A Lemmatization Web Service Based on Machine Learning Techniques

Joël Plisson, Dunja Mladenić, Nada Lavrač,
Tomaž Erjavec

Department of Knowledge Technologies
Jožef Stefan Institute,
Jamova 39, 1000 Ljubljana, Slovenia
joel.plisson@ijs.si

Abstract

Lemmatization is the process of finding the normalized form of words from surface word-forms as they appear in the running text. It is a useful pre-processing step for any number of language engineering tasks, esp. important for languages with rich inflection morphology. This paper presents two approaches to automated word lemmatization, which both use machine learning techniques to learn particular language models from pre-annotated data. One approach is based on Ripple Down Rules and the other on First-Order Decision Lists as learned by the CLog system. We have tested the two approaches on the Slovene language and set-up a generally accessible Web service for lemmatization using the generated models.

1. Introduction

Lemmatization is the process of finding normalized forms of words, called lemmas – these can usefully also be thought of as the headwords in a dictionary. It is an important preprocessing step for many applications dealing with text, including information retrieval, text mining, and applications of linguistics and natural language processing. It also provides a productive way to generate generic keywords for search engines or labels for concept maps.

Lemmatization is often replaced by a simpler, related problem of word stemming. While lemmatization aims at transforming a word to get its canonical form (say, singular nominative for nouns, infinitive for verbs), stemming produces the stem of the word. For instance, the word-forms COMPUTES, COMPUTING, COMPUTED should be most likely stemmed to COMPUT, while their normalized form (lemma) is the infinitive of the verb: COMPUTE. Of course, in an inflectionally poor language such as English, the lemma and stem are quite often identical; this, however, does not hold for inflectionally rich languages, such as Slovene and other Slavic languages.

We have developed a Web service providing access to two models for lemmatization: one developed by using Ripple Down Rule (RDR) learning and the other using a system for learning first-order decision lists, named CLog (Manandhar et al., 1998). Both systems use a tokenizer to split the input text into words (and punctuation) and these words are then the input to the actual lemmatization modules.

This paper describes the developed lemmatization Web service and some testing results on the Slovene language. Section 2. describes the problem of word lemmatization. In Section 3. the two machine learning approaches used for the generation of lemmatization rules are described. Section 4. presents the results of experiments using the MULTEXT-East lexicon (Erjavec, 2004) and the IJS-ELAN (Erjavec, 1999) corpus. Finally, we describe the lemmatization Web service in Section 5.

2. The lemmatization problem

In this paper, lemmatization is viewed as a process that takes a word-form taken from text and replaces its grammatical ending (which can, in cases of stem alternations, also encompass the stem) by the the grammatical ending of the lemma form of the word. Simply put, we define lemmatization in the same way as defined in (Mladenić, 2002), as a replacement of a suffix by another suffix.

The grammatical ending usually depends on the word ending; two words with different endings will not have the same grammatical ending, even if they are used in the same grammatical form. For example, the words PROPERTY and TRAIN will not receive the same grammatical ending even if they are both used in their plural form. PROPERTY will lose Y and receive a suffix IES to become PROPERTIES, while TRAIN will receive the suffix S and change to TRAINS. This observation also holds when trying to recover the normalized form: most frequently, the grammatical ending contains the information about the ending of the normalized form. The suffix IES of the word PROPERTIES indicates that the normalized form ends with a Y. In the proposed approach to word lemmatization, the idea is to create rules to recover the initial suffix from the grammatical ending.

A lot of work has been carried out in word lemmatization, particularly for English. Recent work (Mladenić, 2002; Džeroski and Erjavec, 2004) resulted in lemmatization systems for Slovene: in the first approach if-then rules were induced by the ATRIS rule induction algorithm (Mladenić, 1993; Mladenić, 2002), and in the second approach (Džeroski and Erjavec, 2004) a system for learning first-order decision lists CLog (Manandhar et al. 1998) was used. The main difficulty of Slovene word lemmatization is that Slovene is a highly inflected natural language, having up to 30 different word forms for the same normalized word. Furthermore, these word forms stand in complex relations of syncretism, ending and stem alternations, paradigm selection, defective paradigms, etc.

3. Machine learning approaches used for word lemmatization

A machine learning approach to lemmatization takes as input a set of pairs `(Word, LemmatizedWord)` and results in a set of induced lemmatization rules. As proposed in (Mladenić, 2002), each word in a training set is labeled by a class label, that represents the transformation that should be applied to get the normalized form of the word. To determine the class, the word stem should be found first. Notice that this is not necessary the same stem as obtained by applying a stemmer to the same word. Here, by a stem we refer to the part that the two words (the word and its normalized form) have in common. The words PROPERTY and PROPERTIES have both the stem PROPERT in common. Or in Slovene, the words BRESKEV and BRESKVAH have BRESK in common; in lemmatization we should remove the suffix VAH from BRESKVAH and add the suffix EV to get the normalized form BRESKEV.

In our RDR approach to learning Ripple Down Rules (RDR), (Plisson et al., 2004) a training set for Slovene word lemmatization consists of words, described by their suffixes (the attributes of the learning program) and the transformation (the class) to be applied to the word suffix in order to get the normalized form. As the words have different lengths, they are described by a variable number of attributes $A_i \in [A_1, \ldots, A_n]$ which correspond to word suffixes of *i* letters; the number of attributes depends on the word length *n*. The transformation (the class) in the form S1toS2 corresponds to a mapping S1 $\rightarrow$ S2, where S1 is the suffix of a word and S2 is the suffix of the normalized word; e.g., class label VAHtoEV is assigned to word BRESKVAH, while class label _to_ means that the word does not change.

A more knowledge-intense approach to learning lemmatization was taken in the second system already described in (Džeroski and Erjavec, 2004). Here, CLog learns first-order decision lists, but the input is not just pairs of word-form and lemma, but triples containing also the morpho-syntactic description (MSD) of the word-form. For example, the word-form HODIMO (of the lemma HODITI) can be assigned the MSD `Vmmp1p` - this is, via the MULTEXT-East specifications, equivalent to `PoS: Verb, Type: main, VForm: imperative, Tense: present, Person: first, Number: plural`.

The actual CLog lemmatizer then learns the rules separately for each MSD. Assuming perfect MSDs this leads to a significantly improved performance but the price we pay is in having to apply a part-of-speech tagger to the input text first. We used TnT (Brants, 2000) trained on a medium sized manually annotated Slovene corpus, comprising 100000 words, and its performance improved with a backup lexicon and various heuristics. Still, the tagger makes a significant number of mistakes which then often also result in the actual lemmatization producing wrong results.

3.1. Learning Ripple Down Rules

In comparison with the standard if-then classification rules, the Ripple Down Rules (RDR) representation (Compton and Jansen, 1990; Srinivasan et al., 1991) resembles decision lists (Rivest, 1987) of the form if-then-else: new RDR rules are added by creating **except** or **else** branches to the existing rules. Take a simple Ripple Down Rule for verb lemmatization:

if *2LastCh = ED* ***then*** *transform = EDto_*
except if *3LastCh = IED* ***then*** *transform = IEDtoY*

The rule states that one should remove the suffix ED and add an empty suffix to get the normalized form unless there is an I before ED then we should remove the suffix IED and add the suffix Y. This rule would, for instance, correctly generate the lemmatized form of words WALKED (WALK) and CLASSIFIED (CLASSIFY).

A major additional feature of RDR rules, compared to decision lists, is that exceptions that have created an **except** or **else** branch are added to the branch with a **because** keyword, in order to explain the reason for the creation of the new rule. This feature of Ripple Down Rules turns out to be extremely helpful for better understanding the complex rules. A rule, induced from training examples BRESKEV, POSTAVITEV and ZAHTEV (with normalized forms BRESKEV, POSTAVITEV and ZAHTEVA), is as follows:

if *V* ***then*** *_to_* ***because of*** *[BRESKEV, POSTAVITEV]*
    except if *HTEV* ***then*** *_toA* ***because of*** *[ZAHTEV]*

The result of rule induction is an RDR rule set that can be used to lemmatize words. We apply the rules on each word to be lemmatized and if a rule fires, its conclusion (class value) is the transformation to apply to get the normalized form (lemma) of the word. This means that for using the rules on free text, one must first convert it with a tokenizer into a list of words and present them one by one to the induced RDR rule set.

3.2. Learning decision lists

The formulation of the lemmatization problem considered in this section is as follows. A logic program has to be learned defining the relation `lemmatize(Word, LemmatizedWord)`, where `Word` is an orthographic representation of the word and `LemmatizedWord` is an orthographic representation of its normalized form. `Word` is the input and `LemmatizedWord` the output argument. Given are examples of input/output pairs, such as `lemmatize([b,a,r,k,e,d], [b,a,r,k])` and `lemmatize([w,o,r,k,i,n,g], [w,o,r,k])`. The program for the relation lemmatize uses the predicate mate/6 as background knowledge: this predicate establishes a relation between the word-form and lemma, and their two prefixes and suffixes, e.g. the predicate to derive the lemma LEP from NAJLEPŠI would be mate(A,B,[n,a,j],[],[sx,i],[]).

As a specific example, consider the set of rules induced by CLlog for analysing the genitive singular of Slovene common feminine nouns. The training set for this concept contained 2,646 examples, from which CLog learned 22 rules of analysis. Nine of these were lexical exceptions, and are not interesting in the context of unknown word lemmatisation. We list the generalisations in Figure 1.

```
ncfsg(A,B):-mate(A,B,[],[],[n,o,v,e],[n,o,v,a]),!.
ncfsg(A,B):-mate(A,B,[],[],[e,v,e],[e,v,a]),!.
ncfsg(A,B):-mate(A,B,[],[],[a,v,e],[a,v,a]),!.
ncfsg(A,B):-mate(A,B,[],[],[r,v,e],[r,v,a]),!.
ncfsg(A,B):-mate(A,B,[],[],[i,v,e],[i,v,a]),!.
ncfsg(A,B):-mate(A,B,[],[],[e,s,n,i],[e,s,e,n]),!.
ncfsg(A,B):-mate(A,B,[],[],[i,s,l,i],[i,s,e,l]),!.
ncfsg(A,B):-mate(A,B,[],[],[v,e],[e,v]),!.
ncfsg(A,B):-mate(A,B,[],[],[z,n,i],[z,e,n]),!.
ncfsg(A,B):-mate(A,B,[],[],[i],[]),!.
ncfsg(A,B):-mate(A,B,[],[],[e],[a]),!.
```

Figure 1: A first-order decision list.

From the bottom up, the first rule in Figure 1 describes the analysis for nouns of the canonical first feminine declension, where the genitive singular ending E is replaced by A to obtain the lemma, e.g., MIZE → MIZA. The second rule deals with the canonical second feminine declension where I is removed from the genitive to obtain the lemma, e.g., PERUTI → PERUT. The third rule attempts to cover nouns of the second declension that exhibit a common morpho-phonological alteration in Slovene, the schwa elision. Namely, if a schwa (weak -E-) appears in the last syllable of the word when it has the null ending, this schwa is dropped with non-null endings: BOLEZNI → BOLEZ$\underline{\text{E}}$N. The fourth rule models a similar case with schwa elision, coupled with an ending alternation, which affects only nouns ending in EV, e.g., BUKVE → BUKEV. The fifth and sixth rule again model schwa elision, but applied to second declension nouns MISLI → MISEL; DLESNI → DLESEN), The last five rules all cover cases of first declension nouns which would otherwise be incorrectly subsumed by the fourth VEtoEV rule, e.g., NJIVE → NJIVA.

As can be seen, the rules exhibit explanatory adequacy, as they can be easily linked to linguistic explanations.

4. Experiments using the MULTEXT-East lexicon and the IJS-ELAN corpus

Recently, a revised version of the MULTEXT-East lexicon (Erjavec, 2004) has become available. This version of the MULTEXT-East lexicon contains 557970 different word forms and is currently the largest available resource for Slovene.

The first experiment using RDR was performed using the entire MULTEXT-East lexicon (557970 words). The achieved 93.2%(± 0.1) accuracy, evaluated using 5-fold cross-validation, represent the best results compared to previous work (Mladenić, 2002; Džeroski and Erjavec, 2004).

Another data source, the IJS-ELAN (Erjavec, 1999) corpus, was also used in the experiments. IJS-ELAN is composed of fifteen recent terminology-rich texts and their translations; it contains one million words, about half in Slovene and half in English. For our experiments, we used a part of the lemmatized Slovene corpus, as in (Džeroski and Erjavec, 2004), containing only nouns, verbs and adjectives.

In the second experiment, we trained the RDR algorithm and the CLog algorithm on the MULTEXT-East lexicon and tested induced RDR rules on a part of the IJS-ELAN corpus consisting of 53035 lemmatized nouns, verbs and adjectives. In addition, we have also performed testing on its subset containing 10315 words obtained by removing words present in the IJS-ELAN corpus and also in the MULTEXT-East lexicon thus making it harder to achieve a good accuracy. CLog rules were induced from the previously available MULTEXT-East lexicon with 542029 words and tested on a subset of the IJS-ELAN corpus with 763 nouns, verbs and adjectives that were not part of the training set of words.

Tested on the list of nouns, verbs and adjectives of the IJS-ELAN corpus containing both known and unknown words, the RDR system achieved 90.0% classification accuracy, while testing only on the nouns, verbs and adjectives from the corpus that were not present in the training set resulted in RDR rules reaching 87.6% accuracy. The RDR accuracy is lower than the 92.0% accuracy achieved by CLog and reported in (Džeroski and Erjavec, 2004). The results are shown in Table 1.

MULTEXT-East/IJS-ELAN	RDR-Set1	RDR-Set2	CLog-Set3
Accuracy	87.6	90.0	92.0
# rules	42186	42186	22563

Table 1: Classification accuracy of RDR and CLog (in %). RDR were induced from the entire MULTEXT-East lexicon with 557970 training examples, and tested on subsets of the IJS-ELAN corpus: Set1 with 10315 unknown words and Set2 with 53035 words, respectively. CLog rules were induced from the MULTEXT-East lexicon with 542029 training examples and tested on a subset of the IJS-ELAN corpus with 763 unknown words.

5. The lemmatization Web service

We have implemented the described two approaches to lemmatization and we have automatically generated the lemmatization models by having trained each of them on a large lexicon. Then, we have set a Web service for lemmatization that uses the generated models.

Our Web service is used as follows. The text to be lemmatized should be copied into the input window, the system is run (performing classification using the underlying lemmatization model) and the lemmatized text is provided in the output window. The user can choose between using the RDR or the CLog model. If RDR is selected, then the text to lemmatize should be provided in the Slovenian language. If CLog is selected, then the user can choose between the following natural languages: Slovene, English, Czech, Estonian, Hungarian, Romanian. The provided input text should then be written in the selected language.

To be more precise, the following procedure should be used when using the developed lemmatization Web service:

1. Choose the model of lemmatization you wish to use.
2. Set the language present in the text you want to lemmatize.
3. Choose the output style of the lemmatized text.
4. Enter your text in the Textbox labeled ”Text to lemmatize”.
5. Press the ”Lemmatize” button.
6. Wait for the result to be displayed in the Textbox labeled ”Output”.

We currently support several formats of the output. For both RDR and CLog, the output can be a list of lemmas (the lemmas of the words are printed out in tabular form, one per line). In addition, for RDR two more output formats are supported:

- Text and lemmas (the lemmas of the words are inserted in the text within a tag <lemma></lemma> on the right of the words).
- Lemmatized text (the words are replaced by their lemmas).

For CLog the output can also be TEI-like XML (the text is printed out with respect to the TEI P4 recommendations).

Notice that the user can enter accents and special characters in the input text (as the text entered in the textbox will be encoded using the UTF-8 standard).

6. Conclusions

Notice that the RDR system works much faster that CLog as it does not use additional grammatical information which requires the use of a tagger as is the case in the model using CLog (Džeroski and Erjavec, 2004). Consequently, the RDR system is also easier to implement for new languages as it does not require a manually disambiguated corpus (to train the tagger) but only a lexicon. Both systems have advantages and disadvantages. One is more accurate but slow (the model developed with CLog), while the other is less accurate but faster (the model using RDR). The choice of which of the two systems to use depends on the user requirements. For example, for a search engine we would probably prefer the faster RDR system while for linguistic tasks it would be preferable to use the CLog model which has a better classification accuracy.

The two experiments using the RDR learner and the CLog learner were not performed using exactly the same data: in the CLog experiments (Džeroski and Erjavec, 2004) the previous version of MULTEXT-East and a smaller subset of IJS-ELAN were used. The experiments have confirmed that the proposed RDR and CLog approaches are appropriate for word lemmatization. The RDR and the CLog lemmatization models can be used through the Web service at **http://nl2.ijs.si/analyze/**.

7. Acknowledgements

This work was supported by the Slovenian Ministry of Higher Education, Science and Technology programme Knowledge Technologies, and the IST 6FP Programme of the European Community projects ALVIS Superpeer Semantic Search Engine (IST-1-002068-STP), SEKT Semantically Enabled Knowledge Technologies (IST-1-506826-IP) and PASCAL Network of Excellence (IST-2002-506778). This publication only reflects the authors' views.

8. References

Brants, T., 2000. TnT - A Statistical Part-of-Speech Tagger. *Proceedings of the Sixth Applied Natural Language Processing Conference ANLP-2000*:224–231.

Compton, P. J. and R. Jansen, 1990. A philosophical basis for knowledge acquisition. *Knowledge Acquisition*, 2:241–257.

Džeroski, S. and T. Erjavec, 2004. Machine learning of morphosyntactic structure: Lemmatising unknown slovene words. *Applied Artificial Intelligence*, 1:17–40.

Erjavec, T., 1999. The elan slovene-english aligned corpus. *Proceedings of the Machine Translation Summit VII*:349–357.

Erjavec, T., 2004. Multext-east version 3: Multilingual morphosyntactic specifications, lexicons and corpora. *Proceedings of the 4th Intl. Conf. on Language Resources and Evaluation, LREC'04, ELRA*.

Manandhar, S., S. Deroski, and T. Erjavec, 1998. Learning multilingual morphology with clog. *Proceedings of Inductive Logic Programming: 8th International Workshop (ILP-98), Number 1446 in Lecture Notes in Artificial Intelligence, ed. D. Page*:135–144.

Mladenić, D., 1993. Combinatorial optimization in inductive concept learning. *Proceedings of ICML*:205–211.

Mladenić, D., 2002. Learning word normalization using word suffix and context from unlabeled data. *Proceedings of ICML*, 1(8):427–434.

Plisson, J., N. Lavrač, and D. Mladenić, 2004. A rule based approach to word lemmatization. *Proceedings of IS04*.

Rivest, R. L., 1987. Learning decision lists. *Machine Learning*:229–246.

Srinivasan, A., P. Compton, R. Malor, G. Edwards, C. Sammut, and L. Lazarus, 1991. Knowledge acquisition in context for a complex domain. *Proceedings of the European Knowledge Acquisition Workshop, Aberdeen*.