comparison of different methods of sub-words analysis see *Morpho Challenge* [Kur+10].

Several tools for morpheme/sub-words analysis are freely available. One of the most popular is *Morfessor* [CL05b; CL07] tool, developed within *Morph project* [MC12]. *Morfessor* can apply both MDL [CL05b] or MAP (Maximum a posteriori) methods and consequently belongs to the statistical based approaches. Another interesting tool is *Snowball Stemmer* [Por01], which is not a morpheme analysis tool, but a word stemmer. But it can successfully be applied to derive sub-word vocabulary. There are versions for different languages available. *Snowbal is the rule-based tool.* It uses a special programming language (named Snowball), to describe how to derive stems from word inflections.

But only using sub-word units is not sufficient to get better recognition results. In [Irc+01] a morpheme based recognition system for Czech language with a standard n-gram language model performs slightly worse, the recognition accuracy gets lower from 65.71% to 63.14%. An application of optimized language model in the rescoring pass raises the accuray to 70.38%. This optimization includes separate models for stems and endings prediction.

and is even being disputed [?]. Performance of several systems reported worse, when using sub-words, e.g. in [KP03] re-merging of distinct sub-words is applied to achieve better results. The suspicion could be an application of better language models.

2.2. Language modelling

Further in this section different language modelling techniques for inflectional languages are reviewed.

Because of the data sparsity standard n-gram language models perform poorer for high inflectional languages with non-strict or rather special word orders, in which words with strong cohesion can be located far from each other in comparison to the English language [WW03]. To take this in account many authors make use of some extra information for text corpora, such as part-of-speech tags, word-class tags, etc. [EDSN10; RAD10].

Sub-word units make, in addition to the data sparsity, word histories shorter. But an appropriate history length have an impact to the performance of the system[HPK09]. On the figure 2.1 shows how n-grams are distributed in Finnish.

	1-grams	2-grams	3-grams	4-grams	5-grams	6-grams	7-grams	8-grams	9-grams	10-grams	Total
						[×1000]					
Morph 2k (3-gram)	2	439	2940								3381
Morph 2k (Variable)	2	439	2727	13 230	15 681	9965	3799	1121	289	70	47 341
Morph 10k (3-gram)	10	1444	6920								8374
Morph 10k (Variable)	10	1444	6610	16 073	11 458	5385	1440	298	54	9	42 783
Morph 50k (3-gram)	50	2819	9932								12 801
Morph 50k (Variable)	50	2819	9406	14 691	9436	3748	1057	194	32	4	41 437
Word 500k (3-gram)	500	34 618	4745								39 863
Word 500k (10-gram)	500	34 665	4669	1280	355	103	30	9	3	1	41 615

Figure 2.1.: Distribution of n-grams in Finnish

In [CJ00] authors argue that also in English sometimes even a context of 7 words can not give enough information for the prediction of the next word. Therefore a syntactic structure of sentences is suggested as additional information. A *headword* relation is incorporated into the language modelling. An example parsing is given on the figure 2.2. The important point is that the model is applied in the first-pass of the decoding process, this implies that words can only be processed from left to right and force to use parsing only of the word history and not of the whole sentence. Furthermore, in this approach whole words are used for the modelling.

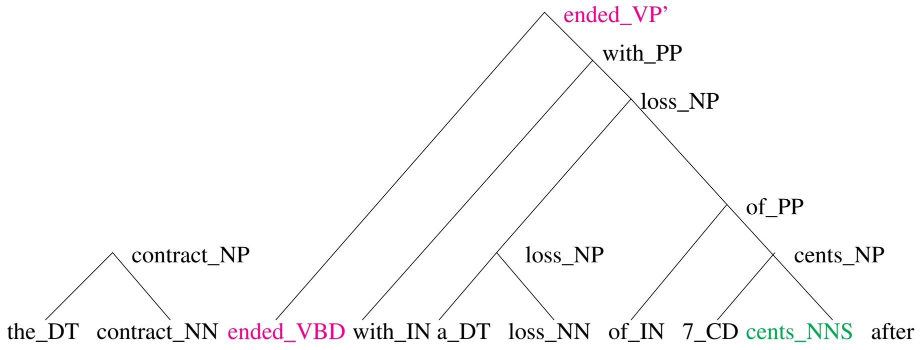


Figure 2.2.: In case of a trigram language model predicts the word “afrer" from a context “7 cents". The structured language model has a context “contract ended”, which intuitively gives a better prediction.

Another approach, that also use syntactic structure of sentences in English is described in [RAD10]. In this work the grammatical features are head-modifier relations between pairs of words together with the labels of the relationships. *Head-modifier* relation is similar to *headword* relation used in [CJ00]. The features are obtained with the dependency grammar. A parsing example is given on the figure 2.3.

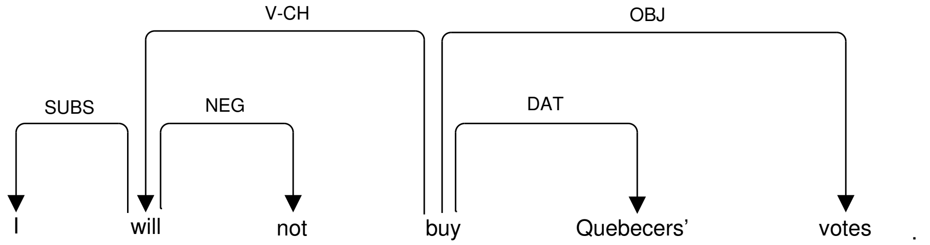


Figure 2.3.: An example English sentence parsed with dependency grammar.

An important difference to the work of Chelba and Jelinek is that Ruokolainen, Alumäe, and Dobrinkat apply their technique in the post processing, i.e. in the rescoring pass. In this case the whole sentence is available for the evaluation, so the syntactic knowledge can be better utilized. As a mathematical framework a *Maximum Entropy* approach is used. By the application of both n-gram and whole sentence maximum entropy language models a performance of 29.7% WER on the Wall Street Jurnal database[PB92] is achieved. The absolute reduction of WER accounts for 1.7%.

A good comparison of speech recognition techniques particularly for language modelling can be found in [Man+11]. In this work the state-of-art class based n-gram language model - *ModelM*[CC10] was outperformed by a neuronal network based syntactic model in the Arabic broadcast transcription. This syntactic model utilize additional syntactic and morphological features such as *head-word* relations and grammatical label of these relations. The improvements totals in the absolute reduction of 0.9% WER by achieving a WER of 7.4%.

To deal with complex morphology of Arabic language El-Desoky, Schlüter, and Ney combine in [EDSN10] sub-words based vocabulary and advanced language modelling. Morphological analysis is done with MADA toolkit [NHR09]. For the language modelling a *Factored Language Modelling* (FLM) framework [KBD07] was applied[1]. As an additional information used in the language model are features got from MADA tool. These are *word*, *lexeme*, *morph* and *pattern*. The *pattern* is defined here as word after subtracting root. The best FLM system decreases WER from 16.5% for traditional full-words system to 15.9% for FLM.

An interesting idea is proposed in [YB09]. The Turkish language was modelled with so called *flex-grams*, which allow skipping several parents and use later grams in the history to estimate a probability of the current word. The language model is represented by tuples of dependency offsets $d = [d_1, d_2, \ldots, d_{n-1}]$:

$$P(W) = \prod_{i=1}^{k} p(w_i | w_{i+d_1}, w_{i+d_2}, \ldots, w_{i+d_{n-1}})$$

The important point is that the model use sub-words derived with *morfessor* as vocabulary units and the best perplexity reduction (about 25%) was achieved by applying different models form stems and endings, namely $d = [-1]$ for endings and $d = [-2]$ for stems. Intuitively, it means that stems give information to predict coming ending, and the previous stem is used to predict current stem.

To alleviate the problem of data sparsity decision trees (DT) were introduced to the language modelling. Random forest is a classification model constructed of multiple decision

[1] The FLM framework can be seen as an extension to n-gram language model, which incorporate additional information sources and generalize the backoff procedure

trees. The idea behind is to randomize a tree growing algorithms in order to achieve a better generalization and overcome data sparsity. Randomized DTs based Random Forests (RF) were being studied in recent [Opa08; XJ04]. In [Opa08] Oparin applies RF language models on university lectures in Czech language. Word form, word lemma, word stem, part-of-speech tag were used as additional linguistic features. The RF language model, being applied in the second pass, i.e. for n-best list rescoring, shows better performance in comparison to standard trigram model. The relative perplexity reduction is reported up to 10% and WER improvement over the trigram model is up to 3.4%

2.3. In this work

In this work a sub-word based search vocabulary and a sub-words based n-gram language model is used for the baseline speech recognition. To generate sub-words text corpus the Snowball stemmer[Por01] is utilized. The stemmer for Russian is distributed within the Snowball package. The text corpus is preprocessed to resolve *ye-yo* ambiguity (See Appendix A). In [Opa08] a similar preprocessing is done: "Presence or absence of ë for a particular stem or inflexion is inferred from linguistic information: namely, POS, class of a stem and inflexion, accent paradigm and values of grammatical categories(which are detected by means of the inflexion and context information, if needed)." Here a vocabulary approach together with a special n-gram model, trained on another text corpus with ë, is taken.

An Maximum Entropy endings discrimination model is introduced for the rescoring pass. This model rearranges hypotheses in according to endings placement. To make training resource-effective models for each ending are trained separately.

To address the problem of shorter contexts in case of sub-words based n-gram model, another full-words n-gram model is applied in the rescoring pass after sub-words merging.

To achieve the final bast score both Maximum Entropy endings discrimination and full-words n-gram models are combined together.

3. Sub-Word Based Vocabulary

One of the important components of any LVSR system is the search vocabulary. That is the list of words, which can be recognized. Due to the high inflectionality high OOV rates can be observed for Russian. One common approach to get the OOVs into the system is to use smaller units instead of words. This make it possible to create OOV words by combining the sub-word units.

There are two classes of algorithms for word decomposition:

- Linguistically motivated units - morpheme
- Statistically calculated clusters of letters

In this work a type of linguistically motivated approaches is used. A rule-based stemming approach.

The current chapter describes also the generation of the sub-word based pronunciation dictionary used for this work.

3.1. Word decomposition

For this work a Snowball [Por01] stemmer was utilized. Snowball is a small string processing language designed for creating stemming algorithms. A stemmer for Russian language is distributed with the package. The stemmer is not a tool for morpheme analysis, but a word stem derivation tool. Therefore, the output of this tool needs to be processed to split up words into subunits. For a given word the stemmer returns a stem. Endings can be than gained by string comparison. Here is a small example for the phrase "необходимое условие" (necessary conditions)

необходимое	→	необходим	→	ое
условие	→	услов	→	ие

The same phase in another grammatical case gives another decomposition:

необходимого	→	необходим	→	ого
условия	→	услов	→	ия

The same phrase in plural gives yet another one:

необходимые	→	необходим	→	ые
условия	→	услов	→	ия

To create a sub-word search vocabulary, a large 1M vocabulary, generated from a training corpus was processed by Snowball. After processing, the size of the vocabulary is reduced to 40k. The count of endings lies under 500. Not only the size of the vocabulary is reduced, but its coverage of words is also improved, i.e. the OOV is reduced. The drawback is that the search units got shorter, which can lead to stronger acoustical confusability.

Extra attention was paid to special cases, such as:

- word compounds that are written with dash
- endings which cannot be acoustically modelled correctly

In the first case it is possible that both parts of such compounds have their own endings. That is why the compounds are split into their parts and processed both by the stemmer. The word decomposition for the search vocabulary differs from the one used for the maximum entropy language modelling for this case. To simplify the merging of sub-words after decoding every word part after the first stem is marked as an ending, e.g.:

где-нибудь → гд ~е ~-нибудь

For acoustic modelling it is not important if a sub-word is marked as a stem or ending. The only important thing is that the original utterance can be reconstructed easily. For that reason the null-ending was utilized for the words, which do not have an explicit ending, but only for the language modelling, e.g.:

где-нибудь → гд ~е _нибудь ~#

For more information, why null-endings are used for the language modelling, refer **??**.

3.2. Pronunciation dictionary

Another problem that occurs when using sub-words as search units is the generation of their pronunciation. Because of different rules (see Appendix **??**) it is difficult to generate the correct pronunciation for sub-words if the context is not given a-priori. That is why an grapheme-based approach is chosen in this work. The idea is to use letters as smallest pronunciation units and rely on statistical polyphone clustering to take into account the context information.

4. Maximum entropy modelling

This chapter is devoted to the maximum entropy language modelling. In the first section theoretical basis of the maximum entropy modelling is discussed. Formal description of features used for the modelling and the training process are explained. The second section focuses on the application of maximum entropy modelling to the language modelling, especial for the evaluation of word endings in the Russian language.

4.1. Theoretical framework

Maximum entropy modelling bases on the following concepts:

- Model constrained by features.
- Maximization of conditional entropy.
- Application different algorithms to estimate parameters.

4.1.1. Features

As it was already mentioned an additional information is wanted to be incorporated in the language modelling to make the model better. For a standard n-gram language model this additional information are word contexts, e.g. bigrams, trigrams, etc. But there a lot of features, which can be useful for the language modelling purposes. These could be part of speech (POS) tags, different grammatical categories, topic information. Such features can be represented by binary feature functions or indicator functions. Here is an example of a bigram binary feature:

$$f_1(x,y) = \begin{cases} 1, & if \quad y = "day" \quad and \quad x = "nice" \\ 0, & otherwise \end{cases}$$

The function f returns 1 for the word y and it's context x, if y and x build a bigram "nice day".

Now, according to the training data, a mean value of this feature can be calculated by:

$$\mu(f_1) = \sum_{x,y} p_e(x,y) f_1(x,y) = \sum_{x,y} relfreq(x,y) f_1(x,y)$$

If the training data is sufficiently large, the mean value represents an expected value of the real distribution:

$$\mathbb{E}(f_1) = \sum_{x,y} p(x,y) f_1(x,y)$$

Now, the model is requested to be unbiased , i.e. to have the same expected value for the feature f_1:

$$\sum_{x,y} p_e(x,y) f_1(x,y) = \sum_{x,y} p_m(x,y) f_1(x,y),$$

where $p_m(x,y)$ is a distribution given by the model.

In general we are interested in modelling of $p(y|x)$ and not of $p(x,y)$. That is why the final constraint equations for the feature f_1 looks like:

$$\sum_{x,y} p_e(x,y) f_1(x,y) = \sum_{x,y} p_e(x) p_m(y|x) f_1(x,y),$$

For each feature such a constrained is created.

4.1.2. Maximization of conditional entropy

Depending on the features, a set of distributions which are conform with all the constrains, consist sometimes of lots of distributions and can even be infinite. The problem is to select the best one.

One approach comes from the information theory and is based on the concept of conditional entropy:

$$H(Y|X) = -\sum_{\substack{x \in X,\\ y \in Y}} p(x,y) \log p(y|x)$$

The idea of the maximum entropy modeling is to choose that model, which maximize the conditional entropy of labels y given an information x (e.g. context):

$$p_{me} = \arg\max_{p_m} H(p_m)$$

In simple words it means that the model makes no further assumptions but the given features.

With help of Lagrange multipliers, which used to solve this constrained optimization problem, it can be shown that the resulting probability distribution has the parametric form:

$$p_{me}(\lambda) = \frac{1}{Z(x)} \exp\left(\sum_i \lambda_i f_i(x,y)\right),$$

where $f_i(x,y)$ are binary feature functions. λ_i are weight factors - parameters of the model. $Z(x)$ is a normalization factor.

4.1.3. Training

A number of algorithms can be used for estimating parameters of maximum entropy model. There are both special methods, such as *Generalized Iterative Scaling*[DR72], *Improved Iterative Scaling*[DPDPL97], and general purpose optimization techniques, such as gradient ascent, conjugate gradient, quasi-Newton methods. For a comparison of algorithms for maximum entropy parameter estimation refer [Mal+02].

"Surprisingly, the widely used iterative scaling algorithms perform quite poorly, and for all of the test problems, a limited memory variable metric algorithm outperformed the other choices."[Mal+02]

The *Limited-memory BFGS* a limited memory variation of *BFGS*[Avr03; Bon03], which is an implementation of variable metric method, is used for this work with help of *CRF++* Toolkit[Kud05].

4.2. Application to the modelling of Russian

In 3 it was pointed to the one of the drawbacks of using sub-words for the speech recognition, namely, that short words can be confused more easily. This applies to endings in Russian, as they are commonly very short. The confusion gets worse, because of the frequent vowels reduction in the Russian language. In this work the maximum entropy modelling is directed to this problem. So, the model is used to solve a classification task by evaluating of the given endings for every word. Endings are suggested by hypotheses in n-best lists.

4.2.1. Feature selection

The correct word ending depends on the following grammatical categories of the corresponding word and words in its context:

- Part of speech
- Gender (for nouns, verbs, pronouns, . . .)
- Case (role of the word in the sentence)
- Tense (for verbs)

Unfortunately, this meta-information is not available while decoding. Conversely, endings represent these grammatical categories. So, to explore the grammatical patterns one can use endings. In other words by looking how endings and words play together in the context, the grammatical patterns can be evaluated. The null-ending, which doesn't matter for the acoustic modelling plays here an important role. It indicate in the same way for the grammar as the other. These ideas lead to the following features for the maximum entropy language modelling:

$\forall k, p \in \mathit{Endings}, \forall s \in \mathit{Stems}$ add features:

$$f(context, e) = \begin{cases} 1, & if \quad e = k \quad and \quad context[k] = p \\ 0, & otherwise \end{cases}, \tag{4.1}$$

and

$$f(context, e) = \begin{cases} 1, & if \quad e = k \quad and \quad context[k] = s \\ 0, & otherwise \end{cases}, \tag{4.2}$$

where $context[k]$ means k^{th} position in the context. Notice, that two features, which have the same ending or word in the context, but at different position, differ. This make sense, as the patterns differ in this case. The context can be defined differently. For this work the context spans for three previous words, one word coming after the current one. Here is a small example:

s_{-5}	e_{-5}	как	~#
s_{-4}	e_{-4}	подчеркнул	~#
s_{-3}	e_{-3}	офицер	~#
s_{-2}	e_{-2}	полиц	~ии
s_{-1}	e_{-1}	жёстк	~ие
s_0	e_0	мер	~ы
s_1	e_1	не	~#
s_2	e_2	применя	~лись

In this case e_0 is to be evaluated. The following features are extracted:

- $e_0 =$ ”~ы”, $s_{-3} =$ ”офицер”
- $e_0 =$ ”~ы”, $s_{-2} =$ ”полиц”
- $e_0 =$ ”~ы”, $s_{-1} =$ ”жёстк”
- $e_0 =$ ”~ы”, $s_0 =$ ”мер”
- $e_0 =$ ”~ы”, $s_1 =$ ”не”
- $e_0 =$ ”~ы”, $e_{-3} =$ ”~#”
- $e =$ ”~ы”, $e_{-2} =$ ”~ии”
- $e =$ ”~ы”, $e_{-1} =$ ”~ие”
- $e =$ ”~ы”, $e_{-1} =$ ”~#”

Different hypotheses in the corresponding n-best list could propose different endings for e_0, than the model evaluates hypotheses and choose the most probable one.

4.2.2. Feature extraction

The feature extraction system bases on the same word stemmer as for generation a search vocabulary used in 3. The only difference is that for feature extraction the null-ending is an explicit one. It is marked with ~# sequence. The preprocessing of the training data and of the tests data is the same: Text is arranged with one word per line. Sentences are processed separately.

4.2.3. Scoring

The evaluation of a single ending gives a conditional probability of it given a context. The total score of the sentence is calculated as a sum of log probabilities of every ending in the sentence even for the null-ending:

$$score = \sum_{i}^{n} \log_{10} p(e_i),$$

where n is the context length.

4.2.4. Resource-effective training

By doing a small calculation is becomes clear, that it is computationally not feasible to train one model to make an evaluation of all endings, as to much *primary memory* is needed to keep feature weights. In according to (4.1) and (4.2):

$$\#Features \simeq \#Voc \times ContextLength \times 2$$

By 40k vocabulary and features with *double* values (8 bytes) it is $\simeq$ 720G. Furthermore, memory for calculation is needed.

That is why, for this work, the similar technique as in [Med+11; MHN09] is used. The idea is to train separate model for every ending. This every model evaluates than only two classes: the ending, which the models stands for and all other endings, i.e.:

- $\mathcal{M}_{\sim\#}$ decides between ”~#” and all other endings
- $\mathcal{M}_{\sim \text{ии}}$ decides between ”~ии” and all other endings
- $\mathcal{M}_{\sim \text{ие}}$ decides between ”~ие” and all other endings
- ...

In testing every appropriate model was used to evaluates the sentence. In this case "appropriate" excludes models, which represent endings absent in the sentence.

5. Experiments

In this chapter the speech recognition system setup, experiments with re-scoring of the on n-best lists and their results are described. In the first section tools which were used during the experiments are listed. Further, in the next section, a dataset for training and testing is described. The experiment setup is given in the third section. Finally, the experiments and their results are described and discussed in the last fourth section.

5.1. Tools

The automatic speech recognition engine that we use in this work is JRTk (Janus Recognition Toolkit) [Fin+97], which is developed and maintained by the Interactive Systems Laboratories at two sites: Karlsruhe Institut of Technology, Germany and Carnegie Mellon University, USA.

Sub-words splitting up are done with help of Python library for the Snowball stemmer[Por01]: *PyStemmer*. The same scripts were used for generation of search vocabulary, sub-words n-gram training corpus and maximum entropy endings discrimination training corpus.

The language modelling tool which is used for the n-gramm of both sub-words and full-words language models is the SRI language modelling toolkit [Sto+02]. It was applied also for the re-scoring experiments. The *ngram* utility from this toolkit was utilized for this purposes.

For the training of maximum entropy language model a CRF++ tool [Kud05] was utilized. This tool, as the name says, can build conditional random fields (CRF) models, which is equivalent to the maximum entropy models, if not using bigram features. So, CRF models can be seen as an generalization of maximum entropy models to the contexts of features.

Oracle re-scoring is done with help of the *sclite* utility from the NIST SCTK Scoring Toolkit[Sct].

5.2. Dataset

The dataset used for this work includes training data for the acoustic models, training data for language models and the test data for the speech recognition and n-best-list re-scoring experiments.

The acoustic training data accounts for about 620 hours broadcast news and radio talks acquired within the *QUAERO*[Qua] project. Another dataset, which consist of read speech mostly in touristic and medical speech domains, provided by *Mobile Technology GmbH*[Mte], was divided into two parts. One part with about 60 hours was used for the acoustic model training. The other part, which accounts for about 3 hours, was used for testing. All the recordings have variety of environment noises at different signal-to-noise ratios with the sampling rate of 16kHz. There are both man and woman voices in the recordings.

Both sub-words based and full-words n-gram and maximum entropy language models were trained on the same text corpus. It accounts for 156M tokens. The text was crawled from the Internet forums in touristic and medical speech domains. Before using text was passed through *yofication* tool to resolve ye-yo ambiguity, described in Appendix **??**. This tool was developed within this work. The manner of functioning is described in Appendix A.

Table **??** summarize used datasets.

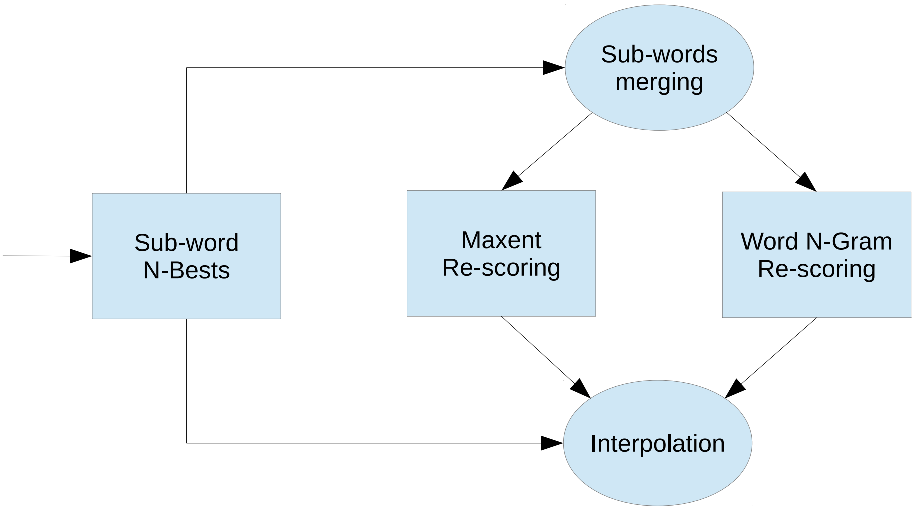
AM training	Broadcast news & radio	620 hours
AM training	Read speech (touristic + medical)	60 hours
LM training	Web forums (touristic + medical)	156M words
Testing	Read speech (touristic + medical)	3 hours

Table 5.1.: Overview of the data used for testing and AM training

5.3. Setup

The experiments flow is shown on the figure 5.3.

Figure 5.1.: LVSR



For each utterance in the test data 100 best hypotheses are generated by a speech recognition system. This system uses sub-words search vocabulary and sub-words based 4-gram model. The scores from the system are a weighted sum of the acoustic model and language model score. The amount 100 for n-bests lists was chosen with no special reason, but seems to be a good choice for this experimental setup, since the oracle WER accounts to 12.5%. Oracle means, in this context, choosing the best hypotheses according to the references.

Sub-words in n-best lists are than merged to full-words. All further processing applies to the full-word based n-best lists.

Score combination is a sum between recognition system score and a weighted score from the re-scoring system. Both scores are logarithmic:

$$S_{total} = S_{reco} + wS_{rescore}$$

5.3.1. Re-scoring with Maxent Endings-Discrimination model

The re-scoring with the maximum entropy model for endings discrimination happens as follow: Each hypotheses in each n-best list are re-scored with all ending models, i.e. models, for which the associated endings are present. Scores from each model are combined to the maximum entropy score.

5.3.2. Re-scoring with Word N-gram model

As a score of full-words 4-gram language model the absolute value of the log-probability of each hypothesis, returned by *ngram* tool, is taken.

5.4. Results

Table 5.4 shows the scores of the full-words 4-gram and the maximum entropy endings discrimination language models from the re-scoring of 100-best lists. These scores are just language model scores without any acoustical score. The maximum entropy model clearly outperforms the n-gram model by more than 7% in WER.

ME endings	Words 4-gram
40.0	47.7

Table 5.2.: Language model score of ME endings discrimination model and full-words 4-gram model.

Combined scores from the recognition system, maximum entropy endings discrimination and full-words 4-gram models are depicted on Figure 5.4. The experiments with different weight factors from 0.1 to 20 shows, that the both models improve the baseline system, though by different weights. The best improvement of the n-gram model comes to 25.5% WER with the improvement of 0.4% WER by weights 2 and 3, while the maximum entropy models achieves 1.2% improvement in WER. It gets 24.7% WER. In this experiments the maximum entropy models shows not only better but also a robuster performance as the n-gram model. Best results are achieved by weights 6, 7 ,8, 9, 10.

Curve of the graphics are intuitive for re-scoring experiments. By the weight 0, results are the same as of the recognition system, and same as language model scores for sufficiently large weights.

Table 5.4 represents the same experiments with numbers.

Additional experiment shows that improvements achieved by maximum entropy and n-gram language models are orthogonal, i.e. an application of both models sums the improvements. Table 5.4 depicts accuracies of baseline system, i.e. the recognition system, and the system with both models applied. The difference totals up to 1.6% WER. So, the best result comes to 24.3% WER.

Figure **??** figure compares accuracy improvements to the improvement of the oracle re-scoring system, which should represent the maximal gain, which could be achieved by n-best list re-scoring of this system.

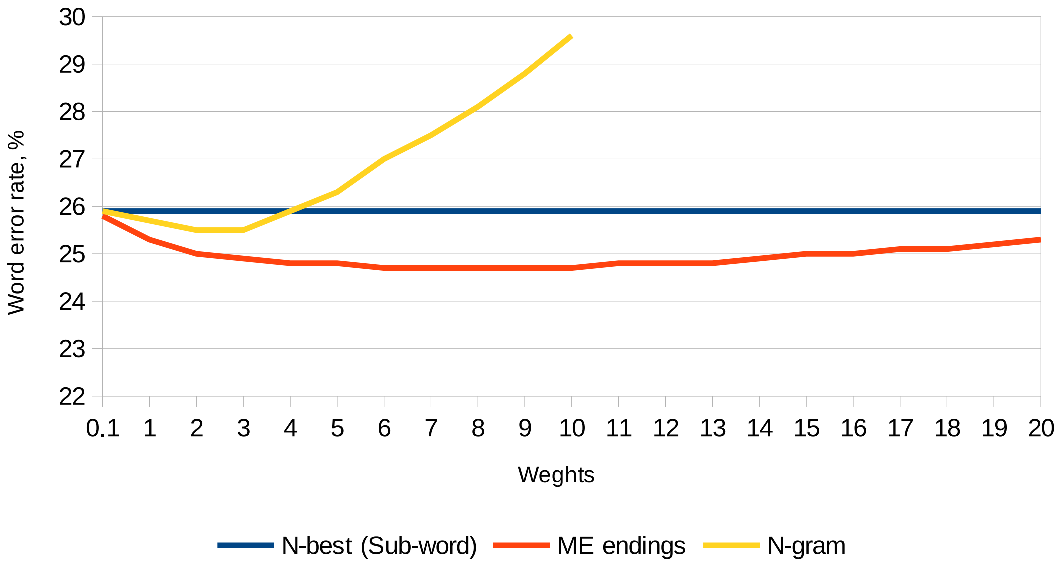


Figure 5.2.: Combined results of recognition and re-scoring systems.

Weight	ME endings	Δ	Words n-gram	Δ
0.1	25.8	0.1	25.9	0.0
1	25.3	0.6	25.7	0.2
2	25.0	0.9	**25.5**	**0.4**
3	24.9	1.0	**25.5**	**0.4**
4	24.8	1.1	25.9	0.0
5	24.8	1.1	26.3	-0.4
6	**24.7**	**1.2**	27.0	-1.1
7	**24.7**	**1.2**	27.5	-1.6
8	**24.7**	**1.2**	28.1	-2.2
9	**24.7**	**1.2**	28.8	-2.9
10	**24.7**	**1.2**	29.6	-3.7
11	24.8	1.1	-	-
12	24.8	1.1	-	-
13	24.8	1.1	-	-
14	24.9	1.0	-	-
15	25.0	0.9	-	-
16	25.0	0.9	-	-
17	25.1	0.8	-	-
18	25.1	0.8	-	-
19	25.2	0.7	-	-
20	25.3	0.6	-	-

Table 5.3.: Combined results of recognition and re-scoring systems.

Full-word system	25.7%
Sub-word system	25.9%
+ Maximum entropy	24.7%
+ Word n-gram	24.3%

Table 5.4.: Comparison of the sub-word recognition system, system with both language models applied simultaneously.

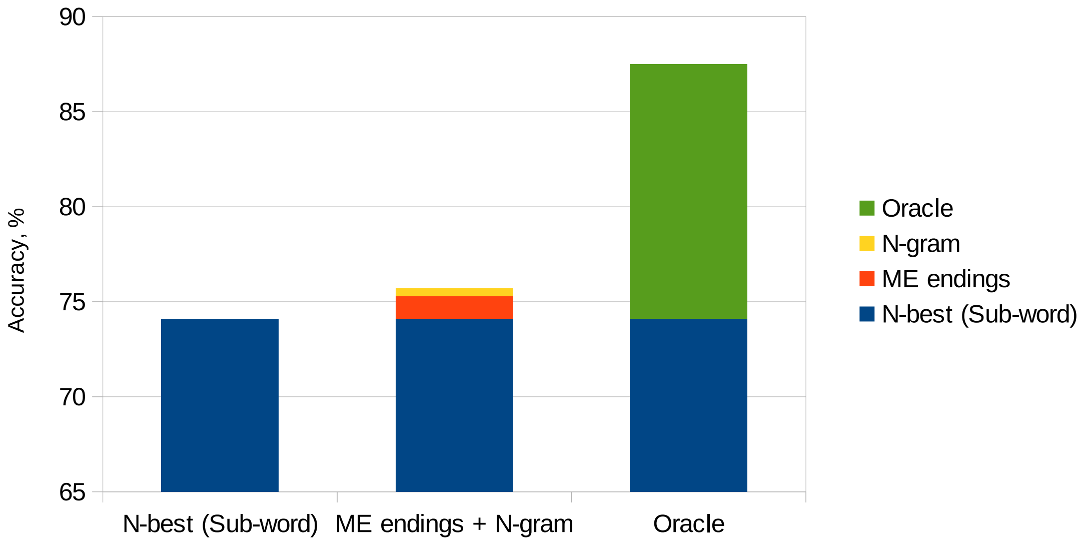


Figure 5.3.: Comparison of the sub-word recognition system, system with both language models applied simultaneously and the oracle re-scoring system.

6. Conclusion

Results of the experiments done within this work show, that applied models both achieve significant improvements, when applied either separately or combined together. The introduced Maximum Entropy endings discrimination language model achieves an absolute improvement of 1.2% WER (relative improvement 4%). Full-words n-gram model increases performance by absolute amount of 0.4% WER. Both models seems to correct different categories of errors, as by combined application the improvement totals to 1.6% WER absolute (6% relative).

There is no principal restriction for an application of introduced models in the first pass of speech recognition. So, if it is preferred, with some extensions they could be applied in the decoding process.

Appendices

A. *Ye-Yo* ambiguity and Yofication

Letter "ё" is the seventh letter in Russian alphabet. This is the only letter with diacritics in the alphabet. This diacritics serves to differentiate it from "е". The status of "ё" is now ambiguous, as it is often replaced by "е" in written language and is not obligatory in Russian orthography. "ё" is always written in cases, when necessary to distinguish between words, having only difference in "е" or "ё", and if it not obvious from the context, e.g. "все (everybody) - всё" (everything). However, the disambiguation is obligatory to produce correct word transcriptions.

Ye-yo ambiguity in Russian affects ASR negatively. For example, the word pair "все - всё" takes the second place in the list of the most confusing words, see Table A. To solve this problem a mixture of lookup dictionary and context based approaches is introduced in this section. The idea is to use a dictionary of ye-yo confusing words to build a word lattice and, then, to re-score the lattice with a language model. This approach achieves 98% accuracy on a small test corpus.

Figure A.1.: List of the most confusing words

1:	32	→	ну	⇔	но
2:	29	→	все	⇔	всё
3:	9	→	ни	⇔	не
4:	7	→	в	⇔	на
5:	7	→	они	⇔	не
6:	7	→	эта	⇔	это

...

A.1. Lookup dictionary

The lookup dictionary, that we use here is based on [Iva12]. It consists of 139k words. Hi- and lower-case spelling is differentiated. Words, which can be written both with е and ё, are specially marked. It is important to notice that words with hyphen are not included, e.g. всё-таки, which could be an improvement to the dictionary. In this work such words are simply split into two parts before prosessing, which can be sub-optimal in some cases.

For yofication all unambiguous words are substituted with the corresponding entry of the ë-dictionary. The lookup is case insensitive. The ambiguous words represent word lattices of each sentence. The lattices are then re-scored with a language model to find the best hypothesis.

A.2. Context

Since context knowledge can be crucial for ambiguous cases with ë, a language model is needed for disambiguation. Therefore an ngram language model is employed here. The case-insensitive ngram model was trained on a corpus (12M words) of newstext from "Литературная газета" [Lgz], which is available online. The text uses ë more or less consistently. The training corpus is quite small, and more data would improve the model.

Experiments show that the best performance is achieved by a trigram language model. It performs better than a 4-gram model. The small training corpus should be the reason. A bigram has not enough context information to solve distinct cases. The evaluation was done on a very small corpus, and the results should be validated on a bigger one, which was not the focus of this work.

Bibliography

[Avr03] Mordecai Avriel. *Nonlinear programming: analysis and methods.* Courier Dover Publications, 2003.

[Ber09] Delphine Bernhard. „MorphoNet: exploring the use of community structure for unsupervised morpheme analysis“. In: *Proceedings of the 10th cross-language evaluation forum conference on Multilingual information access evaluation: text retrieval experiments.* CLEF'09. Corfu, Greece: Springer-Verlag, 2009, pp. 598–608. ISBN: 3-642-15753-X, 978-3-642-15753-0. URL: http://dl.acm.org/citation.cfm?id=1887364.1887450.

[Bon03] J Frédéric Bonnans. *Numerical optimization: theoretical and practical aspects: with 26 figures.* Springer-Verlag New York Incorporated, 2003.

[CC10] S.F. Chen and S.M. Chu. „Enhanced word classing for model m“. In: *Proceedings of Interspeech.* 2010, pp. 1037–1040.

[CJ00] C. Chelba and F. Jelinek. „Structured language modeling“. In: *Computer Speech & Language* 14.4 (2000), pp. 283–332.

[CL05a] M. Creutz and K. Lagus. „Inducing the morphological lexicon of a natural language from unannotated text“. In: (2005).

[CL05b] Mathias Creutz and Krista Lagus. *Unsupervised morpheme segmentation and morphology induction from text corpora using Morfessor 1.0.* Helsinki University of Technology, 2005.

[CL07] Mathias Creutz and Krista Lagus. „Unsupervised models for morpheme segmentation and morphology learning“. In: *ACM Transactions on Speech and Language Processing (TSLP)* 4.1 (2007), p. 3.

[Cre+07] M. Creutz et al. „Morph-based speech recognition and modeling of out-of-vocabulary words across languages“. In: *ACM Transactions on Speech and Language Processing (TSLP)* 5.1 (2007), p. 3.

[DPDPL97] Stephen Della Pietra, Vincent Della Pietra, and John Lafferty. „Inducing features of random fields“. In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 19.4 (1997), pp. 380–393.

[DR72] John N Darroch and Douglas Ratcliff. „Generalized iterative scaling for log-linear models“. In: *The annals of mathematical statistics* 43.5 (1972), pp. 1470–1480.

[EDSN10] A. El-Desoky, R. Schlüter, and H. Ney. „A hybrid morphologically decomposed factored language models for Arabic LVCSR“. In: *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics.* Association for Computational Linguistics. 2010, pp. 701–704.

[Fin+97] Michael Finke et al. „The karlsruhe-verbmobil speech recognition engine“. In: *Acoustics, Speech, and Signal Processing, 1997. ICASSP-97., 1997 IEEE International Conference on.* Vol. 1. IEEE. 1997, pp. 83–86.

[HPK09] T. Hirsimaki, J. Pylkkonen, and M. Kurimo. „Importance of high-order n-gram models in morph-based speech recognition“. In: *Audio, Speech, and Language Processing, IEEE Transactions on* 17.4 (2009), pp. 724–732.

[Irc+01] P. Ircing et al. „On large vocabulary continuous speech recognition of highly inflectional language-Czech“. In: *Seventh European Conference on Speech Communication and Technology.* 2001.

[Iva12] Vladimir Ivanov. *База слов на "Ё" для програмы Yo. Версия: 1.47 @online.* 2012. URL: http://vgiv.narod.ru/yo/yo.html.

[KBD07] K. Kirchhoff, J. Bilmes, and K. Duh. „Factored language models tutorial“. In: (2007).

[KP03] O.W. Kwon and J. Park. „Korean large vocabulary continuous speech recognition with morpheme-based recognition units“. In: *Speech Communication* 39.3 (2003), pp. 287–300.

[Kud05] Taku Kudo. *CRF++: Yet Another CRF Tool Kit.* Apr. 2005. URL: http://code.google.com/p/crfpp/.

[Kur+10] Mikko Kurimo et al. „Morpho Challenge competition 2005–2010: evaluations and results“. In: *Proceedings of the 11th Meeting of the ACL Special Interest Group on Computational Morphology and Phonology.* Association for Computational Linguistics. 2010, pp. 87–95.

[Lew09] M. Paul Lewis, ed. *Ethnologue: Languages of the World.* Sixteenth. Dallas, TX, USA: SIL International, 2009.

[Lgz] *Литературная газета @online.* 2012. URL: http://www.lgz.ru/.

[Mal+02] Robert Malouf et al. „A comparison of algorithms for maximum entropy parameter estimation“. In: *Proceedings of the sixth conference on natural language learning (CoNLL-2002).* 2002, pp. 49–55.

[Man+11] L. Mangu et al. „The IBM 2011 GALE Arabic speech transcription system“. In: *Automatic Speech Recognition and Understanding (ASRU), 2011 IEEE Workshop on.* IEEE. 2011, pp. 272–277.

[MC12] Krista Lagus Mathias Creutz. *Morpho project @ONLINE.* Apr. 2012. URL: http://www.cis.hut.fi/projects/morpho/.

[Med+11] Mohammed Mediani et al. „The kit english-french translation systems for iwslt 2011“. In: *Proceedings of the eight International Workshop on Spoken Language Translation (IWSLT).* 2011.

[MHN09] Arne Mauser, Saša Hasan, and Hermann Ney. „Extending statistical machine translation with discriminative and trigger-based lexicon models“. In: *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1.* Association for Computational Linguistics. 2009, pp. 210–218.

[Mte] *Mobile Technologies GmbH.* URL: http://www.jibbigo.com/.

[NHR09] Owen Rambow Nizar Habash and Ryan Roth. „MADA+TOKAN: A Toolkit for Arabic Tokenization, Diacritization, Morphological Disambiguation, POS Tagging, Stemming and Lemmatization“. In: *Proceedings of the Second International Conference on Arabic Language Resources and Tools.* Ed. by Khalid Choukri and Bente Maegaard. Cairo, Egypt: The MEDAR Consortium, 2009. ISBN: 2-9517408-5-9.

[Opa08] I. Oparin. „Language Models for Automatic Speech Recognition of Inflectional Languages“. PhD thesis. University of West Bohemia, 2008.

[PB92] Douglas B Paul and Janet M Baker. „The design for the Wall Street Journal-based CSR corpus“. In: *Proceedings of the workshop on Speech and Natural Language*. Association for Computational Linguistics. 1992, pp. 357–362.

[Por01] M. Porter. *Snowball: A language for stemming algorithms*. 2001.

[Qua] *Quaero is a European research and development program*. Mar. 2008. URL: `http://www.quaero.org/`.

[RAD10] T. Ruokolainen, T. Alumäe, and M. Dobrinkat. „Using Dependency Grammar Features in Whole Sentence Maximum Entropy Language Model for Speech Recognition“. In: *Proceedings of the 2010 conference on Human Language Technologies–The Baltic Perspective: Proceedings of the Fourth International Conference Baltic HLT 2010*. IOS Press. 2010, pp. 73–79.

[Ris78] J. Rissanen. „Modeling by shortest data description“. In: *Automatica* 14.5 (1978), pp. 465 –471. ISSN: 0005-1098. DOI: `10.1016/0005-1098(78)90005-5`. URL: `http://www.sciencedirect.com/science/article/pii/0005109878900055`.

[Sct] *NIST Speech Recognition Scoring Toolkit*. URL: `http://www.nist.gov/speech/tools/`.

[Sto+02] Andreas Stolcke et al. „SRILM-an extensible language modeling toolkit“. In: *Proceedings of the international conference on spoken language processing*. Vol. 2. 2002, pp. 901–904.

[Whi00] E.W.D. Whittaker. „Statistical language modelling for automatic speech recognition of Russian and English“. In: *Daktaro disertacija, Cambridge University Engineering Department, Cambridge* (2000).

[WW03] E.W.D. Whittaker and PC Woodland. „Language modelling for Russian and English using words and classes“. In: *Computer speech & language* 17.1 (2003), pp. 87–104.

[XJ04] P. Xu and F. Jelinek. „Random forests in language modeling“. In: *Proc. EMNLP*. 2004.

[YB09] D. Yuret and E. Biçici. „Modeling morphologically rich languages using split words and unstructured dependencies“. In: *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*. Association for Computational Linguistics. 2009, pp. 345–348.