We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

6,900
Open access books available

185,000
International authors and editors

200M
Downloads

Our authors are among the

154
Countries delivered to

TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities



WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com



Chapter 11

Incorporating Grammatical Features in the Modeling of the Slovak Language for Continuous Speech Recognition

Ján Staš, Daniel Hládek and Jozef Juhár

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/48506

1. Introduction

The task of creation of a language model consists of the creation of the large-enough training corpus containing typical documents and phrases from the target domain, collecting statistical data, such as counts of word n-tuples (called n-grams) from the a collection of prepared text data (training corpus), further processing of the raw counts and deducing conditional probabilities of words, based on word history in the sentence. Resulting word tuples and corresponding probabilities form the language model.

The major space for improvement of the precision of the language model is in the *language model smoothing*. Basic method of the probability estimation, called *maximum likelihood* that utilizes n-gram counts directly obtained from the training corpus is often insufficient, because it results zero probability to those word n-grams not seen in the training corpus.

One of the possible ways to update n-gram probabilities lies in the incorporation of the grammatical features, obtained from the training corpus. Basic methods of the language modeling work just with sequences of words and does not take any language grammar into account. Current language modeling techniques are based on the statistics of the sequences of words in the sentences, obtained from a training corpora. If the information about the language grammar have to be included in the final language model, it had to be done in a way that is compatible with the statistical character of the basic language model. More precisely, this means to propose a method of extraction of the grammatical features from the text, compile a statistical model based on these grammatical features and finally, make use of these probabilities in refining probabilities of the basic, word-based language model.

The process of extraction of the grammatical information from the text means assigning one of the list possible features for each word in the sentence of the training corpus, forming up several word classes, where one word class consists of each word in the vocabulary of the speech recognition system that can have the same grammatical feature assigned. Statistics

INTECH
open science | open minds

©2012 Staš et al., licensee InTech. This is an open access chapter distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/3.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

collected from these word classes then represent a general grammatical features of the training text, that can be then used to improve original word-based probabilities.

1.1. Data sparsity in highly inflectional languages

Language modeling is an open problem for a long time and still cannot be considered as solved. Most of the research has been performed in the domain of English - as a consequence, most of the proposed methods work well with languages similar to English. Processing languages different from English, such as Slavic languages still have to deal with specific problems.

As it is stated in [25], common aspects of highly inflectional languages with non mandatory word order is inflective nature of Slavic languages, where: "...majority of lexical items modify its basic form according to grammatical, morphological and contextual relations. Nouns, pronouns, adjectives and numerals change their orthographic and phonetic forms with respect to grammatical case, number and gender."

This property, together with rich morphology brings extremely large lexicons. Slavic languages are also characterized by free word order in the sentence. The same meaning can be expressed by more possible word orders in the sentence and grammatical correctness still stays valid.

The main problems of forming a language model of Slavic languages can be summarized as:

- vocabulary size is very high, one word has many inflections and forms;
- size of necessary training text is very large – it is hard to catch all events;
- number of necessary n-grams is very large.

Solutions presented in [25] are mostly based on utilization of grammatical features and manipulation of the dictionary:

- dictionary based on most frequent lemmas;
- dictionary based on most frequent word-forms;
- dictionary based on morphemes.

Each of these methods require a special method of preprocessing of the training corpus and producing a language model. Every word in the training corpus is replaced by the corresponding item (lemma, word-form or a sequence of morphemes) and a language model is constructed using the processed corpus. For a highly inflectional language, where thanks to the large dictionary the estimation of the probabilities is very difficult, language modeling using a extraction of the grammatical features of words seems to be a beneficial way how to improve general accuracy of the speech recognition .

2. Language modeling with sparse training corpus

The biggest issue of building a language model is data sparsity. To get a correct maximum likelihood estimate, all possible trigram combinations should be in the training corpus. This is very problematic, if a bigger dictionary is taken into account. Just in the case of usual, 100k word dictionary that will sufficiently cover the most of the commonly used communication,

there are 10^{15} possible combinations. This amount of text that will contain all combinations that are possible with this dictionary is just impossible to gather and process. In the most cases, training corpora are much smaller, and as a consequence, number of extracted *n*-grams is also smaller. Then it is possible that if a trigram does not exist in a training corpus, it will have zero probability, even if the trigram combination is perfectly possible in the target language.

To deal with this problem, process of adjusting calculated probabilities called *smoothing* is necessary. This operation will move part of the probability mass from the *n*-grams that is present in the training corpus to the *n*-grams that are not present in the training corpus and has to be calculated from data that is available.

Usually, in the case of missing *n*-gram in the language model, required probability is calculated by using available *n*-grams of lower order using *back-off scheme* [19]. For example, if the trigram is not available, bigram probabilities are used to estimate probability of the trigram. Using the same principle, if the bigram probability is not present, unigram probabilities are used for calculation of the bigram probability. This principle for bigram language model is depicted in the Fig. 1.

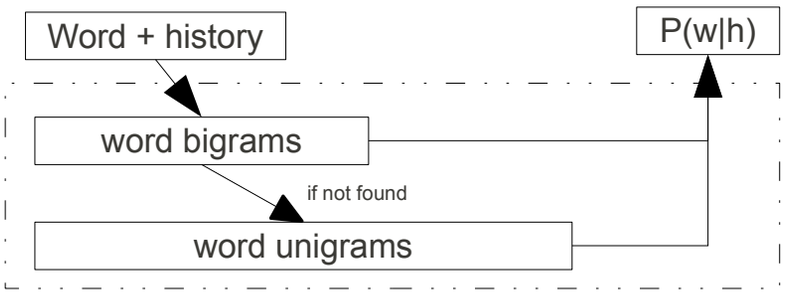


Figure 1. Backoff scheme in a bigram language model

Often, the back-off scheme is not enough by itself for efficient smoothing of the language model, and *n*-gram probabilities have to be adjusted even more. Common additional techniques are methods based on adjusting of *n*-gram counts, such as *Laplace smoothing*, *Good-Touring method*, *Witten-Bell* [5] or *modified Knesser-Ney* [21] algorithms. The problem of this approach is that these methods are designed for languages that are not very morphologically rich. As it is showed in [17, 24], is that this kind of smoothing does not bring expected positive effect for highly inflectional languages with large vocabulary.

Another common approach for estimating a language model from sparse data is *linear interpolation*, also called *Jelinek-Mercer smoothing* [16]. This method allows a combination of multiple independent sources of knowledge into one, that is then used for compose the final language model. In the case of trigram language model, this approach can calculate the final probability as a linear combination of unigram, bigram and trigram *maximum likelihood estimates*. Linear interpolation is not the only method of combining of multiple knowledge sources, other possible approaches are *maximum entropy* [1], *log-linear interpolation* [20] or *generalized linear interpolation* [15].

For a bigram model, a linear interpolation scheme of utilizing bigrams and unigrams is depicted in the Fig. 2. In this case, the final probability is calculated as a linear combination of

both sources according to the equation:

$$P = \lambda P_1 + (1 - \lambda) P_2. \tag{1}$$

Interpolation parameter λ can be set empirically, or can be calculated by one of the optimization methods, e.g. by using *expectation-maximization* algorithm. The coefficient λ have to be chosen, such that the final language model composed from the training corpus fits best the target domain, represented by the testing corpus.

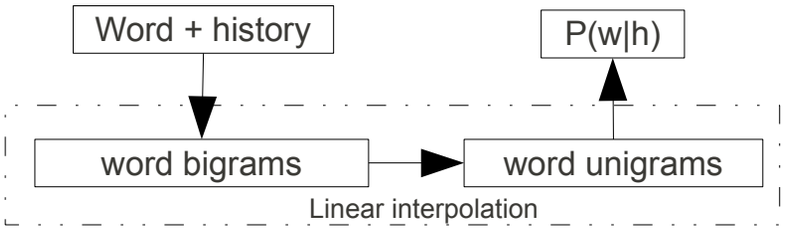


Figure 2. Bigram model with linear combination

3. Class based language models

The presented basic language modeling methods are usually not sufficient for successful real-world automatic speech recognition system. To overcome a data-sparsity problem, *class-based language models* were proposed in [4]. This approach offers ability to group words into classes and work with a class as it was a single word in the language model. This feature means that the class-based language model can considerably reduce sparsity of the training data. Also, an advantage is that the class-based models take into the account dependencies of words, not included in the training corpus.

Probability of a word, conditioned on its history $P(w_i|w_{i-1} \dots w_{i-n+1})$ in the class-based language model can be described using equation [4]:

$$P(w_i|w_{i-1} \dots w_{i-n+1}) = P(c_i|c_{i-1} \dots c_{i-n+1}) P(w_i|c_i), \tag{2}$$

where $P(c_i|c_{i-1} \dots c_{i-n+1})$ is probability of a class c_i, where word w_i belongs, based on the class history. In this equation, probability of a word w according to its history of $n-1$ words $h = \{w_{i-1} \dots w_{i-n+1}\}$ is calculated as a product of class-history probability $P(c_i|c_{i-1})$ and word-class probability $P(w_i|c_i)$.

3.1. Estimation of the class-based language models

As it is described in [18], if using *maximum likelihood estimation*, n-gram probability can be calculated in the same way as in the word-based language models:

$$P(c_i|c_{i-1} \dots c_{i-N+1}) = \frac{C(c_{i-n+1} \dots c_i)}{C(c_{i-n+1} \dots c_{i-1})}, \tag{3}$$

where $C(c_{i-N+1} \dots c_i)$ is a count of sequence of classes in the training corpus and $C(c_{i-n+1} \dots c_{i-1})$ is count of the history of the class c_i in the training corpus.

The word-class probability can be estimated as a fraction of a word count $C(w)$ and class total count $C(c)$:

$$P(w|c) = \frac{C(w)}{C(c)}. \tag{4}$$

Basic feature of the class-based models is lowering number of independent parameters [4] of the resulting language model. For word-based n-gram language model, there is a probability value for each n-gram, as well as back-off weight for lower order n-grams. For class-based model, a whole set of words is reduced to a single class and class-based model describes statistical properties of that class. Another advantage is that the same classical smoothing methods that were presented above can be used for a class-based language model as well.

3.2. Word clustering function

Classes in the class-based model bring bigger level of generalization, rather than manipulating with words, model deals with whole classes of words. Advantage of this approach is significantly lower number of n-grams, where resulting number of n-grams depends on a number of classes.

Generalization of words to classes using the clustering function can reduce data sparsity problem. Each class substitutes whole group of words in the class-based language model, therefore a much larger number of word sequences that are possible in the language can be covered by the language model - there is a much higher probability that a certain word sequence will have a non-zero probability. From this reason, it is possible to see class-based language model as a certain type of language model smoothing - partitioning of the dictionary.

The basic idea of the class-based language model is to take additional dependencies between words into account by grouping words into classes and finding dependencies between these classes. Each word in the given training corpus can be clustered to the corresponding classes, where each class can contain words with similar semantic or grammatical meaning. Words in the classes then share common statistical properties according to their context. In general, one word can belong to multiple classes and one class can contain more words.

In the context of a class-based language model, a class can be seen as a group of words. Each this kind of group can be defined using a function. This word clustering function g that can map any word w from the dictionary V and its context h to one of the possible classes c from the set of all classes C can be described as:

$$g(w,h) \rightarrow c, \tag{5}$$

where class c is from the set of all possible classes G, word w is from vocabulary V and h is surrounding context of the word w. This function can be defined in multiple ways - utilizing expert knowledge, or using data-driven approaches for word-classes induction and can have various features.

If the word clustering function is generalized to include every possible word, class-based language model equation then can it can be written as:

$$P(w|h) = P_g(g(w,h)|g(h))P(w|g(w,h)). \tag{6}$$

4. A method for utilizing grammatical features in the language modeling

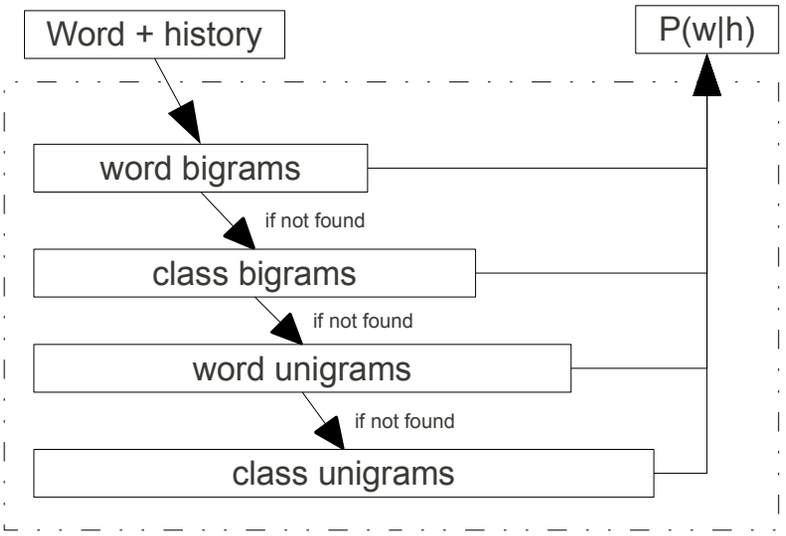


Figure 3. Back-off scheme for the language model

The main problem with class-based language models is how to optimally design a word clustering function g. The word clustering function can be induced in a purely uncontrolled way, based on heuristics "words with the same context belongs to the same class", or some knowledge about the grammatical features of the target language can be used. Inspired by the algorithm proposed in [4] and [32] proposes an automatically induced word classes for building a class-based language model. On the other hand, it seems to be feasible to include some information about the grammar of the language. *Stem-based morphological language model* are proposed in [8, 33].

There is extensive research performed in the task of utilizing grammatical features in a class-based models. The work presented in [23] states that a combination of a plain word-based models with a class-based models can bring improvement in the accuracy of the speech recognition. [22] evaluates a linear combination of the classical word-based language models and grammar-based class models for several languages (English, French, Spanish, Greek).

As it was presented in the previous text, class-based models have some features that are desirable, such as reducing training data sparsity. On the other hand, the main feature of the class-based language model, generalization of the words, can be disadvantage for the most common n-grams. If word-based n-gram exist in the training corpus, it is very possible that it will have more accurate probability than the corresponding class-based n-gram. For less frequent n-grams, class-based language model might be more precise.

It seems that class-based language models using grammatical features might be useful in the automatic speech recognition. Advantage of the class-based model is mainly in the estimation of the probability of events that are rare in the training corpus. On the other hand, events that are relatively frequent, are better estimated by the word-based language model. The ideal solution will be to connect both models, so they can cooperate - frequent cases would be evaluated by the word-based n-grams, less frequent events would be evaluated by the class-based n-grams. Example schematics for this solution would be in the Fig. 3.

From this reason a linear combination of a class-based with word-based language model is proposed. It should be performed in a way that the resulting language model will mostly take into account word-based n-grams if they are available and in the case that here is no word-based n-gram it will fall back to the class-based n-gram that uses grammatical features, as it is showed in the Fig. 3.

4.1. Linear interpolation of the grammatical information

This framework can be implemented using linear interpolation method, where the final probability of the event is calculated as a weighted sum of two components - class-based and word-based model. As it is known that the word based-model almost always give better precision, coefficient of the word-based component should be much higher than the class-based component. Probability of the event will be affected by the class-based component mainly in the case, when the word-based component $P_w(w|h)$ will give zero probability, because it was not seen in the training corpus. On the other hand, the class-based components $P_g(g(w)|g(h))$ might be still able to provide non-zero probability, because rare event can be estimated from words that are similar in the means of the word clustering function g.

Class-based language model utilizing grammatical features, then consists of two basic parts: word-based and class-based language model that was constructed using word clustering function (as it is in the Fig. 2).

Probability of a word, according to its history $P(w|h)$ then can be calculated using equation:

$$P(w|h) = \lambda P_w(w|h) + (1-\lambda)P_g(g(w)|g(h))P(w|g(w)), \tag{7}$$

where P_w is probability returned by the word-based model and P_g is probability returned by the class-based model with the word-clustering function g that utilizes information about grammar of the language.

This kind of language model consists of the following components:

- vocabulary V that contains a list of known word of the language model;
- word-based language model constructed from the training corpus that can return word-history probability $P_w(w|h)$;
- word clustering function $g(w,h)$ that maps words into classes;
- class-based language model $P_g(g(w)|g(h))$ created from a training corpus and processed by the word clustering function $g(w,h)$;
- word-class probability function that assigns a probability of occurrence of a word in the given class $P(w|g(w))$;
- interpolation constant λ from interval $(0,1)$ that expresses weight of the word-based language model.

The first part of this language model can be created using classical language modeling methods from the training corpus. To create a class-based model, the training corpus has to be processed by the word clustering function and every word has to be replaced by its corresponding class. From this processed training corpus, a class-based model can be built. During this process, a word-class probability function has to be estimated. This function expresses probability distribution of words in the class. The last step is to determine the interpolation parameter λ, should be set to values close (but lower) to 1.

4.2. Extracting grammatical features

The hardest part of this process seems to be processing of the training corpus by the grammatical feature extraction function. Description of the sentence by its grammatical features is basically a classification task, where each word and its context has assigned one feature from a list of all possible features. This conforms to the Eq. 5, that puts words into classes.

Common grammatical features (usable for a highly inflective languages with non-mandatory word order in the sentence):

- part-of-speech, a label that expresses grammatical categories of the word in a sentence, such as number, case or grammatical gender;
- lemma, a basic form of the word;
- word suffix, a part of the word that inflects according to the grammatical form of the word;
- word stem, a part of the word that does not inflect and usually carries meaning of the word.

There are more possible methods of segmentation of words into such features. Basically, they can be divided into two groups - *rule-based* and *statistics-based methods*. In the following text, a rule-based method for identifying a stem or a suffix of the word will be presented and a statistic-based method for finding the word lemma or part-of-speech will be introduced.

4.2.1. Suffix and stem identification method

Suffix or stem identification method belongs to the field of the morphological analysis of the language. There is a number of specialized methods and tools, such as *Morfessor* [6] using uncontrolled learning methods. More methods of the morphological analysis are provided in [9]. Disadvantage of the majority of proposed methods is that they are not very suitable for Slavic languages. From this reason, a specialized method is necessary.

Because Slovak language is characterized by a very rich morphology, mainly on the suffix side, a simple method, taking specifics of the language is presented. The method is based on suffix identification, based on a list, obtained by counting suffixes in the list of all words (from [29]) and taking suffixes with high occurence.

First necessary thing is a list of suffixes. This list can be obtained by studying a dictionary of words, or some simple count-based analysis can be used.

1. a dictionary of the most common words in the language has been obtained;
2. from each word longer than 6 characters, a suffix of length 2, 3 or 4 characters has been extracted;
3. number of occurrences of each extracted suffix has been calculated;
4. a threshold has been chosen and suffixes with count higher than the threshold has been added to the list of all suffixes.

If the list of the most common suffixes is created, it is possible to easily identify the stem and suffix and stem of the word.

1. if the word is shorted than 5 characters, suffix cannot be extracted;
2. if word is longer than 5 characters, word ending of length $n = 5$ is examined. If it is in the list of the most common suffixes, it is the result. If the ending of length n is not in the list, algorithm continues with $n - 1$;
3. if no suffix has been identified, word is considered as a class by itself.

Disadvantage of this method is that it is statistically based and it is not always precise and some suffixes found might not be grammatically correct.

This suffix or stem assignment function then can be used as a word clustering function that can assign certainly one class to every word. Words with the same suffix or stem will then belong to the same class and according to the properties of the language, they will share similar statistical features.

4.2.2. Part-of-speech or lemma identification using statistical methods

Suffix extraction task is not too complicated and a simple algorithm can be used to achieve plausible results. It is straight-forward and one word will always the same suffix, because no additional information, such as context is considered. On the other hand, the process of identification of the part-of-speech is much more difficult and context-dependent. In the case of the Slovak language, the same word can have many different tags assigned, depending on the surrounding context of the word.

Part-of-speech (POS) tag, assigned to a word expresses grammatical categories of a word in a sentence. The same word can have multiple POS tags and surrounding context of the word has to be used to specify correct grammatical category. This task is difficult also for trained human annotator and requires sufficient knowledge about the grammar of the language.

Using a set of hand-crafted rules to assign a POS tag to a word did not show up to be useful. In a morphologically rich language, the number of possible POS tags is very high and covering every case by a rule seems to be exceptionally complex. Even if there are some approaches using uncontrolled learning techniques [10], useful in the cases, when no a priori knowledge is available the most commonly used approach is in statistical methods that make use of the hand-annotated corpora.

The most commonly used classification methods are based on a *hidden Markov models*, e.g. *HunPOS* [11], based on *TnT* [2] that is a statistical model is trained on a set of manually annotated data. In some approaches, an expert knowledge can be directly inserted to the statistical system, e.g. *rule-based tagger* [13].

Common statistical approaches include:

- Brill tagger (transformation learning) [3];
- hidden Markov model classifier [11];
- maximum entropy (log-linear regression) classifier [27];
- averaged perceptron methods [30, 34].

Lemma assignment task is very similar to the part-of-speech assignment task, and very similar methods can be used. The part-of-speech or lemma assignment function can be used as a word clustering function, when forming a class-based language model. The problem with this approach is that it is possible that one word can belong to more classes at once that can bring a lower precision of the language model.

4.2.3. Hidden Markov model based on word clustering

Hidden Markov models is a commonly used method for a sequential classification of the text. This kind of classifier can be used for various tasks, where a disambiguation is necessary, such as part-of-speech tagging, lemmatization or named entity recognition. Also, it is essential for other tasks in the automatic speech recognition, such as acoustic modeling. The reason for a high popularity of this method is very good performance, both in precision and speed and well-described mathematical background.

The problem of assigning the best sequence of tags or classes $g_{best}(W) = \{c_1, \dots c_n\}$ to a sequence of words $W = \{w_1, w_2 ... w_n\}$ can be described by the equation:

$$g_{best}(W) = arg \max_i P(g_i(W)|W), \tag{8}$$

where the best sequence of classes $g_{best}(W)$ is assigned from all class sequences that are possible for the word sequence W, according to the probability of occurrence of the class sequence in the case of the given word sequence W.

There are several problems with this equation. First, the number of possible sequences $g_i(W)$ is very high and it is not computationally feasible to verify them individually. Second, there has to be a framework for expressing probability of the sequence $P(g_i(W)|W)$ and calculating its maximum.

The hidden Markov model is defined as a quintuple:

- G_0 - a priori probability distribution of all classes;
- G - set of possible states (classes);
- W - set of possible observations (words);
- A - state transition matrix, that expresses probability $P(c_i|c_{i-1})$ of occurrence of the class c_i, if class c_{i-i} preceded in the sequence;
- B - observation probability matrix, that gives probability $P(w_i|c_i)$ of word w_i for the class c_i.

For construction of the hidden Markov model for the task of POS tagging, all of these components should be calculated, as precisely as it is possible. The most important part of the whole process is manually prepared training corpus, where each word has a class assigned by hand. This process is very difficult and requires a lot of work of human annotators.

When annotated corpus is available, estimation of the main components of the hidden Markov model, matrices A and B is relatively easy. Again, the maximum likelihood method can be

used, together with some smoothing techniques:

$$A = P(c_i|c_{i-1}) = \frac{C(c_{i-1}, c_i)}{C(c_{i-1})} \tag{9}$$

and

$$B = P(w_i|c_i) = \frac{C(w_i, c_i)}{C(c_i)}, \tag{10}$$

where $C(c_{i-1}, c_i)$ is count of the pair of succeeding classes c_{i-1}, c_i, $C(c_i)$ is count of the class c_i in the training corpus. After matrices A and B are prepared, the best sequence of classes for the given sequence of words can be calculated using a method of dynamic programming - the Viterbi algorithm.

As it was stated above, the Slovak language is characterized by its rich morphology and large vocabulary and this fact makes the task of the POS tagging more difficult. During experiments, it has shown up that these basic methods are not sufficient, and additional modification of the matrices A and B is required.

For this purpose, a *suffix-based smoothing method* has been designed, similar but not the same as in [2]. Here, an accuracy improvement can be achieved by calculation of the suffix-based probability $P(g_{suff}(w_i)|c_i)$. This probability estimate uses the same word-clustering function for assigning words into classed as is presented above. Again, the observation probability matrix is adjusted using a linear combination:

$$B = \lambda P(w_i|c_i) + (1 - \lambda)P(g_{suff}(w_i)|c_i). \tag{11}$$

This operation helps to better estimate probability of the word w_i in for the class c_i, even if a pair w_i, c_i does not exist in the training corpus. The second component of the expression improves the probability estimate with counts of words, that are similar to the word w_i.

5. Experimental evaluation

Basically, language models can be evaluated in two possible ways. In the extrinsic evaluation is language model tested in simulated real-life environment and performance of the whole automatic speech recognition system is observed. The result of the recognition is compared to the annotation of the testing set. Standard measure for extrinsic evaluation is *word error rate* (*WER*) that is calculated as:

$$WER(W) = \frac{C_{INS} + C_{DEL} + C_{SUB}}{C(W)}, \tag{12}$$

where C_{INS} is number of false inserted words, C_{DEL} is number of unrecognized words and C_{SUB} is number of words that were confused (substituted), when the result of the recognition is compared to the word sequence W.

WER is evaluation of the real output of the whole automatic speech recognition system. It evaluates user experience and is affected by all components of the speech recognition system.

On the other hand, intrinsic evaluation is the one "that measures the quality of the model, independent on any application" [18]. For n-gram language models, the most common

evaluation metric is *perplexity* (*PPL*). "The perplexity can be also viewed as a weighted averaged branching factor of the language model. The branching factor for the language is the number of possible next words that can follow any word" [18]. Similarly to the extrinsic method of evaluation, a testing corpus is required. The resulting perplexity value is always connected with the training corpus. According to the previous definition, perplexity can be expressed by the equation:

$$PPL(W) = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w|h)}}, \tag{13}$$

where $P(w|h)$ is a probability, returned by the tested language model and expresses probability of all words conditioned by its histories from the testing corpus of the length N.

Compared to the extrinsic methods of evaluation, it offers several advantages. Usually, evaluation using perplexity is much faster and simpler, because only testing corpus and language model evaluation tool is necessary. Also, this method eliminates unwanted effects of other components of the automatic speech recognition system, such as acoustic model of phonetic transcription system.

5.1. Training corpus

The most important thing that is necessary for building good language model is correctly prepared training data in a sufficient amount. For this purpose, a text database [12] has been used. As it was mentioned in the introduction section, the training corpus should be large enough to cover the majority of the most common *n*-grams. Also, training data must be as similar as possible to the target domain.

The basic corpus of the adjudgements from the Slovak ministry of Justice has been prepared. The problem is that this corpus is not large enough and have to be complemented with texts from other domains. To enlarge this corpus with more general data, web-based newspaper oriented corpus of the text data from major Slovak newspaper web-sites has been collected. For the vocabulary, 381 313 the most common words has been selected. Contents of the training corpus is summarized in the Table 1.

Corpus	Words	Sentences	Size
Judicature	148 228 795	7 580 892	1.10 GB
Web	410 479 727	19 493 740	2.86 GB
Total	570 110 732	27 074 640	3.96 GB

Table 1. Training corpus

For training a class-based model utilizing grammatical features, further processing of the training corpus is required.

One of the goals of this study is to evaluate usefulness of the grammatical features for the language modeling. The tests are focused on following grammatical features that were mentioned in the previous text:

- part-of-speech;
- word lemma;
- word suffix;
- word stem.

For this purpose, a set of tools, implementing a word clustering function has been prepared. For the POS and lemma, a statistical classifier based on the hidden Markov model has been designed. This classifier has been trained on a data from the [29] (presented in [14]). Method of the statistical classifier is similar to the [2], but uses additional back-off method, based on a suffix extraction described in the previous section.

For identification of the suffix or the stem of the word, just simple suffix subtraction method presented above has been used. When compared to the statistical classifier, this method is much simpler and faster. Also, this kind of word clustering function is more uniform, because one word can belong to just one class. Disadvantage of this approach is that it does not allow to identify a suffix or stem to the words that are short. In this case, for those words that cannot be split, word is considered as a class by itself.

Word clustering by the suffix identification is performed in two versions. Version 1 is using 625 suffixes compiled by hand. Version 2 is using 7 578 statistically identified suffixes (as it was described above). For each grammatical feature, the whole training corpus has been processed and every word in the corpus had a class assigned, according to the used word clustering function.

To summarize, 7 training corpora has been created, one for each grammatical feature examined. First training corpus was the baseline, and other 6 corpora were created by processing of the baseline corpus:

- part-of-speech corpus, marked as POS1 has been created using our POS tagger, based on hiden Markov models;
- lemma-based corpus, marked as LEM1, has been created using our lemma tagger, based on hidden Markov models;
- suffix-based corpus 1, marked as SUFF1, has been created using suffix extraction method with 625 hand compiled suffixes;
- stem-based corpus 1, marked as STEM1, has been created using suffix extraction method with the same suffixes;
- suffix-based corpus 2, marked an SUFF2, has been created using suffix extraction method using statistically obtained 7 578 suffixes;
- stem-based corpus 2, marked as STEM2, obtained by the same method.

5.2. Basic language model preparation

These seven corpora then were able to enter the process of language model creation. For every prepared corpus, a language model has been built.

First necessary step is creation of the dictionary. For the baseline corpus, a dictionary of 381 313 the most frequented words has been selected.

For the class-based models, according to the Eq. 2, besides class-based language model probability $P(c|h_c)$, also word-class probability $P(w|c)$ is required. Again, using maximum likelihood, this probability has been calculated as:

$$P(w|c) = \frac{C(w, g(w))}{C(g(w))}, \tag{14}$$

where $C(w, gw)$ is number of occurrences of word W with class $c = g(w)$ and $C(g(w))$ is number of words in class $g(w)$. The processed corpora will be used for creation of the class-based language model.

Taking prepared training corpus, SRILM Toolkit [31] has been used to build trigram model with baseline smoothing method.

5.3. Basic language model evaluation

	SUFF1	SUFF2	POS1	LEM1	STEM1	STEM2	Baseline
Classes (unigram count)	924	37 704	1 255	122 141	115 318	81 780	329 690
Size (MB)	72	450	489	555	595	522	854
PPL basic	266.41	61.64	355.11	75.3	80.42	89.31	39.76
PPL ($\lambda = 0.98$)	37.32	29.56	38.87	38.87	35.74	34.73	n/a

Table 2. Evaluation of the language model perplexity

Result of this step is 7 language models, one classical word-based models and 6 class-based models, one for each word clustering function.

For quick evaluation, a perplexity measure has been chosen. As an evaluation corpus, 500 000 sentences of held-out data from the court of law adjudgements has been used. Results of the perplexity evaluation and characterization of the resulting language models are in the Table 2.

The results have shown that despite expectations. Perplexity of the class-based models constructed from the processed training corpora is always higher than the perplexity of the word-based models. Higher perplexity means, that the language model does not fit testing data so good. Word-based language model seems to be always better than the class-based model, even if there are some advantages of the class-based language model. But, class-based language models could be useful. Thanks to the word clustering function, they still provide extra information that is not included in the baseline model. The hypothesis say, that in some special cases, the class-based language model can give better result than the word-based model. The way, how this extra information can be utilized is linear interpolation with the baseline model, so it contains both word-based and class-based n-grams.

5.4. Creating class-based models utilizing grammatical features

A new set of the interpolated language models have been compiled using methodology described in the previous section. Each class-based model has been taken and together with the baseline word-based model, they were composed together using *linear interpolation*.

Weight of the word based model has been set to $\lambda = 0.98$ and also word class probability calculated in the previous step has been used.

The result of this process is again a class-based model. This new class-based model utilizing grammatical features contains two types of classes. Word-based class, where class contains only one member, the word. The second type of class is grammar-based class, where class contains all words, that are mapped by the word clustering function.

This new set of the interpolated language models have been evaluated for perplexity. Results are in the Table 2. It is visible that after the interpolation, perplexity of the interpolated models has decreased very much. This fact confirms the hypothesis about usability of the grammar-based language models.

5.5. Automatic speech recognition with class-based models utilizing grammatical features

The main reason for improvement of the language model is the improvement of the automatic speech recognition system. The correct evaluation of the language models in the task of the automatic speech recognition would not be complete without extrinsic test in the simulated real-world tasks of the recognition of the pre-recorded notes for adjudgements. Therefore, the main tool for evaluation is an automatic speech recognition system, originally designed for the judicature [28] with acoustic model [7].

For this purpose, two testing sets named APD1 and APD2 were used. Both test sets are focused on the task of transcription of the dictation for use at the Ministry of Justice. Each test set contain over 3 000 sentences that were recorded and annotated. After recognition of the recording, WER has been calculated by comparing the annotation with the result of the recognition using target language model.

To evaluate robustness of the language model, out-of-domain test has also been constructed. Purpose of this test is to find out, how the system will perform in a conditions that are different than the planned. For this test, a set of recordings from broadcast-news database [26] has been used.

Each test is summarized in the Table 3, where name, number of words, number of sentences and size can be found. Results of extrinsic tests can be shown in the Table 4.

Test Set	Sentences	Words
APD1	3 010	41 111
APD2	3 493	41 725
BN	4 361	40 823
Eval Corpus	500 000	15 777 035

Table 3. Test set

Test set	SUFF1	SUFF2	POS1	LEM1	STEM1	STEM2	Baseline
APD1	12.53	12.09	12.38	12.40	12.44	12.50	12.28
APD2	11.37	11.14	11.23	11.25	11.47	11.36	11.32
BN	21.91	21.40	21.87	21.84	21.70	21.63	21.23

Table 4. Language model WER [%] evaluation

5.6. Results of experiments and discussion

To summarize, the whole process of creating and evaluating a class-based language model that utilizes grammatical information can be described as:

1. train set and test set has been prepared;
2. baseline dictionary has been selected;
3. baseline language model has been prepared;
4. for each method of feature extraction, a class-based training corpus has been set-up. Each word in the train set and test set had a grammatical class assigned;
5. from each class-based training corpus, a class-expansion dictionary has been calculated. The dictionary contains information as a triplet $(c, P(w|c), w)$;
6. for each class-based training corpus, a class-based model has been prepared;
7. perplexity of the obtained class-based model has been evaluated;
8. for each class-based model, a linear interpolation with has been performed;
9. for every resulting class-based interpolated model perplexity has been calculated.

First conclusion from these experiments (see Table 2) is that the classic word based language models generally give better precision than the class-based grammar models. Their main advantage is the smoothing ability - estimating probability of the less frequent events using words that are grammatically similar. This advantage can be utilized using linear interpolation, where final probability is calculated as a weighted sum of the word-based component and class-based component. That will help in better distribution of the probability mass in the language model - thanks to the grammar component, more probability will be assigned to the events (word sequences) not seen in the training corpus. Effect of the grammar component is visible in the Table 2, where using simple suffix extraction method and linear interpolation helped to decrease perplexity of baseline language model by 25%.

Effect of the decreased perplexity has been evaluated in extrinsic tests - recognition of dictation of the legal texts. From this these tests, summarized in the Table 4 can be seen, how decreased perplexity affects final precision of the recognition process. In the case of the suffix extraction method, 2% relative WER reduction has been achieved. Interesting fact is that change of the perplexity not always led to decreasing of the WER. From this fact it is possible to say that perplexity of the language model is not always expressing quality of the language model in the task of the automatic speech recognition, where final performance is affected by more factors, and can be used just as a kind of clue in next necessary steps.

Next conclusion is that not every grammatical feature can be useful for increasing precision of the speech recognition. Each test shows notable differences in the perplexities and word error rates for each created language model. After a closer look at the results, it can be seen that those features that are based more on the morphology of the word, such as suffix or part-of-speech perform better than those that are more based on the semantics of the word, such as stem or lemma-based features (compare to [23]). Also, when comparing suffix extraction method 1 and 2, we can see that statistically obtained high number of classes yield better results, than the handcrafted list of suffixes.

6. Conclusion

This presented approach has shown that using suffix-based extraction method, together with interpolated class-based model can bring much smaller perplexity of the language model and considerably lower *WER* in the automatic speech recognition system. Even if a class-based models do not bring important improvement of the recognition accuracy, they can be used as a back-off schema in the connection with the classical word-based language models, using linear interpolation.

Class-based language models with utilization of the grammatical features allow:

- optimize search network of the speech recognition system by putting some words into classes;
- have ability to incorporate new words into the speech recognition system without the need of re-training the language model;
- better estimate probabilities of those *n*-grams that did not occur in the training corpus.

Disadvantages of the class-based language models:

- relatively larger search network (it includes both words and word-classes);
- more difficult process of the training language models.

The future work in this field should be focused on even better usability of this type of language model. First area that have not been mentioned in this work is the size of the language model. Size of the language model influences loading times, recognition speed and used disk space of the real-word speech recognition system. Effectively pruned language model should also bring better precision, because it removes *n*-grams that can be calculated from lower-order *n*-grams.

The second area that deserves more attention is the problem of language model adaptation. Thanks to the class-based nature of this type of language model, new words and new phrases can be inserted into the dictionary by the user and this feature should be inspected precisely.

This word has introduced a methodology for building a language model for highly inflective language such as Slovak. It can be also usable for similar languages with rich morphology, Polish or Czech. It brings a better precision and ability to include new words into the language model by the user without the need of re-training of the language model.

Acknowledgement

The research presented in this paper was supported by the Ministry of Education under the research project MŠ SR 3928/2010-11 (50%) and Research and Development Operational Program funded by the ERDF under the project ITMS-26220220141 (50%).

Author details

Ján Staš, Daniel Hládek and Jozef Juhár
Department of Electronics and Multimedia Communications
Technical University of Košice, Slovakia

7. References

[1] Berger, A., Pietra, V. & Pietra, S. [1996]. A maximum entropy approach to natural language processing, *Computational Linguistics* **22**(1): 71.

[2] Brants, T. [2000]. TnT: A statistical part-of-speech tagger, *Proc. of the 6th Conference on Applied Natural Language Processing*, ANLC'00, Stroudsburg, PA, USA, pp. 224–231.

[3] Brill, E. [1995]. Transformation-based error-driven learning and natural language processing: A case study in part-of-speech tagging, *Computational Linguistics* **21**: 543–565.

[4] Brown, P., Pietra, V., deSouza, P., Lai, J. & Mercer, R. [1992]. Class-based n-gram models of natural language, *Computational Linguistics* **18**(4): 467–479.

[5] Chen, S. F. & Goodman, J. [1999]. An empirical study of smoothing techniques for language modeling, *Computer Speech & Language* **13**(4): 359–393.

[6] Creutz, M. & Lagus, K. [2007]. Unsupervised models for morpheme segmentation and morphology learning, *ACM Transactions on Speech and Language Processing* **4**(1).

[7] Darjaa, S., Cerňak, M., Beňuš, v., Rusko, M.and Sabo, R. & Trnka, M. [2011]. Rule-based triphone mapping for acoustic modeling in automatic speech recognition, *Springer-Verlag, LNAI 6836* pp. 268–275.

[8] Ghaoui, A., Yvon, F., Mokbel, C. & Chollet, G. [2005]. On the use of morphological constraints in n-gram statistical language model, *Proc. of the 9th European Conference on Speech Communication and Technology*.

[9] Goldsmith, J. [2001]. Unsupervised learning of the morphology of a natural language, *Computational Linguistics* **27**(2): 153–198.

[10] Graça, J. V., Ganchev, K., Coheur, L., Pereira, F. & Taskar, B. [2011]. Controlling complexity in part-of-speech induction, *Journal of Artificial Intelligence Research* **41**(1): 527–551.

[11] Halácsy, P., Kornai, A. & Oravecz, C. [2007]. HunPos - An open source trigram tagger, *Proc. of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, Stroudsburg, PA, USA, pp. 209–212.

[12] Hládek, D. & Staš, J. [2010]. Text mining and processing for corpora creation in Slovak language, *Journal of Computer Science and Control Systems* **3**(1): 65–68.

[13] Hládek, D., Staš, J. & Juhár, J. [2011]. A morphological tagger based on a learning classifier system, *Journal of Electrical and Electronics Engineering* **4**(1): 65–70.

[14] Horák, A., Gianitsová, L., Šimková, M., Šmotlák, M. & Garabík, R. [2004]. Slovak national corpus, *P. Sojka et al. (Eds.): Text, Speech and Dialogue, TSD'04*, pp. 115–162.

[15] Hsu, B. J. [2007]. Generalized linear interpolation of language models, *IEEE Workshop on Automatic Speech Recognition Understanding, ASRU'2007*, pp. 136–140.

[16] Jelinek, F. & Mercer, M. [1980]. Interpolated estimation of Markov source parameters from sparse data, *Pattern recognition in practice* pp. 381–397.

[17] Juhár, J., Staš, J. & Hládek, D. [2012]. Recent progress in development of language model for Slovak large vocabulary continuous speech recognition, *Volosencu, C. (Ed.): New Technologies - Trends, Innovations and Research* . (to be published).

[18] Jurafsky, D. & Martin, J. H. [2009]. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition (2nd Edition)*, Prentice Hall, Pearson Education, New Jersey.

[19] Katz, S. [1987]. Estimation of probabilities from sparse data for the language model component of a speech recognizer, *IEEE Transactions on Acoustics, Speech and Signal Processing* **35**(3): 400–401.

[20] Klakow, D. [1998]. Log-linear interpolation of language models, *Proc. of the 5th International Conference on Spoken Language Processing*.

[21] Kneser, R. & Ney, H. [1995]. Improved backing-off for m-gram language modeling, *Proc. of ICASSP* pp. 181–184.

[22] Maltese, G., Bravetti, P., Crépy, H., Grainger, B. J., Herzog, M. & Palou, F. [2001]. Combining word-and class-based language models: A comparative study in several languages using automatic and manual word-clustering techniques, *Proc. of EUROSPEECH*, pp. 21–24.

[23] Nouza, J. & Drabkova, J. [2002]. Combining lexical and morphological knowledge in language model for inflectional (czech) language, pp. 705–708.

[24] Nouza, J. & Nouza, T. [2004]. A voice dictation system for a million-word Czech vocabulary, *Proc. of ICCCT* pp. 149–152.

[25] Nouza, J., Zdansky, J., Cerva, P. & Silovsky, J. [2010]. Challenges in speech processing of Slavic languages (Case studies in speech recognition of Czech and Slovak), *in* A. E. et al. (ed.), *Development of Multimodal Interfaces: Active Listening and Synchrony*, LNCS 5967, Springer Verlag, Heidelberg, pp. 225–241.

[26] Pleva, M., Juhár, J. & Čižmár, A. [2007]. Slovak broadcast news speech corpus for automatic speech recognition, *Proc. of the 8th Intl. Conf. on Research in Telecomunication Technology, RTT'07*, Liptovský Ján, Slovak Republic, p. 4.

[27] Ratnaparkhi, A. [1996]. A maximum entropy model for part-of-speech tagging, *Proc. of Empirical Methods in Natural Language Processing*, Philadelphia, USA, pp. 133–142.

[28] Rusko, M., Juhár, J., Trnka, M., Staš, J., Darjaa, S., Hládek, D., Cerňák, M., Papco, M., Sabo, R., Pleva, M., Ritomský, M. & Lojka, M. [2011]. Slovak automatic transcription and dictation system for the judicial domain, *Human Language Technologies as a Challenge for Computer Science and Linguistics: 5th Language & Technology Conference* pp. 365–369.

[29] SNK [2007]. Slovak national corpus.
URL: *http://korpus.juls.savba.sk/*

[30] Spoustová, D., Hajič, J., Votrubec, J., Krbec, P. & Květoň, P. [2007]. The best of two worlds: Cooperation of statistical and rule-based taggers for Czech, *Proc. of the Workshop on Balto-Slavonic Natural Language Processing: Information Extraction and Enabling Technologies*, pp. 67–74.

[31] Stolcke, A. [2002]. SRILM – an extensible language modeling toolkit, *Proc. of ICSLP*, Denver, Colorado, pp. 901–904.
[32] Su, Y. [2011]. Bayesian class-based language models, *Proc. of ICASSP*, pp. 5564–5567.
[33] Vergyri, D., Kirchhoff, K., Duh, K. & Stolcke, A. [2004]. Morphology-based language modeling for arabic speech recognition, *Proc. of ICSLP*, pp. 2245–2248.
[34] Votrubec, J. [2006]. Morphological tagging based on averaged perceptron, *Proc. of Contributed Papers, WDS'06*, Prague, Czech Republic, pp. 191–195.

IntechOpen

IntechOpen