Knowledge and Rule-Based Diacritic Restoration in Serbian

Cvetana Krstev
University of Belgrade
Faculty of Philology
cvetana@matf.bg.ac.rs

Ranka Stanković
University of Belgrade
Faculty of Mining and Geology
ranka@rgf.rs

Duško Vitas
University of Belgrade
Faculty of Mathematics
vitas@matf.bg.ac.rs

Abstract

In this paper we present a procedure for the restoration of diacritics in Serbian texts written using the degraded Latin alphabet. The procedure relies on the comprehensive lexical resources for Serbian: the morphological electronic dictionaries, the Corpus of Contemporary Serbian and local grammars. Dictionaries are used to identify possible candidates for the restoration, while the data obtained from SrpKor and local grammars assists in making a decision between several candidates in cases of ambiguity. The evaluation results reveal that, depending on the text, accuracy ranges from 95.03% to 99.36%, while the precision (average 98.93%) is always higher than the recall (average 94.94%).

1. Motivation

In Serbia, the use of Cyrillic alphabet is prescribed by law (Zakon, 2010: article 1), while the use of Latin alphabet is permitted in special situations (traffic signs, street names, etc.). However, due to historical and other reasons Latin alphabet is widely used.[1] One of the reasons is that the Cyrillic alphabet had poor support in the digital world before Unicode was fully implemented. But the Serbian version of the Latin alphabet does not coincide fully with the English version; some letters are not used – *q*, *w*, *x* and *y* – while other letters, the ones with diacritics – *š*, *đ*, *č*, *ć* and *ž* – are not represented in the ISO 8859-1 Latin 1 encoding scheme, the most used 8-bit superset of ASCII. These circumstances hindered the use of the Serbian variant of the Latin alphabet in some applications in the past and forced a search for other solutions. One of these solutions was to drop diacritics (use *s* instead of *š*, *z* instead of *ž*, and *c* instead of both *č* and *ć*) and replace *đ* with the digraph *dj*, which squeezed the Serbian Latin alphabet to ASCII.[2]

The full implementation of Unicode rendered these solutions unnecessary. And once the degraded use of the Latin alphabet seemed to belong to the past, the new social media applications, especially those relying on short messages, such as Twitter and SMS, revived it. These messages are usually not only short but also written very quickly and it is easier to use the basic Latin alphabet.[3] Besides that, texts without diacritic marks, completely or partially, can emerge as a result of an inadequate transformation from PDF into raw text, as well as a result of a poor OCR.

In the past, some interesting Serbian texts were prepared using the degraded Latin alphabet, that could be used as a corpus material.[4] Many short messages are produced every day and they are a source

[1] Note that the Law on the usage of Language and Script is presented on the web site that collects all Serbian laws and regulations in Latin script `http://www.paragraf.rs/propisi/zakon_o_sluzbenoj_upotrebi_jezika_i_pisama.html`

[2] Such use of the Latin alphabet is in the computer jargon sometimes called *ošišana latinica* 'shaved Latin' or *ćelava latinica* 'bold Latin'. One should note that, in the past, applications were not as user-friendly as they are today. Consequently, even when support for the Serbian Latin alphabet existed many users did not know how to use it.

[3] Note also that SMS messages cost more when characters beyond ASCII are used, since less characters can then be used in one message.

[4] One example: Web site `http://www.yurope.com/people/nena/Zabeleske/` contains a literary works from a number of Serbian writers, all written in the degraded Latin script.

Keywords: diacritic restoration, morphological dictionary, corpus, word n-grams, local grammars

for various research. However, all this cannot be done if the degraded Latin alphabet were not restored to the regular Serbian Latin alphabet.

This paper is organized as follows: in Section 2. we discuss some related work on the same and similar problems, in Section 3. we detail the lexical resources which serve as a base for our solution presented in Section 4. In Section 5., the results of the evaluation are discussed. Finally, in Section 6. we conclude with some remarks and hints for the use of similar procedures for solving similar problems.

2. Related Work

The problem of the degraded alphabet occurs in many languages and in various forms. One of its forms is the omission of the diacritics. Therefore, the solutions to this problem are named "diacritic restoration" or "diacritization". A lot of work has been dedicated to solving this problem for many languages that use the Latin alphabet, including Croatian (Šantić et al., 2009), French (Yarowsky, 1999), Hungarian (Novák and Siklósi, 2015; Acs and Halmi, 2016), Lithuanian (Kapočiūtė-Dzikienė et al., 2017), Romanian (Tufiş and Ceauşu, 2008; Iftene and Trandabat, 2009; Petrică et al., 2014), Slovak (Hládek et al., 2013), Spanish (Atserias et al., 2012; Francom and Hulden, 2013), Turkish (Adali and Eryiğit, 2014) and Vietnamese (Pham et al., 2017). To the best of our knowledge no work was reported for Serbian, besides a solution aiming at several South Slavic languages (Ljubešić et al., 2016).[5] Besides for the Latin script, a similar problem arises for the Arabic script (Alghamdi et al., 2010; Belinkov and Glass, 2015), and in that case it is sometimes named "the vowel restoration" since the diacritics are mainly used to represent the vowels and gemination. Finally, although many of the cited works claim that their systems are language independent, several authors presented specifically language-independent solutions or solutions that can be applied to several related languages. For instance, the solution reported in (Iftene and Trandabat, 2009) is aiming at the law-resourced languages, while those described in (Haertel et al., 2010) and (Ljubešić et al., 2016) were designed for the Semiotic languages and the South Slavic languages, respectively.

The incentive to develop a system for the diacritic (vowel) restoration is obvious – there is a need to obtain a correct text written according to the norms of a certain language. However, as we mentioned in Section 1., recently this problem has received the some attention due to the large masses of texts produced for many languages in the form of short messages using the "ASCII" Latin script (Acs and Halmi, 2016; Adali and Eryiğit, 2014; Ljubešić et al., 2016) that need further processing, for instance by a text-to-speech system (Petrică et al., 2014; Ungurean et al., 2008). Actually, our work on the diacritization in Serbian was spurred by a need to correct and normalize Twitter messages (Mladenović et al., 2017).

Concerning methods used, few systems are primarily rule-based (like (El-Sadany and Hashish, 1988) for Arabic verbs) or knowledge-based (for instance, (Tufiş and Ceauşu, 2008) for Romanian). The main drawback to these approaches is that they rely on lexicons and other NLP resources (e.g. POS taggers) that may not be available and/or would not cover the non-standard word forms (that are usually found in social media messages) (Ljubešić et al., 2016).

The diacritization problem is seen by some authors as a spelling-check problem (Atserias et al., 2012), a disambiguation problem (Yarowsky, 1999), a classification problem (Acs and Halmi, 2016; Adali and Eryiğit, 2014), or a machine-translation problem (Novák and Siklósi, 2015; Ljubešić et al., 2016; Pham et al., 2017). These problems are solved through various statistical approaches which can be grouped into two main categories: character-based and word-based. The attractiveness of the character-based approaches lies in the fact that they are language independent and do not need extensive resources (Alghamdi et al., 2010; Kapočiūtė-Dzikienė et al., 2017). The word-based methods are usually language-dependent as they rely on at least some language resources, while using some kind of a language model: *n*-grams (of words) (Atserias et al., 2012), Hidden-Markov-Models (HMM) (Gal, 2002; Ibraheem, 2017) or neural networks (Belinkov and Glass, 2015; Pham et al., 2017).

In this paper we discuss a rule-based and a knowledge based approach for restoring diacritics in a Serbian text written in the Latin script. An advantage of the rule/knowledge-based systems is that their

[5]There is a web page offering the solution for this problem `http://www.slovomajstor.com/`; however, the author(s) and the methods are not disclosed.

work is transparent and their results can more easily be explained and corrected, should that be necessary. Moreover, as we had a rich lexicon and other NLP tools at our disposal, the main reason for not using a rule/knowledge-based system was no longer relevant. Besides that, our work is inspired by the idea to develop a system that could be adapted to solve several related problems (as will be mentioned in Section 6.).

3. Textual Resources

The input of our procedure for restoring the diacritics is a Serbian text that does not use the diacritics. Once this text is tokenized, it consists of two types of simple words: (a) Words that will not be affected by our procedure, because they contain neither letters *c*, *s* and *z* that only use the diacritics nor the digraph *dj*, for instance *majka* 'mother'; these words will be denoted W_a. (b) Words that will be affected by our procedure because they contain either one or more letters that sometimes contain the diacritics or the digraph *dj*, even if the diacritic is actually not missing or a sequence *dj* represents the consonant cluster. For instance, both *zvono* 'bell', and *zvaka* (representing *žvaka* 'bubble-gum') and *podjednak* 'equal' and *takodje* (representing *takođe* 'also') will be included among the words of the type W_b.

Our procedure is based on the following basic ideas:

1. For each word W_b we intent to offer all possible Serbian words that use diacritics, including the original word if it exists in the language. For instance, if W_b = *liscem* then our procedure should identify the following candidates: *lišcem* 'face (diminutive, instrumental case)', *lišćem* 'foliage (instrumental case)', and the original word *liscem* 'male fox (instrumental case)'. Potential word forms *lisćem*, *lisčem*, *liščem* would not be considered since they do not exist in the Serbian language. Simultaneously, if W_b = *lucice*, our procedure would identify *lučice* 'port (diminutive, genitive case)' and *lučiće* 'to separate (future tense, 3rd person)', but not the original word since it does not exist in the Serbian language. If a W_b word exists in Serbian and no words with diacritics can be derived from it, no further actions are performed on it. For all these words we refer to the morphological dictionary of Serbian.

2. For each word W_b all the possible candidates $(W_{b1}, W_{b2}, \ldots, W_{bn})$ should be ranked according to the possibility of their occurrence in a text. In the case of the examples given above, *lišćem* is more frequent then both *liscem* and *lišcem* (the latter two have approximately the same frequency), while *lučice* is more frequent then *lučiće*. For these statistics we refer to the Corpus of Contemporary Serbian.

3. For each word W_b that has more then one possible candidate W_{bi} our procedure uses heuristics, lexicons and rules to choose the right one (more details in Section 4.).

4. For each word W_b that has no candidate at all our procedure does not offer any solution at this time.

In order to put the idea (1) into practice we have transformed the Serbian morphological dictionary (SMD)[6] into the appropriate format. Namely, we have extracted from SMD of the simple forms all those with the diacritic marks (we will name them W_c words) as well as those that are of the type W_b. We have transformed all words of the type W_c to words of the type W_b by stripping the diacritics and replacing *đ* with the digraph. At the same time, we removed all unnecessary information and recorded the original form. After that, we collated all identical word forms of the type W_b into one entry. For example, the original dictionary entries for the example given above were transformed in the following way:

```
liscem,lisac.N+Zool:ms6v   liscem,.X+CR=liscem
lišćem,lišće.N+Conc:ns6q   liscem,.X+CR=lišćem   liscem,.X+CR=liscem_lišćem_lišcem
lišcem,lišce.N+Dem:ns6q    liscem,.X+CR=lišcem
```

[6]Serbian Morphological Dictionaries cover both simple- and multi-word units (MWU). Dictionaries of simple-word units have more than 5 million grammatical forms generated from more than 141,000 lemmas, while the dictionaries of MWUs have more than 18,000 entries. To each grammatical form, whether it is a simple- or a multi word unit, various morphological, syntactic, semantic and other information is assigned (Krstev, 2008).

The obtained dictionary entry suggests that the word form W_b = *liscem* represents three Serbian word forms W_{b1} = *liscem*, W_{b2} = *lišćem*, W_{b3} = *lišcem*. We will dub a dictionary of such entries SMD_DR. The same procedure was applied both to the dictionary of simple words and to the dictionary of multi-word units (MWU).

Our dictionary is case-sensitive. In SMD, all common words are written in lower-case letters, and they match the text words in a case-insensitive manner. In SMD, the initial letters of all proper names are upper-case, while all letters of acronyms are written in upper-case, and they match the text words in a case-sensitive manner. The same applies to our dictionary SMD_DR. This approach should allow us to cover all possibilities without introducing any incorrect candidates. Take, for instance, the word forms *liže* 'to lick (present tense, 3rd person singular)' and *Lize* 'Lisa (feminine name, the genitive case)':

```
liže,lizati.V+Imperf:Psz   lize,.X+CR=liže
Lize,Liza.N+NProp:fs2v     Lize,.X+CR=Lize   Lize,Liza.X+CORR=Lize_liže
```

If the word form W_b = *lize* is written in lower-case, only one solution is offered – *liže*, if it is written in upper-case it can represent either a proper name or a verb form.

The resulting SMD_DR has 943,804 entries, 95% of which have only one candidate, while 4.5% have two candidates. The maximum number of candidates is 8 with one such entry:

```
Celice,.N+CR=Čeliče_Celiće_Ćeliće_Čeliće_čeliče_ćelice_celice_celiće
```

A word form W_b = *Celice* can represent a vocative of the surname *Čelik*, the accusative plural of the surnames *Celić*, *Ćelić* and *Čelić*, and various forms of the verbs *čeličiti* 'to steel, to harden' and *celiti* 'to heal' and the nouns *ćelica* 'bald spot (diminutive)' and *celica* 'ground'.

In order to implement the idea (2) we enhanced the dictionary SMD_DR with additional information from 100 million word excerpt collected in the Corpus of Contemporary Serbian (SrpKor).[7] From the list containing tokens and their respectable frequencies we calculated the total number of tokens ($totalNumTokens$).[8] The frequency calculation was different for dictionary entries with initial upper-case and for those that were entirely in lower-case. In the first case, only occurrences with the initial letter in upper-case were taken in the account, while in the second case, all occurrences were counted, regardless of the case.

$$relFreq = Round\left(\frac{freq \cdot 10000000}{totalNumTokens + 0.5}, 0\right)$$

Relative frequency for 0 was 0, for 1–10 was 1, for 11–21 was 2, 22–32 was 3,... Maximal absolute frequency was 3,706,356 and corresponding relative 340,596. For MWUs frequencies were not calculated.

Occurrence in the corpus	**Number of candidates**					
	1	**2**	**3**	**4**	**5**	>6
no occurrence in the corpus	742,941	24,004	1,509	47	9	2
%	82.82	57.92	42.09	10.22	9.68	12.50
at least one occurs in the corpus, but not all		9,325	1,274	261	66	12
%		22.50	35.54	56.74	70.97	75.00
all occur in the corpus	154,136	8,115	802	152	18	2
%	17.18	19.58	22.37	33.04	19.35	12.5
Total	897,077	41,444	3,585	460	93	16

Table 1: Distribution of entries in SMD_DR according to the number of candidates and their occurrence in the Corpus of Contemporary Serbian

The data in Table 1 may may create an impression that the problem of diacritics restoration is not very severe – only 5% of entries from SMD_DR offer more than one correction, and many of offered

[7] http://www.korpus.matf.bg.ac.rs/prezentacija/korpus.html

[8] Tokens are defined as the strings containing letters of the Serbian Latin alphabet only.

candidates do not occur in SrpKor at all, and can thus be routinely rejected (in many applications). However, a changed perspective may help us understand that this issue is actually a very pressing one. There are 147 entries with two candidates and 4 entries with 3 candidates, and all of them occur in SrpKor with a high frequency ($refFreq \geq 100$). A prominent example is W_b = *reci*, where each candidate occurs in a 100 million word corpus with a frequency higher than 2,000:

```
reci,.X+CR=reci(237)_reči(2607)_reći(1448)
```

All forms are frequent and highly ambiguous: *reci* can be a form of the nouns *reka* 'river' and *redak* 'line' and of the verb *reći* 'to say', while *reči* is a form of the noun *reč* 'word'.

4. The Procedure for Diacritic Restoration

In order to implement idea (listed as 3. in Section 3.) we have developed a set of rules and implemented them as cascades of the Finite-State Transducers (FSTs) in the corpus processing system Unitex[9]. The regular SMD is applied to texts that need to be corrected. Each word form is assigned a dictionary interpretation. This means that a W_b word form can either be left without an interpretation (the case of *lize* given in Section 3.) or some interpretation can be assigned to it (the case of *reci* – it can be either a form of the nouns *reka* or *redak*, as the dictionary suggests, or it has to be corrected).

We developed two working cascades: the first one retrieves data from the lexical resources – the dictionaries and the *n*-gram lists – and at the same time resolves straightforward cases, while the second one assigns one solution to each W_b word in a text by applying the set of rules, as given in Figure 1.

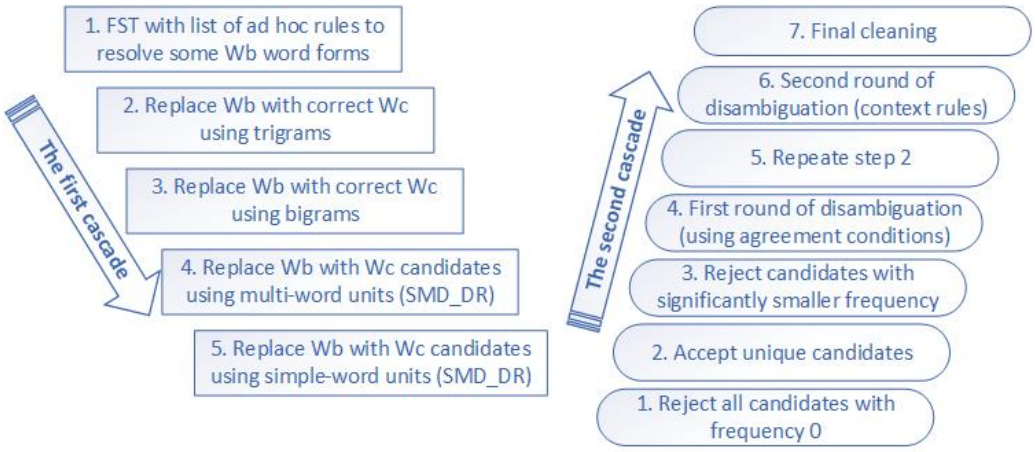

Figure 1: The two cascades of the Finite-State Transducers (FSTs)

The steps in the first cascade are:

1. This FST implements a list of *ad hoc* rules that are able to resolve some W_b word forms by analysing their context. For example, the word form *sto* can be a numeral 'hundred' or a noun 'table', or a W_b form of *što*, a relative and interrogative pronoun. The rules to confirm *sto* as correct are: (a) if *sto* is followed by word forms *puta* or *posto* as in the conventional phrases 'hundred times' and 'hundred percent' (in the later case *posto* is confirmed as well); (b) if it is followed by a numeral (e.g. *sto hiljada* 'hundred thousand'); (c) if it is followed by an adjective-noun phrase in the genitive case plural, as required by the Serbian agreement rules. The application of the rules (b) and (c) is possible because the SMD dictionaries were applied to the text before its correction. Naturally, they will work only if the context words are not the W_b words in need of correction.

2. From the list of the most frequent trigrams obtained from SrpKor we have chosen 30 most frequent ones that contain at least one W_b word and replaced it with a correct W_c word. Two rules of this type are: *sto se tice* → *što se tiče* 'concerning' and *na taj nacin* → *na taj način* 'in that way'.

[9]Unitex is a lexically-based corpus processing suite that has a strong support for finite-state processing – `unitexgramlab.org`

3. From the list of the most frequent bigrams obtained from SrpKor we have chosen 50 most frequent ones that contain at least one W_b word and replaced it with a correct W_c word. Two rules of this type are: *sto ce* → *što će* 'that will' and *je takodje* → *je takođe* 'is also'.[10]

4. A SMD_DR of the multi-word units is consulted and a W_b multi-word form is replaced by the list of candidates assigned to the corresponding entry. For instance, *klucna rec* → *ključna reč*(0) 'key word'. In the case of MWUs, a W_b word form has in most cases only one candidate; for that reason, the frequencies were not calculated for them and are currently not used by the second cascade.

5. A SMD_DR of the simple-word units is consulted and a W_b simple-word form is replaced by the list of candidates assigned to the corresponding entry. The W_b simple words are replaced if they (a) do not appear as such in SMD (the unknown words) or (b) they do appear in SMD and may be the correct choice.

After the application of the first cascade a new version of a text is obtained in which almost all W_b words are either confirmed, or corrected or have a list of possible solutions assigned to them. However, all this operations are not done without a trace; on the contrary, all modifications, as well as their type, are visible in the text, because that is important for the work of the second cascade. After the application of the first cascade a text is transformed (see Table 2).

the source text	**the annotated text**
Jer je imao dovoljno vremena da spreci zlocin cak i nakon reci kojima je podstrekivao sina.	Jer je imao dovoljno vremena da 5a_(spreči(302)_spreći(0)) 5a_(zločin(456)) 3_(čak i) 1_(nakon reči) kojima je podstrekivao 5b_(sina(518)_šina(54)).
U novinama vise nije bilo ni reci o ratnoj steti.	U novinama 5b_(vise(35)_više(17628)) 1_(nije bilo ni reči o) 4_(ratnoj šteti(0)).

Table 2: Two sentences without the diacritics (left), and their annotated version after the application of the first cascade (right). Numbers correspond to the steps in the cascade that perform the annotation.

The role of the second cascade is to produce a clean text with the diacritics restored. For that purpose, dictionaries (SMD) are applied to an annotated text. As a result, all words in the text (W_a, W_b and W_c) obtain one or more dictionary interpretation. Those W_b words that are not words in Serbian (or are not in SMD), of course, do not get a dictionary interpretation. Words of the type W_c are those that resulted from the work of the first cascade. The steps of the second cascade are the following:

1. All candidates that do not occur in SerKor (frequency is 0) are rejected, if there is an alternative candidate occurring in SerKor (frequency higher than 0), e.g. *1_5b_(zemlja(1518)_žemlja(0))* → *1_5b_(zemlja(1518))* (*zemlja* 'earth/ground', *žemlja* 'bread roll (Ijekavian)').

2. This FST accepts unique candidates (only one candidate exists for a W_b word according to SMD), e.g. *5a_(ništa(3531))* → *ništa* 'nothing'.

3. This FST rejects candidates which have significantly lower frequency of occurrence than some alternative candidate. This FST is subject to changes depending on user's views what "significantly smaller" means: ten times less, hundred times less, less than hundred/greater than hundred, etc. For instance, *5b_(tacno(1)_tačno(1219))* → *5b_(tačno(1219))* (*tačno* 'right/correct', *tacno* 'tray (the vocative case').

4. This FST performs the first round of disambiguation. It looks in the unambiguous context (either W_a words or the previously disambiguated words) of the list of possible candidates that can confirm

[10]The size of lists of bigrams and trigrams was chosen rather arbitrarily in belief that the ambiguity problem will not occur among the most frequent n-grams. This decision has to be reconsidered in future.

at least one of the candidates and reject others. Possibilities are: (a) an adjective that is unambiguous is followed by a list of candidates among which is a noun that agrees with the adjective in the gender, the number and the case, for instance, *žrtvene 5b_(jarče(2)_jarce(1))* → žrtvene 5b_(jarce(1)) 'lit. sacrificial goat; scapegoat'; (b) a noun that is unambiguous is preceded by a list of candidates among which there is an adjective that agrees with the noun in the gender, the number and the case, e.g., *5b_(čelo(212)_celo(181)) popodne* → *5b_(celo(181)) popodne* 'whole afternoon'; (c) a preposition that is unambiguous is followed by a list of candidates among which there is an adjective, a noun or a pronoun in the case required by the preposition, e.g., *Iz 5b_(reci(237)_reči(2607)_reći(1448))* → *Iz 5b_(reči(2607)*. If among the candidates at least one is confirmed by the context, those that are not confirmed are rejected, while all that are confirmed are accepted.

5. The step 2) is repeated, because, as a result of the intervening steps, some candidates may have become unique.

6. The second round of disambiguation takes into account the context with regard to: (a) some specific candidate lists, like *reci(237)_reči(2607)_reći(1448)* or *nišu(820)_nisu(9675)*, or some more general cases: (b) the reflexive particle *se* followed by a form of a reflexive verb; (c) the negative form of the auxiliary verb *neće* followed by an impersonal verb form (such as the infinitive); (d) the particle *ne* followed by a personal verb form. Examples of these decisions are: (a) *1_5b_(nišu(820)_nisu(9675))* → *Nišu* (since an upper-case initial letter is required in the middle of a sentence, it most probably refers to *Niš*, a city in Serbia); (b) *(da) se 5a_(suši(30)_šuši(1))*; → *(da) se suši* '(to) dry itself'; (c) *(da se) neće 5b_(obuci(49)_obući(30)_obuči(5))* → *(da se) neće obući* 'not (to) dress oneself'; (d) *ne 5b_(tući(67)_tući(12)_tuci(1))* (me) → *ne tuci* (me) 'do not beat (me)'.

7. In the last step, appart from the final cleaning (such as the deletion of the duplicates) the last decisions are made in order for the resulting text to be completely resolved: (a) in 5b cases (among candidates one is without diacritics), the one without diacritics is chosen; (b) in 5a cases (among candidates all are W_c words) the one with the highest frequency is chosen.

5. Evaluation

In order to estimate the extent of the problem as well as how various rules contributed to its solution in Table 3 we present the data that correspond to the annotation and disambiguation of steps of the first and the second cascade when applied to two sample texts, one belonging to Ekavian and the other to Ijekavian pronunciation. The samples have similar size, the Ekavian has 2,024 word tokens, Ijekavian 1,930 word tokens. One cannot assume that a number of changes by the first and the second cascade should sum up to equal totals, since the second cascade sometimes resolves several annotations together, while, on other hand, other annotations are addressed in several steps. Also, the disambiguation steps, both in the first and in the second cascade, sometimes resolve more than one W_b word. The importance of the dictionaries is justified by the fact that in both texts, neither of which is very long, there were unique candidates with 0 frequency in SrpKor (5 in the Ekavian text and 7 in the Ijekavian text).

The first cascade			**The second cascade**		
Step	**Text Ek**	**Text Ijk**	**Step**	**Text Ek**	**Text Ijk**
1 (disambiguation)	8	12	1 ($W_c \notin$ SrpKor)	23	20
2 (trigrams)	1	1	2 (W_c is unique)	210	311
3 (bigrams)	9	16	3 (frequency)	55	64
4 (MWU)	3	0	4 (disambiguation 1)	12	5
5a ($W_b \notin$ SMD)	201	257	5 (W_c is unique)	96	78
5b ($W_b \in$ SMD)	138	145	6 (disambiguation 2)	2	4
			7 (final cleaning)	64	91

Table 3: The contribution of each step in cascades to the diacritic restoration.

For evaluation we used a set of 65 correctly typed documents of different length, type and domain. These documents are new, that is they are not part of the SrpKor used for the calculation of frequencies. We prepared all documents for evaluation by stripping the diacritics, and than applying the restoration procedure.[11] First, we aligned sentences of the source and restored files. After that, words were aligned and compared. For each document we counted the number of words that were correctly restored TP (true positive), number of words that were rightly not restored TN (true negative), number of erroneously restored words FP (false positive), number of words without expected restoration FN (false negative). In Table 4 the first two rows contain absolute and relative values that were calculated by taking into account only different word tokens (types), while the second two rows contain results that were obtained by counting all the occurrences.

Calculating	**Total**	$\mathbf{W_b}$	$\mathbf{W_a}$	**TP**	**TN**	**FP**	**FN**
by types	836,764	549,978	286,786	214,249	318,232	3,198	14,299
relative %	1.000	65.727	34.273	38.956	57.863	0.581	2.600
by tokens	3,493,785	1,682,680	1,811,105	575,618	1,070,181	6,211	30670
relative % (W_b)	1.000	48.162	51.838	34.208	63.600	0.369	1.823
relative % (Total)				16.475	30.631	0.178	0.879

Table 4: The basic data about the evaluation set, and evaluation results on the whole set.

We calculated the standard measures: the precision, the recall, the accuracy and F_1 for each document. The average, the maximum and the minimum values of these measures as observed in this set are presented in Table 5. Note that all these measures were calculated only for the words that were candidates for restoration (W_b words). The overview of the fluctuation of four measures for a few selected documents is given in Figure 2. It can be seen that, except in the case of one specific document, the precision is always significantly higher than the recall.

Calculating		**Precision**	**Recall**	**Accuracy**	**F-measure**		
types	average %	98.529	93.744	96.819	96.077		
	maximum %	99.677	96.142	98.097	97.671		
	minimum %	91.611	89.534	94.416	93.005	**Correct**	**Incorrect**
tokens	average %	98.933	94.941	97.808	96.896	98.944	1.056
	maximum %	99.958	98.428	99.358	99.187	99.647	2.247
	minimum %	95.752	88.467	95.029	92.855	97.753	0.353

Table 5: Precision $P = tp/(tp + fp)$, Recall $R = tp/(tp + fn)$, Accuracy $Acc = (tp + tn)/(tp + tn + fp + fp)$, F-measure $F_1 = 2PR/(P + R)$; correct words $((T - (fp + fn))/T)$, incorrect words $(fp + fn)/T$.

We compared results that were obtained in our sample of 65 texts with those published for Croatian in (Šantić et al., 2009) and Croatian and Serbian in (Ljubešić et al., 2016). In order to compare our results with those presented in (Šantić et al., 2009: 317) we calculated the percentages of correct and incorrect words in the restored text (the two last columns in Table 5) and we observed that, on the average, we obtained a slightly higher number of correct words (98.94 vs. 98.81). In order to compare our results with those discussed in (Ljubešić et al., 2016: 3615) we calculated percentages of false positives and false negatives relative to the total number of tokens (not just W_b) and we observed that our precision (0.18 vs. 0.12) and recall (0.88 vs. 0.41) are slightly lower.[12] In future we plan to apply our procedure and some language independent tools to the same set of texts (both standard and non-standard) in order to test and evaluate different approaches in the proper manner.

[11] The same approach was previously employed by many authors who were concerned with similar problems: (Šantić et al., 2009: 316), (Tufiş and Ceauşu, 2008: 6), (Iftene and Trandabat, 2009: 39), (Francom and Hulden, 2013: 3).

[12] In both cases we compared only results that authors in (Šantić et al., 2009) and (Ljubešić et al., 2016) obtained on a fully diacriticized text and/or on a standard text.

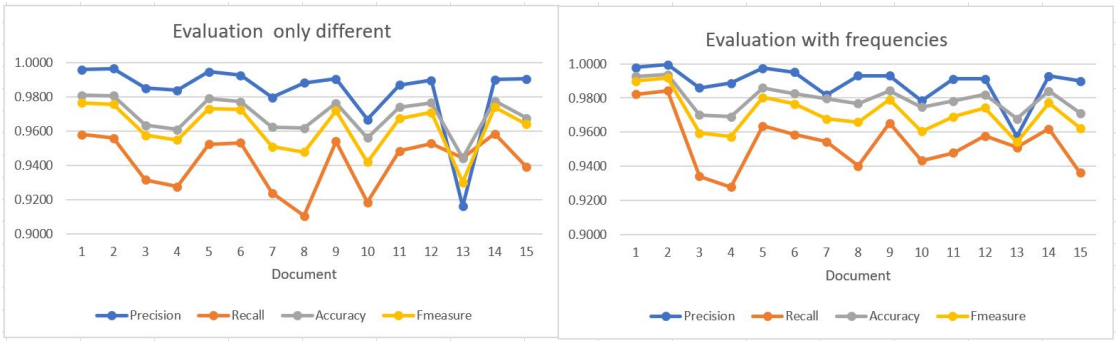

Figure 2: The evaluation report for a subset of documents: left – calculation on the W_b types; right – calculation on the W_b tokens

6. Concluding Remarks and Future Work

In this paper, we have shown that the problem of the diacritic restoration can be successfully solved by using a rule-based approach that relies on the lexical resources. This solution exhibits the advantage of transparency which is usually characteristic of such methods. Namely, any successful or unsuccessful change of the text can be easily explained and that can lead to the improvement of the solution. Our approach has some additional advantages. First, the improvement of the lexical resources, longer and more reliable lists of bigrams and trigrams, the enrichment of the e-dictionaries, particularly dictionaries of MWUs, will contribute to system's better results. Second, the system is highly modular, which means that the order of steps can be easily changed and some steps removed or replaced for particular purposes. There are some disadvantages as well. First of all, considerable time was invested in its development. Also, its performance time does not make this solution applicable for interactive applications (e.g. mobile devices). Second, the extensive use of dictionaries implies that the procedure works only on the standard and the reasonably correct texts (missing only the diacritics)This means that its performance would be less impressive on non-standard texts, such as the Twitter posts. For non-standard texts, the procedure has to be used in conjunction with the other tools that deal with abbreviations, non-standard spelling, foreign words, etc., as we have already done (Mladenović et al., 2017).

The solution discussed in this paper can be adapted for solving some similar problems. First, the same solution can be applied to the texts that are missing diacritics only partially. In that case, only dictionaries SMD_DR have to be modified. For instance, for the example from Section 3. the dictionary entries would be, not only

```
liscem,.X+CR=liscem_lišćem_lišcem,
```

but also

```
lišcem,.X+CR=lišcem_lišćem
lisćem,.X+CR=lišćem.
```

Next, very promising experiments have already been conducted aiming to correct the texts obtained by OCR and to transform Serbian texts from one variant to another (from Ekavian to Ijekavian and vice versa, e.g. *lepa devojka* (Ek) 'beautiful girl' $\leftrightarrow$ *lijepa djevojka* (Ijk)).

Finally, our goal is also to experiment with the hybrid solutions that would use explicit language models in the candidate selection phase.

Acknowledgements

This research was partially supported by the Serbian Ministry of Education and Science under the grants #III 47003 and 178006.

References

Acs, J. and Halmi, J. (2016). Hunaccent: Small Footprint Diacritic Restoration for Social Media. In Utka, A., Vaičenonienė, J., and Butkienė, R., Eds., *Normalisation and Analysis of Social Media Texts (NormSoMe) Workshop Programme*, pages 1–4.

Adali, K. and Eryiğit, G. (2014). Vowel and diacritic restoration for social media texts. In *Proceedings of the 5th Workshop on Language Analysis for Social Media (LASM)*, pages 53–61.

Alghamdi, M., Muzaffar, Z., and Alhakami, H. (2010). Automatic restoration of Arabic diacritics: a simple, purely statistical approach. *Arabian Journal for Science and Engineering*, 35(2C):125–135.

Atserias, J., Fort, M. F., Nazar, R., and Renau, I. (2012). Spell Checking in Spanish: The Case of Diacritic Accents. In *LREC*, pages 737–742.

Belinkov, Y. and Glass, J. R. (2015). Arabic Diacritization with Recurrent Neural Networks. In *EMNLP*, pages 2281–2285.

El-Sadany, T. and Hashish, M. (1988). Semi-automatic vowelization of Arabic verbs. In *10th NC Conference, Jeddah, Saudi Arabia.*

Francom, J. and Hulden, M. (2013). Diacritic error detection and restoration via part-of-speech tags. In *Proceedings of the 6th Language and Technology Conference*.

Gal, Y. (2002). An HMM Approach to Vowel Restoration in Arabic and Hebrew. In *Proceedings of the ACL-02 Workshop on Computational Approaches to Semitic Languages*, SEMITIC '02, pages 1–7, Stroudsburg, PA, USA. Association for Computational Linguistics.

Haertel, R. A., McClanahan, P., and Ringger, E. K. (2010). Automatic Diacritization for Low-resource Languages Using a Hybrid Word and Consonant CMM. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 519–527, Stroudsburg, PA, USA. Association for Computational Linguistics.

Hládek, D., Staš, J., and Juhár, J. (2013). Diacritics Restoration in the Slovak Texts Using Hidden Markov Model. In *Language and Technology Conference*, pages 29–40. Springer.

Ibraheem, A. O. (2017). *A Concept Paper on a New Model for Automatic Diacritic Restoration*.

Iftene, A. and Trandabat, D. (2009). Recovering diacritics using Wikipedia and Google. In *Knowledge engineering: Principles and techniques, Proceedings of the international conference on knowledge engineering KEPT2009*, pages 37–40.

Kapočiūtė-Dzikienė, J., Davidsonas, A., and Vidugirienė, A. (2017). Character-Based Machine Learning vs. Language Modeling for Diacritics Restoration. *Information Technology And Control*, 46(4):508–520.

Krstev, C. (2008). *Processing of Serbian – Automata, Texts and Electronic dictionaries*. Faculty of Philology, University of Belgrade.

Ljubešić, N., Erjavec, T., and Fišer, D. (2016). Corpus-based diacritic restoration for South Slavic languages. In *Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC 2016).*

Mladenović, M., Krstev, C., Mitrović, J., and Stanković, R. (2017). Using Lexical Resources for Irony and Sarcasm Classification. In *Proceedings of the 8th Balkan Conference in Informatics*, BCI '17, pages 13:1–13:8, New York, NY, USA. ACM.

Novák, A. and Siklósi, B. (2015). Automatic Diacritics Restoration for Hungarian. Association for Computational Linguistics.

Petrică, L., Cucu, H., Buzo, A., and Burileanu, C. (2014). A Robust Diacritics Restoration System Using Unreliable Raw Text Data. In *Spoken Language Technologies for Under-Resourced Languages*.

Pham, T.-H., Pham, X.-K., and Le-Hong, P. (2017). On the Use of Machine Translation-Based Approaches for Vietnamese Diacritic Restoration. *arXiv preprint arXiv:1709.07104*.

Šantić, N., Šnajder, J., and Bašić, B. D. (2009). Automatic Diacritics Restoration in Croatian Texts. *INFuture2009: Digital Resources and Knowledge Sharing*, pages 309–318.

Tufiş, D. and Ceauşu, A. (2008). DIAC+: A professional diacritics recovering system. *Proceedings of LREC 2008*.

Ungurean, C., Burileanu, D., Popescu, V., Negrescu, C., and Dervis, A. (2008). Automatic diacritic restoration for a TTS-based e-mail reader application. *UPB Scientific Bulletin, Series C*, 70(4):3–12.

Yarowsky, D. (1999). A comparison of corpus-based techniques for restoring accents in Spanish and French text. In *Natural language processing using very large corpora*, pages 99–120. Springer.

Zakon, Ed. (2010). *Zakon o službenoj upotrebi jezika i pisma[Law on Official Usage of Language and Script]*. Službeni glasnik Republike Srbije.