Corpus-Based Diacritic Restoration for South Slavic Languages

Nikola Ljubešić,[*‡] **Tomaž Erjavec,**[*] **Darja Fišer**[†*]
[*] Department of Knowledge Technologies, Jožef Stefan Institute
Jamova cesta 39, SI-1000 Ljubljana, Slovenia
tomaz.erjavec@ijs.si
† Faculty of Arts, University of Ljubljana
Aškerčeva cesta 2, SI-1000 Ljubljana, Slovenia
darja.fiser@ff.uni-lj.si
[‡] Dept. of Information and Communication Sciences, University of Zagreb
Ivana Lučića 3, HR-10000 Zagreb, Croatia
nikola.ljubesic@ijs.si

Abstract

In computer-mediated communication, Latin-based scripts users often omit diacritics when writing. Such text is typically easily understandable to humans but very difficult for computational processing because many words become ambiguous or unknown. Letter-level approaches to diacritic restoration generalise better and do not require a lot of training data but word-level approaches tend to yield better results. However, they typically rely on a lexicon which is an expensive resource, not covering non-standard forms, and often not available for less-resourced languages. In this paper we present diacritic restoration models that are trained on easy-to-acquire corpora. We test three different types of corpora (Wikipedia, general web, Twitter) for three South Slavic languages (Croatian, Serbian and Slovene) and evaluate them on two types of text: standard (Wikipedia) and non-standard (Twitter). The proposed approach considerably outperforms charlifter, so far the only open source tool available for this task. We make the best performing systems freely available.
Keywords: computer-mediated communication, diacritic restoration, South-Slavic languages

1. Background and Motivation

In computer-mediated communication, such as emails, instant messages, tweets etc. users of Latin-based scripts often replace characters with diacritics with their ASCII equivalents for ergonomic reasons, especially when typing on tablets and smartphones. Such text is typically easily understandable to humans but very difficult for computational processing because many words without the diacritics are ambiguous or unknown. This is why diacritics restoration is an active research area.

There are two main approaches to diacritic restoration: letter-level and word-level (Mihalcea and Nastase, 2002). While the letter-level approaches generalise better and therefore do not require a lot of training data (Tufiş and Ceauşu, 2008), word-level approaches (Šantić et al., 2009; Novák and Siklósi, 2015) tend to yield better results, especially for languages where the diacritics have a grammatical and/or semantic role. Most approaches in this category rely on a lexicon (Yarowsky, 1999; Tufiş and Chiţu, 1999; Šantić et al., 2009) but these are expensive resources that are often not available, especially for less-resourced languages. Additionally, these approaches do not deal with word forms not covered by the lexicon, an issue that is all the more accentuated in non-standard text. Other approaches require reliable morphological, syntactic or semantic tools for the language at hand, at the very least a PoS tagger (Yarowsky, 1994; Simard, 1998; Tufiş and Chiţu, 1999), still an obstacle for many lesser resourced languages or language varieties, such as nonstandard user-generated content.

Recently, statistical machine translation was also used for diacritic restoration of Hungarian (Novák and Siklósi, 2015). However, we consider the method to be too complex for the task at hand as diacritic restoration does not require calculating word or phrase alignments, does not use phrases in the translation model and does not need to perform reordering. Interestingly, although diacritic restoration is a well known problem, the only open source tool available for this task is charlifter[1].

The presented work focuses on three: Slovene, Croatian, and Serbian, the latter as written in the Latin alphabet.[2] The three languages belong to the South Slavic languages, where Croatian and Serbian are rather close and, for the most part, mutually intelligible, with Slovene more distant. All three languages use the so called Gaj alphabet and distinguish five letters written with diacritics, namely *č*, *š*, *ž*, *ć* and *đ* and their upper case equivalents; the letters represent palatalised versions of their diacriticless equivalents. Slovene uses only the first three of these letters, however, the latter two are also found in Slovene texts, typically in surnames. While the percentages vary slightly, about 13% percent of the tokens in running text contain one or more letters with diacritics in the three languages.

To the best of our knowledge, the only previous attempt to perform diacritic restoration on Croatian was Šantić et al. (2009). The authors approach the problem by identifying ambiguous tokens in a morphological dictionary and using a bigram language model built from 2.5 million words of newspapers and literary works for their disambiguation. They achieve a dictionary-only baseline of 97% and 98.81% accuracy when using the language model as well. The system is not freely available. We are not aware of any prior work on this problem for Slovene nor Serbian. In this paper we present diacritic restoration models that are

[1] https://sourceforge.net/projects/lingala/files/charlifter/

[2] Serbian is written both in Cyrillic and in Latin alphabet, but the former is not relevant to our experiments.

trained on corpora as these are nowadays not difficult to acquire. We test three different types of corpora (Wikipedia, general web, Twitter) for three South Slavic languages and evaluate them on two types of text: standard (Wikipedia) and non-standard (Twitter). We outperform charlifter considerably (average error reduction of 86%) and publish the systems as well as the best performing models for each language as open source.

2. Datasets

2.1. Resources

For training our models we use, on each of the three languages, three types of data: Wikipedia texts, general Web texts and non-standard texts from Twitter. As our goal is to be able to cover texts which are written in standard language as well as those that are often non-standard, we use, as our test sets, texts from Wikipedia for the former and those from Twitter for the latter.

The three Wikipedia corpora were compiled from Wikipedia dumps with a generic Wikipedia corpus extraction script. We keep only sentences containing 100 characters or more, thereby removing most of the remaining Wikipedia markup noise.

The Web corpora come from the WaC corpora of the three focus languages (Ljubešić and Klubička, 2014; Erjavec and Ljubešić, 2014). As we wanted to obtain a good training set for the system, and Web data can contain texts without diacritics, we retained for our datasets only those texts where at least 20% of tokens contain diacritics. This is quite a strict filter but given the large volumes of web data available the resulting dataset is still very large.

The Twitter corpora were complied from large collections of tweets collected since mid-2013 with the TweetCaT tool (Ljubešić et al., 2014). The discrimination between Croatian and Serbian users, as these languages are very similar, was performed with a dedicated tool (Ljubešić and Kranjčić, 2015). In order for our Twitter datasets to contain mostly non-standard language, we retained only those tweets that were automatically annotated with a low or medium level of linguistic standardness (Ljubešić et al., 2015). Finally, we discarded all the tweets for which less than 10% of the tokens contain diacritics. In this case our filtering criterion was less cautious than in the case of ordinary web data as the amount of available Twitter data was much lower.

2.2. Preprocessing

The resulting datasets were tokenised and sentence segmented, except for the Twitter datasets, where the entire tweet was taken as the basic unit. Furthermore, all the tokens were converted to lower case because this decreases data sparsity. During our initial experiments retaining letter casing proved not to be informative for the languages in question. Re-casing the tokens after diacritics were restored is a straightforward task as in most cases the number of letters in a token does not change.

We split our datasets into training, development and test data. We removed sentence / tweet duplicates from our training data as, based on our initial experiments, removing duplicates did not hurt the perfomance of the systems. The sizes of the training datasets are given in Table 1.

	hr	sr	sl
Wikipedia	28,071,421	33,813,463	19,616,468
Web	268,861,709	102,792,821	131,326,854
Twitter	1,949,341	13,716,161	6,733,457
Σ	297,686,292	148,870,067	157,021,950

Table 1: Training datasets (in number of words)

To tune our systems to standard and non-standard data we put aside development datasets from the Wikipedia and the Twitter datasets, containing 10,000 text instances for each text type and language.

For testing on standard data we took additional 10,000 text instances from the Wikipedia datasets. For testing on non-standard data, given that we can assume that there are diacritic-related errors or omissions even in the filtered data, we produced test sets of 2,000 tweets for each of the three languages that were proofread by a linguist. These tweets come from the same initial collection of non-standard tweets but do not follow the limitation on the number of words containing diacritics as we wanted the test set to be representative of the non-standard language in general.

We did not remove sentence / tweet duplicates from development and test data as we wanted to simulate a maximally realistic scenario.

3. Experimental setup

We define the diacritic restoration problem as a token-level translation problem where each input token is "translated" to its diacritised variant. To train the translation system we simply remove diacritics from the original texts, thereby obtaining a token-aligned parallel dataset.

We use two approaches to solving this word-by-word-translation problem:

- lexicon approach (lexicon) – applying the most frequent translation, as observed in the training data
- corpus approach (TM+LM) – combining the information about the probability of a translation (TM) and the probability of observing a translation in the given context (LM) via a simple log-linear model estimated on our development data

Some approaches like (Šantić et al., 2009) use context probabilities only and disregard the translation probabilities. We too have experimented with such an approach in the early stages. While this approach has shown to outperform the lexicon approach, it performed significantly worse than the full corpus approach. Given that the time and space complexity of the pure LM approach is very similar to the TM+LM one, we discarded it from our final experiments.[3]

[3]Inspecting the results of the pure LM approach showed that a large number of errors was due to specific contexts present in both the test and the training data where on the training data side diacritics were mistakenly not present. By combining both translation and context probabilities most of such training data impurities are dealt with on the side of the translation probabilities.

Training data	Method	wiki			tweet		
		hr	sr	sl	hr	sr	sl
	original	0.8615	0.8614	0.8844	0.8715	0.8397	0.8790
	charlifter	0.9790	*0.9674*	0.9706	0.9508	*0.9436*	0.9330
wiki	lexicon	0.9926	0.9916	0.9925	0.9784	0.9653	0.9607
	TM+LM	0.9938	0.9930	0.9948	0.9803	0.9669	0.9669
tweet	lexicon	0.9646	0.9737	0.9810	0.9860	0.9888	0.9796
	TM+LM	0.9650	0.9753	0.9822	0.9863	0.9906	0.9882
web	lexicon	0.9930	0.9899	0.9923	0.9911	0.9835	0.9785
	TM+LM	0.9946	0.9913	0.9949	0.9933	0.9865	0.9878
wikitweetweb	lexicon	0.9936	0.9924	0.9933	0.9917	0.9893	0.9820
	TM+LM	**0.9957**	**0.9947**	**0.9962**	**0.9938**	**0.9917**	**0.9912**
Error reduction		32.81%	30.26%	43.28%	25.30%	22.43%	51.11%

Table 2: Results obtained with different settings on different training and testing corpora; the last row refers to the error reduction of the TM+LM method in comparison to the lexicon method, both trained on the wikitweetweb data. Charlifter does not have a model for Serbian so Croatian was used; to indicate this these results are in italic.

For estimating the probability of a token translation in both approaches we use the maximum likelihood estimate of a diacritised form given the dediacritised one and encode it in a simple hash-of-hashes structure, while for estimating the context probability we use KenLM (Heafield, 2011) with default parameters.

For estimating the two parameters of our log-linear model, λ_{TM} and λ_{LM}, we perform an exhaustive search over all their combinations in the [0.0,1.0] range with a 0.1 step. As our objective function we use token accuracy averaged over neighboring points, performing thereby simple search space smoothing.

4. Results

4.1. Comparison of the settings

The results of the experiments on various training and testing data are given in Table 2. We compare our results to the *original* baseline (no intervention) and with *charlifter*. On standard data the baseline gives 14% token error for Croatian and Serbian and 12% for Slovene, which uses fewer diacritics. We observe a similar error rate on non-standard text, with a much higher error rate on Serbian data for which at this point we do not have an explanation for. *Charlifter* outperforms the *original* baseline with 3% token error on standard and 6% on non-standard data. Even the simple lexicon approach trained on Wikipedia data outperforms charlifter in both domains on all three languages by a wide margin.

The best performing system is obtained using the TM+LM method trained on all available data, i.e. a concatenation of the Wikipedia, Twitter and web data.

As expected, the problem is overall easier on standard than on non-standard data. On each single setting and language TM+LM outperforms the simple lexicon approach. The best performing system reduces the error (encoded in the last row of Table 2) between 22% and 51%.

The optimised λ parameters for Croatian and Serbian are quite similar with higher weights for LM on the Wikipedia data and slightly higher weights for TM on the tweet data. For Slovene, on the other hand, the system tuned both on standard and non-standard data gives a much higher weight to the LM, suggesting the final decision is more context-dependent in Slovene than in the other two languages. This resonates with our intuition as there are more highly frequent and ambiguous words in Slovene than in the other two languages, such as *se* vs. *še* (reflexive particle vs. *more / still*) or *nas* vs. *naš* (*us* vs. *our*).

It is interesting to note that using Web data for training a model outperforms Wikipedia data even in-domain in every language except Serbian for which we have the lowest amount of Web data. We can therefore assume that the reason for the Web data to be competitive with in-domain data on the Wikipedia test set is its sheer amount. On non-standard language, however, apart from the amount of data, the domain of the training data matters just as well. This can be clearly observed on Croatian where a small amount of tweets (2 million words) significantly outperforms the 14 times bigger dataset from Wikipedia.

Another interesting observation is that on the best performing system when all data is merged, adding Twitter to Web data improves the results even on the Twitter test set, although the amount of Twitter data is 10 to 200 times lower than the amount of Web data. A similar phenomenon can be observed on the Wikipedia test data, which suggests that the three datasets are complementary regarding the phenomena present in both Twitter and Wikipedia texts.

We additionally inspect the best performing system by calculating precision, recall, percentage of false positives and percentage of false negatives in Table 3. For Slovene standard text, approximately each 600th word will be erroneously rediacritised and each 500th word will fail to be rediacritised. For non-standard text, each 300th will be misrediacriticised and each 200th will be skipped. Precision is consistently higher than recall, even though we optimise on accuracy, the reason for which is probably the word forms containing diacritics that are not observed in the training data.

4.2. Crosslingual experiments

We ran an additional set of experiments where we used the models trained on one language to restore diacritics in the other language. The results, which are presented in Table 4, show that Croatian and Serbian are mutually useful but that using Serbian data on Croatian seems to be more advanta-

	wiki				tweet			
	P	R	FP	FN	P	R	FP	FN
hr	0.9901	0.9784	0.14%	0.30%	0.9831	0.9681	0.21%	0.41%
sr	0.9908	0.9705	0.12%	0.41%	0.9837	0.9642	0.26%	0.57%
sl	0.9852	0.9819	0.17%	0.21%	0.9745	0.9524	0.30%	0.58%

Table 3: Precision, recall, false positive and false negative error token percentage of the best performing system

train	test	wiki	tweet
sl	hr	0.9486	0.9518
sr	hr	0.9915	0.9884
hr	hr	0.9957	0.9938
sl	sr	0.9457	0.9281
hr	sr	0.9886	0.9839
sr	sr	0.9947	0.9917

Table 4: Experiments on cross-lingual diacritic restoration

geous than the opposite. A possible explanation is the lower level of standardisation / prescription in Serbian. Serbian is also a more informative training set because it uses both the ekavian (e.g. mleko for milk) and the ijekavian dialects (e.g. mlijeko for milk, while Croatian uses the ijekavian dialect only.

On the other hand, using Slovene data for restoring diacritics on Croatian or Serbian, or vice versa, does cut the token error of dediacritised text by half, but is far from the excellent results obtained with the models trained on the target language. An interesting phenomenon is that Slovene works much better on Croatian non-standard data than on Serbian non-standard data, probably because of the dialectal similarity of Croatian and Slovene.

4.3. Error analysis

In order to gain insight into the nature of the errors made by our best performing model, we performed an error analysis on the Wikipedia and Twitter datasets for Slovene.

We manually examined 100 errors in each dataset and classified them into 9 categories. As Table 5 shows, the main reasons for errors are quite different in the two datasets. In Wikipedia, the biggest problems are proper names unseen in the training set (30%) and domain-specific rare words, often derived from foreign words or proper names (28%); in both cases, these words do not respect the orthographic patterns followed by native Slovene words. In tweets, on the other hand, ambiguous words that exist both with and without diacritics (37%) and the omission of spaces (31%), either for saving space and time, or as a common phenomenon in hashtags, are the main causes of erroneous rediacritisation.

It is the ambiguous words (21% in standard text and 37% in non-standard text) that present the most serious errors, which is why our efforts need to be focused on resolving these first. They could be better handled either with much more data in our language model, which does not seem to be very realistic, or with abstracting from surface forms to morphosyntactic categories as ambiguous words almost never share this category. A quite simple approach to utilising morphosyntax in rediacritisation could be training morphosyntactic taggers on dediacritised data and resolving ambiguity through morphosyntactic tagging of data lacking diacritics. The applied tags should be highly useful in resolving this ambiguity.

	wiki	tweet
proper noun	30	6
rare word	28	6
ambiguous word	21	37
foreign word	8	3
typo	6	6
tokenization issue	4	31
correct variant	3	3
multiplied letters	0	5
test set error	0	3
total	100	100

Table 5: Error analysis on Slovene

5. Conclusions

In this paper we proposed a corpus-based approach to learning rediacritisation models for texts of varying standardness degrees in South Slavic languages. Our results show that inexpensive corpus-based methods drastically outperform the only freely available tool charlifter.

Best results are obtained when taking into account both the probability of a form given its dediacritised version and the probability of the form in the given context (TM+LM). On the other hand, very good results can be obtained already when using just the probability of each form given its dediacritised version (lexicon). It is important to note that this approach requires much less memory: the best Slovene system using just the lexicon approach needs around 1GB of memory, while the TM+LM approach uses almost 10GB.

Overall, the best type of data for training diacritic restorers of both standard and non-standard texts is Web data. While it outperforms even in-domain Wikipedia data, mostly because of its amount, adding smaller volumes of strictly non-standard data does still improve the results on non-standard texts.

The code and the models of the best performing systems for all three languages in both variants, the lexicon and the TM+LM one, are made available on `https://github.com/uzh/reldi/tree/master/tools/diacritic_restoration`.

6. Acknowledgements

The work described in this paper was funded by the Slovenian Research Agency national basic research project J6-6842 “Resources, Tools and Methods for the Research of Nonstandard Internet Slovene”, the EU FP7 grant

agreement PIAP-GA-2012-324414 (Abu-MaTran) and the Swiss National Science Foundation grant IZ74Z0_160501 (ReLDI).

7. Bibliographical References

Erjavec, T. and Ljubešić, N. (2014). The slWaC 2.0 Corpus of the Slovene Web. In *Language technologies: Proceedings of the 17th International Multiconference Information Society IS2014*, Ljubljana, Slovenia.

Heafield, K. (2011). KenLM: Faster and smaller language model queries. In *In Proc. of the Sixth Workshop on Statistical Machine Translation*.

Ljubešić, N. and Klubička, F. (2014). {bs,hr,sr}WaC – web corpora of Bosnian, Croatian and Serbian. In *Proceedings of the 9th Web as Corpus Workshop (WaC-9)*, pages 29–35, Gothenburg, Sweden. Association for Computational Linguistics.

Ljubešić, N. and Kranjčić, D. (2015). Discriminating between Closely Related Languages on Twitter. *Informatica*, 39(1):1–8.

Ljubešić, N., Fišer, D., and Erjavec, T. (2014). TweetCaT: a Tool for Building Twitter Corpora of Smaller Languages. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, Reykjavik, Iceland. European Language Resources Association (ELRA).

Ljubešić, N., Fišer, D., Erjavec, T., Čibej, J., Marko, D., Pollak, S., and Škrjanec, I. (2015). Predicting the Level of Text Standardness in User-generated Content. In *Proceedings of Recent Advances in Natural Language Processing*.

Mihalcea, R. and Nastase, V. (2002). Letter Level Learning for Language Independent Diacritics Restoration. In *Proceedings of the 6th Conference on Natural Language Learning - Volume 20*, COLING-02, pages 1–7, Stroudsburg, PA, USA. Association for Computational Linguistics.

Novák, A. and Siklósi, B. (2015). Automatic Diacritics Restoration for Hungarian. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2286–2291, Lisbon, Portugal, September. Association for Computational Linguistics.

Šantić, N., Šnajder, J., and Bašić, B. D. (2009). Automatic Diacritics Restoration in Croatian Texts. In Hrvoje Stančić, et al., editors, *The Future of Information Sciences, Digital Resources and Knowledge Sharing*.

Simard, M. (1998). Automatic Insertion of Accents in French Text. In *EMNLP*, pages 27–35.

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

Tufiş, D. and Chiţu, A. (1999). Automatic insertion of diacritics in Romanian texts. In *Proceedings of the 5th International Workshop on Computational Lexicography COMPLEX*, pages 185–194.

Yarowsky, D. (1994). Decision lists for lexical ambiguity resolution: Application to accent restoration in Spanish and French. In *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*, pages 88–95. Association for Computational Linguistics.

Yarowsky, D., (1999). *Natural Language Processing Using Very Large Corpora*, chapter A Comparison of Corpus-Based Techniques for Restoring Accents in Spanish and French Text, pages 99–120. Springer Netherlands, Dordrecht.