Letter Level Learning for Language Independent Diacritics Restoration

Rada Mihalcea
Department of Computer Science
University of North Texas
Denton, TX, 76203-1366
rada@cs.unt.edu

Vivi Nastase
School of Information Technology
University of Ottawa
Ottawa, ON, Canada
vnastase@site.uottawa.ca

Abstract

This paper presents a method for diacritics restoration based on learning mechanisms that act at letter level. The method requires no additional tagging tools or resources other than raw text, which makes it independent of the language, and particularly appealing for languages for which there are few resources available. The algorithm was evaluated on four different languages, namely Czech, Hungarian, Polish and Romanian, and an average accuracy of over 98% was observed.

1 Introduction

Diacritics restoration is the problem of inserting diacritics into a text where they are missing. With the continuously increasing amount of texts available on the Web, tools for automatic insertion of diacritics become an essential component in many important applications such as Information Retrieval, Machine Translation, Corpora Acquisition, and others. Spelling correction may have a direct impact on the processing quality in many of these applications. For instance, in the absence of a tool for diacritics recovery, a search for the Romanian word *peşte*(*fish*) retrieves *peste*(*over*) as well, *paturi* can be wrongly translated as *beds*, where the intended meaning was *blankets* (the translation of *pǎturi*), and so forth.

The problem as such is not very difficult, and previous work has demonstrated that a good dictionary can lead to over 90% accuracy in accent restoration for French and Spanish (Yarowsky, 1994), (Simard, 1998), (Galicia-Haro et al., 1999). The method described by Michael Simard in (Simard, 1998) is an improvement over a similar method proposed by El-Bèze (El-Bèze et al., 1994). It relies on Hidden Markov Models and learns from surrounding words for an overall reported accuracy of 99%. Tufiş and Chiţu (Tufiş and Chiţu, 1999) propose a similar approach for diacritics insertion in Romanian texts. Yarowsky (Yarowsky, 1999) gives a comprehensive overview of accent restoration techniques. Most of the algorithms he presents rely on dictionaries and surrounding words in deciding whether to select a form or another for a given ambiguous word. A different approach is proposed by Nagy et. al in (Nagy et al., 1998), where strings extracted from texts are used to derive statistics, with high precision reported on French texts. Their work is similar with the approach proposed in (Angell et al., 1983), where trigram similarity measures are employed for automatic spelling correction.

The majority of studies performed so far in this field have addressed well known and widely spread languages such as French or Spanish, and very few studies have emphasized less popular languages like Polish, Slovene, Turkish or other languages that employ diacritics in their spelling. Table 1 lists the diacritics encountered in European languages with Latin-based alphabets[1]. As seen in the table, a large number of languages face the problem of diacritics restoration.

From the 36 European languages considered, English is the only one that does not have diacritics restoration problems. The few words that use special characters are borrowed from other languages (e.g. *fiancé*, *café*), and these words have no diacritics free correspondent with which they could be confused. It has been observed though that English has a higher semantic ambiguity than other languages. The absence of diacritics may be part of the explanation of this phenomenon[2].

The applicability of word based methods for diacritics restoration is limited when:
(1) Electronic dictionaries are not available, or only limited size dictionaries are made public. Moreover, when the dictionary itself lacks diacritics, methods relying on diacritics restoration from dictionaries become useless.
(2) Tools for morphological and/or syntactic analysis, which are considered to be helpful for the prob-

[1]The table lists only lower case letters. The information in this table was compiled from lists of diacritics in European languages available at http://www.tiro.com/di_intro.html

[2]Studies performed on bilingual parallel corpora have shown that the vocabulary built from an English text is about half the size of the vocabulary build for the same text written in a different language. Senseval competition (Kilgarriff, 2001) has also reported significantly lower precision for English with respect to other languages, in a word sense disambiguation task.

Language	Diacritics	Language	Diacritics
Albanian	ç ë	Italian	à é è í ì ï ó ò ú ù
Basque	ñ ü	Lower Sorbian	ć č ě ł ń ŕ ś š ź ž
Breton	â ê ñ ù ü	Maltese	ċ ġ ħ ż
Catalan	à ç è é í ï l· ò ó ú ü	Norwegian	å æ ø
Czech	á č d' é ě í ň ó ř š t' ů ú ý ž	Polish	a̧ ć ȩ ł ń ó ś ź ż
Danish	å æ ø	Portuguese	â ã ç ê ó ô õ ü
Dutch	á à â ä é è ê ë i í ì î ï ó ò ô ö ú ù û ü	Romanian	â ă î ş ţ
English	none	Sami	á ï č d- ń ņ š t- ž
Estonian	ä č õ ö š ü ž	Serbo-Croatian	ć č d- š ž
Faroese	á æ d- í ó øú ý	Slovak	á ä č d' é í Ĺ ñ ó ô ŕ š t' ú ý ž
Finnish	ä å ö š ž	Slovene	č š ž
French	à â æ ç è é ê ë î ï ô œ ù û ÿ	Spanish	á é í ó ú ü ñ
Gaelic	á é í ó ú	Swedish	ä å ö
German	ä ö ü ß	Turkish	ç ğ i ı ö ş ü
Hungarian	á é í ó ö ő ú ü ű	Upper Sorbian	ć č ě ł ń ó ř š ž
Icelandic	á æ∂ é í ó ö ú ý ♭	Welsh	â ê î ô û ŵ ŷ

Table 1: Diacritics in European languages with Latin based alphabets.

lem of diacritics restoration, are inexistent or are not publicly available.
(3) Size of usable corpora containing diacritics is limited. The size of the corpora available on the Web or in other public forms influences the size of the vocabulary that can be built *ad-hoc* out of these texts. Moreover, Web publishers choose in many cases to avoid diacritics, for reasons of simplicity, uniformity or just the lack of means for diacritics encoding.

We propose in this paper a technique for diacritics restoration based on learning performed at letter level, rather than word level. The strongest advantage of this method is that it provides the means for generalization beyond words.

Specifically, instead of learning generalizations that apply at word level, that would translate to rules such as *"anuncio should change to anunció when it is a verb"*, or *"peste should change to peşte when followed by the noun casă"*, we are interested in finding generalizations at letter level, which can translate to *"s followed by i and preceded by white space should change to ş"*. This latest type of rules are more general and they have higher applicability when only small dictionaries are available, when many unknown words are encountered in the input text, or when there are no usable tools for morphological or syntactic analysis.

It is clear that letters constitute the smallest possible level of granularity in language analysis, and therefore have the highest potential for generalization. Instead of having about 150,000 units that are potential candidates for the algorithm (the approximate size of the general purpose vocabulary of a language), we have more or less 26 characters that will constitute the entry to the disambiguation mechanism.[3]

The method is particularly useful for languages that lack publicly available text processing tools or large electronic dictionaries with diacritics. Well studied and widespread languages such as French and Spanish can benefit as well from this methodology when dealing with unknown words.

The algorithm was evaluated on four different languages, namely Czech, Hungarian, Polish and Romanian, and an average precision of over 98% at letter level was observed. Moreover, this method does not require any other tools or resources other than a corpus of raw text with diacritics. Due to the simplicity of the algorithm, the processing speed is very high, about 20 pages of text per second, measured on an off the shelf PC. This includes both the extraction of context statistics and the learning phase.

2 Experimental setup

The goal of the experiments reported in this paper is to see whether learning at letter level is possible to the end of solving the problem of diacritics reinsertion. Besides providing an additional method for diacritics restoration, the purpose of doing learning at such a low level is to supply a methodology for languages that have only few lexical and semantic resources and for which diacritics restoration via learning at word level is hard to perform.

[3]The actual numbers depend on the language considered. It was shown, for instance, that about 85% of the French words do not have any spelling that includes diacritics, and hence only about 20,000 words are potentially ambiguous. On the other hand, only 7 letters are ambiguous in French.

2.1 Data

As mentioned earlier, the experiments were performed on four different languages: Czech, Hungarian, Polish and Romanian. We have selected languages that are not widely spread, and consequently do not have many publicly available tools or resources. In order to apply the algorithm, the only requirement is a medium size corpus of raw text with diacritics.

The data was collected over the Internet, using mainly newspapers archives or literal texts available electronically. For the four languages, the main sources used to construct the textual corpus are as follows: (1) for Czech, the archive from *Lidovky* newspaper, and literary texts by *Kafka, Hašek* and *Čapek*; (2) for Hungarian, the archive from *Digitális Irodalmi Akadémia*, and a novel by *Petöfi Sándor*; (3) for Polish, the archive of *Wiedza i życie*; (4) for Romanian, the archive made available by the *România Literară* newspaper. Additionally, other texts with diacritics were collected from various Web sites, to the end of building for each language a corpus of at least one million words.

Next, the HTML files collected from the Web were converted into text files. Upper case letters were converted into lower case. After these steps, we were left with a corpus of 1,46 million words for Czech, 1,72 million words for Hungarian, 2,50 million words for Polish, and about 3 million words for Romanian.

2.2 Learning algorithms

We decided to use an instance based learning algorithm for our diacritics restoration task. The reason for this decision is twofold. First, it has been advocated that forgetting exceptions is harmful in Natural Language applications, and instance based learners are known for their property of taking into consideration every single training example when making a classification decision (Daelemans et al., 1999). Secondly, this type of algorithms are efficient in terms of training and testing time. For our experiments, we used the TiMBL (Daelemans et al., 2001) implementation[4].

Since we work at the low level of letters, the target attribute to be learned is constituted by the ambiguous letters. It can be therefore any of the ambiguous letters listed in Table 1. For the four languages considered in these experiments, the sets of ambiguous letters are shown in Table 2. Upper case diacritics are not considered, since they have been previously converted to lower case.

[4]Similar experiments were performed with a decision tree classifier, namely C4.5 (Quinlan, 1993). The results obtained were similar with those obtained with the instance based learner, at significantly higher running times.

2.3 Features

The features used in any learning algorithm have tremendous influence over the final accuracy. We do not make use of part of speech taggers or any other morphological or syntactic analyzers. Furthermore, we do not want to rely on surrounding words, since the data we have is limited, and therefore learning at word level would result in many cases of unknown words.

We have therefore decided for very simple features, for the extraction of which no particular processing is required. We only look at surrounding letters, with a special notation assigned to white spaces, commas, dots and colons (these characters may affect the learning process, since they are considered special characters by TiMBL or C4.5). This set of features performs surprisingly well in terms of accuracy.

For each ambiguous pair of letters, we scan the text and generate all possible examples encountered in the corpus. The attributes in an example are formed by N letters to the left and right of the ambiguous letter, and the target attribute is the ambiguous letter itself. Samples of feature vectors are presented below.

l , i , n , SP , (, u , b , SP , i , n , s.
e , CO , SP , r , o , - , g , a , r , d , ş.
g , a , r , d , i , t , u , l , CO , SP , s.
e , SP , o , r , a , DO , SP , t , o , t , ş.

These are the examples that were fed to the learning algorithm for the *s* - *ş* ambiguous pair from Romanian. CO, DO and SP are the replacement codes we use to denote comma, dot or white characters. Note that the surrounding letters with diacritics are converted into their diacriticless equivalent.

3 Results

From the total set of examples extracted for each ambiguous letter set, 50,000 examples are set aside for testing, and the rest is used to train the learner. Table 2 shows the results obtained for the four languages. For each language, the table lists the sets of ambiguous letters, the number of examples extracted from the corpus for each such ambiguous set, a simple baseline computed as the precision achieved when the most frequent element of the ambiguous set is selected by default, the F-measures (Van Rijsbergen, 1979) computed for each individual letter, and finally the precision achieved with the instance based learner applied at letter level (*IBLLL*). The average accuracy determined for all four languages is 98.17%.

The average accuracy per language is influenced by the size of the corpus. The corpora collected for Czech and Hungarian contain about 1.4-1.7 million

Ambiguous set	Total examples	Baseline	Individual F-measures	Overall IBLLL
		Czech		
a á	649,886	75.01%	97.97%/93.92%	96.96%
c č	217,570	72.21%	98.00%/94.60%	97.08%
d d'	271,070	99.05%	99.93%/84.93%	99.86%
e é ě	768,051	74.59%	98.74%/91.85%/91.81%	97.02%
i í	504,298	60.43%	96.95%/95.27%	96.29%
n ň	439,552	98.97%	99.88%/89.06%	99.71%
o ó	566,521	99.08%	99.93%/63.44%	99.86%
r ř	319,352	65.55%	98.53%/95.36%	97.60%
s š	380,805	84.44%	99.34%/96.42%	98.88%
t t'	387,214	99.05%	99.92%/83.90%	99.85%
u ů ú	264,408	80.89%	95.83%/77.15%/95.55%	93.51%
y ý	191,317	65.55%	96.23%/92.82%	95.06%
z ž	219,082	66.49%	99.02%/98.06%	98.70%
Average		80.44%		97.83%
		Hungarian		
a á	1,198,294	73.51%	97.90%/94.19%	96.91%
e é	1,306,944	76.34%	97.65%/92.36%	96.40%
i í	647,137	89.14%	99.71%/97.65%	99.49%
o ó ö ő	678,012	71.15%	97.33%/90.17%/95.27%	96.10%
u ú ü ű	207,753	56.00%	97.79%/95.47%/97.33%	97.31%
Average		75.32%		97.04%
		Polish		
a ą	1,387,019	88.83%	98.35%/86.68%	97.07%
c ć	657,669	91.50%	99.68%/96.57%	99.42%
e ę	1,305,584	89.23%	99.54%/96.18%	98.47%
l ł	506,041	59.29%	98.99%/98.53%	98.80%
n ń	878,824	96.75%	99.92%/97.81%	99.85%
o ó	1,230,389	88.67%	99.93%/99.46%	99.87%
s ś	688,677	88.67%	99.90%/99.26%	99.83%
z ź ż	896,909	86.26%	99.84%/99.01%	99.73%
Average		87.18%		99.02%
		Romanian		
a ă â	2,161,556	74.70%	97.17%/91.31%	96.14%
i î	2,055,147	88.20%	99.18%/98.80%	99.69%
s ş	866,964	76.53%	99.23%/98.23%	99.07%
t ţ	1,157,458	85.81%	99.30%/96.43%	98.75%
Average		80.92%		98.30%
Average over all languages		81.88%		98.17%

Table 2: Results obtained in solving diacritics ambiguity, in Czech, Hungarian, Polish and Romanian

words, and the precision achieved on these two languages is lower than for Polish and Romanian, which have corpora of 2.5-3 million words. We expect therefore to see an increase in these figures when increasing the size of training data. Experiments performed on one of the four languages have confirmed this hypothesis.

Interestingly, the number of diacritics in a language do not necessarily influence the precision obtained in diacritics restoration. For instance, the precision achieved on Hungarian, which has a total of five ambiguous letter sets, is smaller than the precision achieved on Czech, which has an impressive number of diacritics (thirteen ambiguous sets). Yet, the corpus used for Hungarian (1.7 million words) is slightly larger than the corpus used for Czech (1.4 million words).

As a general observation about the results obtained, the learning curve is very steep in the beginning, precision growing fast with every bit of new data available. After a certain point the learning slows down, for each percent in precision improvement a large amount of data will be necessary.

In terms of window size, the best accuracy was observed for ten surrounding letters (i.e. $N = 5$). We have therefore studied in more detail this case, and determined learning rates associated with several sets of ambiguous letters. We selected one of the four languages for further analyses, including correlation measurements between corpus size and precision, and discussion of results. The language of choice was Romanian, since this is the language that has the largest corpus among the four study languages.

Table 3 shows the results obtained for $N = 5$. Tests were performed with training corpora of various sizes, ranging from 2,000,000 examples to as few as 10 examples, to the end of finding the learning rate and the minimum size of a corpus required for a satisfactory precision. All experiments are performed with a test set size of 50,000 examples. The table shows also the baseline, defined as the precision obtained when the most frequent letter in the ambiguous set is selected by default.

The results shown in Table 3 are plotted in Figure 1. It is interesting to observe that the most important part of the learning process is achieved with the first 10,000 examples. We have measured that about 100,000-250,000 running characters (approx. 25-60 pages of text) are needed to generate 10,000 diacritics examples, a relatively small corpus. From there on, a significant number of examples is required for every single percent of improvement in accuracy. We also show in bold the first precision figure that exceeds the baseline, as an indicative of the smallest size of training set where a form of learning is observed. Notice that as few as 1,000 examples are enough to perform *some* learning.

According to (Banko and Brill, 2001) there are algorithms that do not scale very well to the size of the corpora. From the comparative results discussed above, we observe that the letter level learning algorithm that we propose is scalable to the size of the corpus, and most importantly, it fares extremely well for very little input data.

Using the entire set of examples extracted from the corpus, the disambiguation of the *i-î* pair is almost 100% correct. For this diacritic letter, we now have one instance wrong out of 300 instances, whereas the baseline implies one instance wrong for every eight instances, therefore a significant improvement. The worst precision is achieved in the case of *a-ă* pair. From a simple error analysis, it turns out that the main reason for this is the fact that many Romanian nouns have their base form ending in *ă*, whereas their articulated form ends in *a*. For instance, *masă* and *masa* are two forms, one articulated and one not, for the same noun *table*. The learner is therefore tricked by many identical usages for these letters. A simple solution for this would be to avoid in the learning process those examples that contain an *a* or *ă* letter at the end of a word. The results obtained under this simplifying assumption are reported in Table 3 under the heading *a-ă(2)* As shown in the table, more than four percents are gained in precision with this simple condition.

We have also employed C4.5 on the same training data, but no improvements were observed with respect to the results from Table 2. The disadvantage of using C4.5 for this task is that the learning phase is slower than with the TiMBL implementation. On the other hand, C4.5 has the capability of generating expressive rules. *"L_1=e and L_2=space then s"*(99.5%), *"L_1=t and L_2=space then s"* (98.7%), *"L_{-4} = p and L_{-1}=v and L_1=t L_2=e then ş"*(95.5%), are examples of such rules, where L_i denotes a surrounding letter at the relative position i with respect to the ambiguous letter. Notice that these rules do not say anything about whether or not the letters belong to one single word. The learning algorithm simply relies on letters, regardless of the word they belong to. Consequently, pseudo-homographs words (as in *peste* and *peşte* - see Section 1) are equally addressed by this method, as the algorithm has the capability of going across words.

3.1 Different window sizes

We have experimented various window sizes to determine the size of the context that would best model our problem. We considered window sizes of two, six, ten, fourteen and eighteen surrounding letters (i.e. $N = 1, 3, 5, 7, 9$). Comparative results, again for the four Romanian ambiguous letter sets, are reported in Table 4.

When no context is available, window sizes of N=3 can be used without too much loss in precision. Nevertheless, as stated earlier, the best accuracy is attained for a window of ten surrounding letters (N=5).

4 Comparison with related work

These results are best compared with the work reported by Tufiş and Chiţu (Tufiş and Chiţu, 1999), who employed one of the languages used in our experiments, namely Romanian. According to Tufiş and Chiţu, the task of diacritics recovery in Romanian is harder than with other languages, since Romanian makes more intensive use of diacritics. As reported in their experiments, only about 60% of the Romanian words are diacritics free, compared to the studies reported in (Simard, 1998) which show that about 85% of the French words are spelled with no accents.

The approach presented by Tufiş and Chiţu uses dictionaries, a tokenizer and part of speech tagger,

	Ambiguous pair				
	a-ă	*a-ă*(2)	*i-î*	*s-ş*	*t-ţ*
Data set size	2,161,556	1,369,517	2,055,147	866,964	1,157,458
Baseline	74.70%	85.90%	88.20%	76.53%	85.81%
Training size	Precision obtained with a test set of 50,000 examples				
2,000,000	96.14%	-	99.69%	-	-
1,000,000	95.10%	99.14%	99.58%	-	98.75%
750,000	94.83%	98.97%	99.53%	99.07%	98.63%
500,000	94.57%	98.79%	99.46%	98.86%	98.40%
250,000	94.00%	98.37%	99.28%	98.87%	98.26%
100,000	93.03%	97.56%	98.96%	98.54%	97.81%
50,000	92.10%	96.86%	98.57%	98.13%	97.40%
25,000	90.99%	95.75%	98.11%	97.58%	96.92%
10,000	88.99%	93.75%	97.31%	96.53%	96.20%
5,000	87.56%	92.76%	96.65%	95.61%	95.10%
4,000	86.91%	91.86%	96.49%	94.99%	94.53%
3,000	86.39%	90.99%	96.19%	94.18%	94.30%
2,000	85.81%	89.93%	95.49%	93.47%	93.56%
1,000	83.49%	**88.36%**	93.78%	92.31%	91.85%
500	80.61%	85.66%	93.07%	90.75%	89.74%
250	77.89%	83.17%	92.75%	87.41%	**87.23%**
100	**74.80%**	84.04%	**91.41%**	82.13%	84.46%
50	72.79%	82.73%	88.05%	86.53%	77.54%
25	72.45%	81.34%	88.15%	**78.26%**	78.52%
10	73.38%	85.90%	88.20%	75.88%	85.81%

Table 3: Results obtained in solving diacritics ambiguity in Romanian, using a window size of ten surrounding letters

Ambiguous pair	Window size				
	N=1	N=3	N=5	N=7	N=9
a-ă	85.63%	95.79%	96.14%	96.10%	96.10%
i-î	94.18%	99.13%	99.69%	99.68%	99.43%
s-ş	88.09%	99.06%	99.07%	99.02%	99.00%
t-ţ	89.45%	98.57%	98.75%	98.67%	98.25%

Table 4: Comparative results for various window sizes (Romanian)

and learning is performed at word level, for an overall performance of 97.4%. We cannot directly compare our results, as both methods and evaluations are fundamentally different. The average precision of 98.30% obtained with our algorithm on the Romanian language (respectively 99.23% when *a-ă* at the end of a word is not considered during learning) is measured at letter level, whereas the accuracy they report is determined at word level.

The algorithm presented in this paper overcomes previous approaches in that very high precisions and processing speeds are obtained without any preprocessing tools or dictionaries being required. This algorithm is therefore applicable to any language, with the only requirement being a medium size corpus of texts with diacritics.

5 Conclusion

The experiments presented show that the automation of diacritics insertion based on learning at letter level performs extremely well. The fact that the resource requirements are so modest compared to methods that use learning at word level - a medium size corpus with diacritics versus part-of-speech taggers, dictionaries, etc. - make it very appropriate and easy to use especially for languages for which such resources are not available. The method can also be very useful in cases where word-based methods encounter the problem of unknown words.

Experiments were performed using four languages: Czech, Hugarian, Polish and Romanian. Raw texts are fed to the learning mechanism, and an average accuracy of over 98% at letter level was

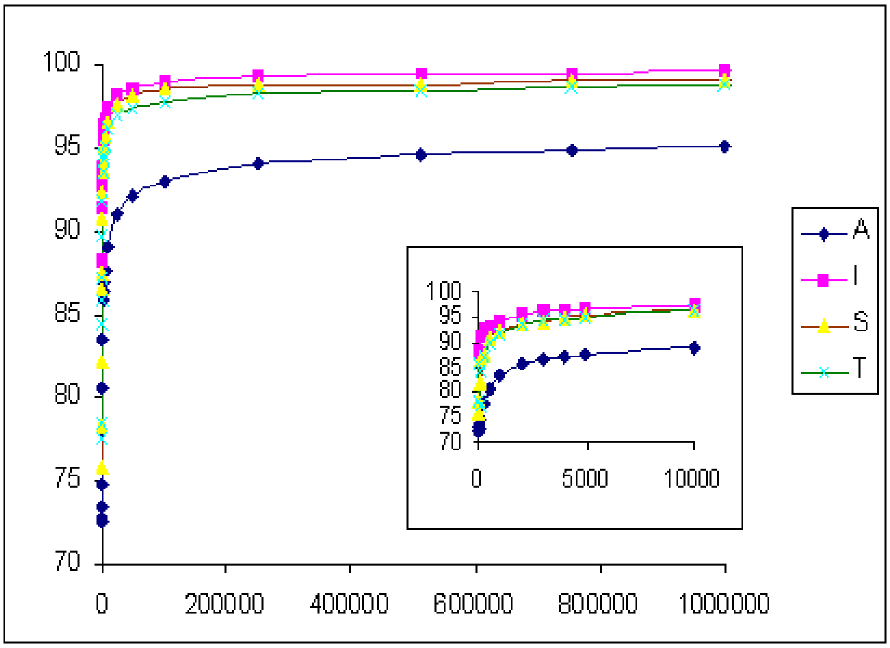

Figure 1: Learning rates for the four ambiguous diacritics in Romanian. The chart in the middle represents a zoom of the 0-10,000 range area.

observed. Moreover, because of the simplicity of the algorithm, the learning is performed very fast, at a speed of about 20 pages of text per second on an off the shelf PC.

Acknowledgments

The authors would like to thank Jan Žižka from Masaryk University, Brno, Czeck Republic, and Dariusz Kogut from the Warsaw University of Technology, Warsaw, Poland, for their help in identifying sources of electronic collections of texts with diacritics. They would also like to thank the anonymous reviewers for their helpful comments and suggestions.

References

R.C. Angell, G.E. Freund, and P. Willett. 1983. Automatic spelling correction using a trigram similarity measure. *Information Processing and Management*, 19(4):255–261.

M. Banko and E. Brill. 2001. Scaling to very very large corpora for natural language disambiguation. In *Proceedings of the 39th Annual Meeting of the Association for Computational Lingusitics (ACL-2001)*, Toulouse, France, July.

W. Daelemans, A. van den Bosch, and J. Zavrel. 1999. Forgetting exceptions is harmful in language learning. *Machine Learning*, 34(1-3):11–34.

W. Daelemans, J. Zavrel, K. van der Sloot, and A. van den Bosch. 2001. Timbl: Tilburg memory based learner, version 4.0, reference guide. Technical report, University of Antwerp.

M. El-Bèze, B. Mérialdo, B. Rozeron, and A. Derouault. 1994. Accentuation automatique des textes par des méthodes probabilistes. *Techniques et sciences informatique*, 16(6):797–815.

S.N. Galicia-Haro, I.A. Bolshakov, and A.F. Gelbukh. 1999. A simple Spanish part of speech tagger for detection and correction of accentuation error. In *Proceedings of the Second International Workshop on Text, Speech and Dialogue*, pages 219–222, Plzen, Czech Republic, September.

A. Kilgarriff, editor. 2001. *Proceedings of SENSEVAL-2, Association for Computational Linguistics Workshop*, Toulouse, France.

G. Nagy, Nagy N., and M. Sabourin. 1998. Signes diacritiques: perdus et retrouvés. In *Actes du 1er Collque International Francophone sur l'Écrit et le Document CIFED '98*, pages 404–412, Queébec, Canada.

J. Quinlan. 1993. *C4.5: programs for machine learning*. Morgan Kaufman.

M. Simard. 1998. Automatic insertion of accents in French text. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing EMNLP-3*, Granada.

D. Tufiş and A. Chiţu. 1999. Automatic diacritics insertion in Romanian texts. In *Proceedings of the International Conference on Computational Lexicography COMPLEX'99*, Pecs, Hungary, June.

C.J. Van Rijsbergen. 1979. *Information Retrieval*. London: Butterworths. available on-line at http://www.dcs.gla.ac.uk/ Keith/Preface.html.

D. Yarowsky. 1994. Decision lists for lexical ambiguity resolution: Application to accent restoration in Spanish and French. In *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*, pages 88–95, Las Cruces, NM.

D. Yarowsky, 1999. *Corpus-based techniques for Restoring accents in Spanish and French Text*, pages 99–120. Kluwer Academics Publisher.