An Extensible Multilingual Open Source Lemmatizer

Ahmet Aker[a,b] and **Johann Petrak**[a] and **Firas Sabbah**[b]
Department of Computer Science, University of Sheffield[a]
Department of Information Engineering, University of Duisburg-Essen[b]
a.aker@is.inf.uni-due.de, johann.petrak@sheffield.ac.uk
firas.sabbah@stud.uni-due.de

Abstract

We present GATE DictLemmatizer, a multilingual open source lemmatizer for the GATE NLP framework that currently supports English, German, Italian, French, Dutch, and Spanish, and is easily extensible to other languages. The software is freely available under the LGPL license. The lemmatization is based on the Helsinki Finite-State Transducer Technology (HFST) and lemma dictionaries automatically created from Wiktionary. We evaluate the performance of the lemmatizers against TreeTagger, which is only freely available for research purposes. Our evaluation shows that DictLemmatizer achieves similar or even better results than TreeTagger for languages where there is support from HFST. The performance drops when there is no support from HFST and the entire lemmatization process is based on lemma dictionaries. However, the results are still satisfactory given the fact that DictLemmatizer is open-source and can be easily extended to other languages. The software for extending the lemmatizer by creating word lists from Wiktionary dictionaries is also freely available as open-source software.

1 Introduction

The process of lemmatization is an important part of many computational linguistics applications such as Information Retrieval (IR) and Natural Language Processing (NLP). In lemmatization, inflected forms of a lexeme are mapped to a canonical form that is referred to as the lemma. The task of finding the correct lemma for a word in context is often complicated by the fact that a word can be the inflected form of more than one lexeme each of which may have different lemmas. Lemmas can be used in various ways for NLP, for instance, to improve the performance of text similarity metrics. For this application, all words are mapped to their lemma before a similarity is calculated. Lemmas are also often used in information retrieval and information extraction to better identify and group terms which occur in their inflected forms.

The task of finding lemmas is different and harder than finding stems. Stemming is often used as a much cruder heuristic approach to map inflectional forms of words to some canonical form, but unlike lemmatization does not differentiate between different lexemes which could have the same inflectional form and it is possible for the stem of a word to not be a valid lexeme of the language.

The TreeTagger (Schmid, 2013) software provides lemmatization for 20 languages including English, German, Italian, French, Dutch and Spanish. However, it is not open source and it is not straightforward to use it for non-research or commercial applications. There exist a few other lemmatizers which are open for non-research purposes (Lezius et al., 1998; Perera and Witte, 2005; Bär et al., 2013; Cappelli and Moretti, 1983)[1]. However, these lemmatizers are mostly concerned with only one language and do not provide a broad coverage like the TreeTagger.

In this paper, we describe GATE DictLemmatizer, a plugin for the GATE NLP framework[2] (Cunningham et al., 2011) that performs lemmatization for English, German, Italian, French, Dutch, and Spanish and is freely available under the LGPL license. The GATE NLP framework is one of the most widely used frameworks for

[1] https://github.com/giodegas/morphit-lemmatizer
[2] https://gate.ac.uk

Proceedings of Recent Advances in Natural Language Processing, pages 40–45,
Varna, Bulgaria, Sep 4–6 2017.
https://doi.org/10.26615/978-954-452-049-6_006

applied natural language processing. It is implemented in Java, freely available under the permissive LGPL license and can be extended through plugins.

Our method combines the Helsinki Finite-State Transducer Technology (HFST)[3] (Lindén et al., 2011) and word-lemma dictionaries obtained from Wiktionary. Since we use separate dictionaries depending on the word category, the method also depends on a POS tagger for the language. The word dictionaries are obtained automatically from Wiktionary[4] data dumps. The code for creating the dictionaries automatically is available as free and open-source software.[5] This software can be used to easily add dictionaries for new languages to the DictLemmatizer. The plugin also contains the HFST models for the 4 languages for which models are available: English, German, French and Italian.[6]

The rest of the paper is structured as follows. First we describe our method of performing lemmatization (Section 2). Our lemmatizer uses automatically generated lemma dictionaries. The process of obtaining such dictionaries from Wiktionary is outlined in Section 3. In Section 4 we detail the release information. Next, in Section 5 we evaluate the performance of our lemmatizer. We use the TreeTagger for comparison. We conclude in Section 6.

2 Method

To obtain lemmas we combine two strategies: the Helsinki Finite-State Transducer Technology (HFST)[7] and word-lemma dictionaries obtained from Wiktionary[8]. For both strategies, it is necessary to know the coarse-grained word categories such as "noun", "verb", "adposition" for each word.

For this purpose, the lemmatizer requires the Universal POS tags[9] from the Universal Dependencies project. In GATE (Cunningham et al., 2011), POS tags can be created using different methods or plugins, however for the evaluation in this paper we use the ANNIE POS-tagger (Cunningham et al., 2002) for English and the Stanford CoreNLP POS tagger (Toutanova et al., 2003) for all other languages. These language-specific POS tags are then converted to Universal Dependencies tags using mappings adapted from `https://github.com/slavpetrov/universal-pos-tags` (Petrov et al., 2011).

The lemmatizer first tries to look up each word form in the dictionary that matches the language and word category of the word. Currently there are lists for the following categories: adjective, adposition, adverb, conjunction, determiner, noun, particle, pronoun, verb. If the word form is found in the dictionary, the corresponding lemma is used. Pre-generated dictionaries for the six supported languages are included with the plugin.

If the word could not be found in the dictionary, an attempt is made to find the lemma by using the HFST model for the language, if it is available. The HFST model returns for each word all possible morphological variants. This makes it difficult to directly find the lemma for the word. We therefore implemented rules that use the Universal POS tag information and extract the correct lemma. E.g. for the word "computers" the HFST returns the following options:

```
compute[V]+ER[V/N]+N+PL
computer[N]+N+PL
```

Since we know from the POS tagger that "computers" is a noun we can use that information and extract from the HFST list the entry that refers to a noun (`[N]`) - "computer".

The HFST models are freely available only for a few languages. For any language where there is no HFST model, our lemmatizer will rely only on the Wiktionary-based dictionaries.[10]

3 Parsing dictionaries

We implemented a Java based tool that allows users to extract lemma information from the Wiktionary API. With this tool it is easy to create dictionaries for additional languages not included in the lemmatizer distribution. We refer to this tool as Wiktionary-Lemma Extractor. It fetches for a

[3] `http://www.ling.helsinki.fi/kieliteknologia/tutkimus/hfst/`

[4] `https://www.wiktionary.org/`

[5] `https://github.com/ahmetaker/Wiktionary-Lemma-Extractor`

[6] `https://sourceforge.net/projects/hfst/files/resources/morphological-transducers/`

[7] `http://www.ling.helsinki.fi/kieliteknologia/tutkimus/hfst/`

[8] `https://www.wiktionary.org/`

[9] `http://universaldependencies.org/u/pos/all.html`

[10] In this case, it is also possible to make DictLemmatizer work without any POS tags at all by merging the original dictionaries per word type into one dictionary for unknown/unidentified POS type.

given word form its lemma from the Wiktionary page. In addition the tool expects the language information, such as English, German, etc. Once these pieces of information are provided the tool fetches through the Wiktionary API the English version of the Wiktionary page for the queried word. The English Wiktionary page is divided into different areas where each area conveys a particular information such as lemma, synonym, translation, etc. Our tool isolates the lemma area and finds the non-inflected form for the queried word. The queried word and the non-inflected form are saved into a database to be used as dictionary lookup.

4 Software Availability

4.1 GATE DictLemmatizer Plugin

Most of the tools and resources for the GATE NLP framework are created as separate plugins which can be used as needed for a processing pipeline. The approach for finding lemmas described earlier has been implemented as a GATE plugin and is freely available from `https://github.com/GateNLP/gateplugin-dict-lemmatizer`. This plugin only implements the lemmatization part since there are already several plugins for tokenisation, sentence splitting, and POS-tagging included or separately available for GATE.

4.2 Wiktionary-Lemma Extractor

Similar to the GATE Plugin for lemmatization we make our Wiktionary-Lemma Extractor publicly available through github[11]. Along with the code we also provide a client that ease the creation of new dictionaries. The client just expects the input of the target language such as English, German, Turkish, Urdu, etc. The client first collects all possible words for that particular language from Wiktionary titles, determines for each title word its lemma and finally extract the lemma dictionaries. These lemma dictionaries can then be directly injected into the GATE Plugin.

5 Evaluation

We evaluated DictLemmatizer and compared it to TreeTagger on the following corpora:

- English British National Corpus (EN-BNC) (Consortium, 2007; Clear, 1993)
- German Tiger Corpus (DE-Tiger) (Brants et al., 2004)
- Universal Dependencies English tree bank (EN-UD) (Bies et al., 2012)
- Universal Dependencies French tree bank (FR-UD)
- Universal Dependencies German tree bank (DE-UD)
- Universal Dependencies Spanish tree bank (ES-UD)
- Universal Dependencies Spanish Ancora corpus (ES-Ancora)

For more information on the Universal Dependencies tree banks see McDonald et al. (2013).

All corpora were converted to GATE documents using format specific open-source software[12][13][14]. The software and setup for carrying out all evaluation is also available online.[15]

Note that for this comparison, the GATE Generic Tagger Framework plugin[16] was used to wrap the original TreeTagger software. This plugin does not use the full processing pipeline of the original TreeTagger software[17] but instead just uses the `tree-tagger` binary to retrieve per-token information.

All corpora were converted so that the token boundaries from the corpus were preserved for the conversion to GATE format. However, for the evaluation, the tokens produced by the annotation pipeline are based on the GATE tokenizer and can therefore differ from the correct tokens as present in the tree bank. We list the performance of the tokeniser used together with the performance of the lemmatizer on the tokens which match exactly. Some corpora use corpus-specific ways to represent multi-token words or multi-word tokens which cannot be represented in an identical way as GATE annotations and so these cases get excluded

[11] `https://github.com/ahmetaker/Wiktionary-Lemma-Extractor`

[12] `https://github.com/GateNLP/corpusconversion-bnc`

[13] `https://github.com/GateNLP/corpusconversion-tiger`

[14] `https://github.com/GateNLP/corpusconversion-universal-dependencies`

[15] `https://github.com/johann-petrak/evaluation-lemmatizer`

[16] `https://gate.ac.uk/userguide/sec:parsers:taggerframework`

[17]

from the evaluation. Results of the evaluation reported in accuracy are shown in Table 1.

From the results in Table 1 we can see that for English and German, DictLemmatizer outperforms TreeTagger. For French, TreeTagger achieves better performance in the test corpora; however, for the training corpora DictLemmatizer achieves better results. For Spanish, the TreeTagger results are much better. The reason for this is that apart from TreeTagger's outstanding performance on the Spanish corpora, the HFST inducer is not used in our Lemmatizer for Spanish because there is no HFST model available, so the lemmatization is performed only using the lemma dictionaries obtained from Wiktionary.[18] Although there is a big performance difference for the Spanish language, we consider that this result is satisfactory given the restrictions.

To get a better indication of the performance of each of the two strategies for the other languages, we also performed evaluations using the DictLemmatizer where we used only the dictionary-based or only the HFST-based approach. Table 2 shows the results for all corpora except Spanish (where only the dictionary is used by default). We can see that for English and German the performance of using just HFST and using just lemma dictionaries achieve comparable results, though using only lemma dictionaries is always slightly better. This pictures looks different when we look at the French language. There using only HFST clearly wins against using only the lemma dictionaries and achieves around 10% better accuracy. Nevertheless both resources are complementary and when combined boost the results as seen in Table 1.

In addition to the Wiktionary source, the word lists can be extended by an annotated training corpus. We tested this by finding the 500 most frequent incorrect assignments on each of the Universal Dependencies training corpora grouped by target POS tag and adding those to the dictionaries for each language. The evaluations using those extended word lists are shown in 1 with the indication "DL-TR". This improves the accuracy on all Universal Dependencies training and test sets and on the BNC corpus, but slightly decreases accuracy on the Tiger corpus.

Along with the accuracy figures, we also recorded the time needed to process each corpus (see Table 1). The timing information only gives a rough indication because we show the results of a single run only, and because the machine was under different load for different runs. Also note that the times for the TreeTagger include the overhead of wrapping the original TreeTagger binary for use in a Java plugin. The implementation of the Generic Tagger Framework plugin executes the binary for every document, so the timing information includes the overhead for this and thus also depends on the average document size for a corpus, while the timing for the DictLemmatizer does not. However, from these rough results we can still see that DictLemmatizer is always significantly faster than the TreeTagger for the concrete GATE-plugin implementations that were compared.

6 Conclusion

In this paper we presented a lemmatizer for six languages: English, German, Italian, French, Dutch and Spanish that is easily extensible to other languages. We compared the performance of our lemmatizer to the one of TreeTagger. Our results show that our lemmatizer achieves similar or better results when there is support from HFST. In case there is no HFST support we still achieve satisfactory results.

Both the DictLemmatizer and the lemma dictionary collector software are available freely for commercial use under the LGPL license. The dictionary collector can be used to easily extend the lemmatizer to new languages that are currently not included in DictLemmatizer.

Acknowledgments

This work was partially supported by the European Union under grant agreement No. 687847 COMRADES and PHEME project under the grant agreement No. 611223.

References

Daniel Bär, Torsten Zesch, and Iryna Gurevych. 2013. Dkpro similarity: An open source framework for text similarity. In *ACL (Conference System Demonstrations)*. pages 121–126.

Ann Bies, Justin Mott, Colin Warner, and Seth Kulick. 2012. English web treebank LDC2012T13. Web

[18] In our evaluation we focused on languages which are rich in resources and high performing lemmatizers such as English, German and French and also supported by HFST and languages that are less rich in terms of resources and also has no support by the HFST tool such as Spanish.

Corpus	TT	DL	DL-TR	Time DL	Time TT
EN-BNC	0.927	0.938	**0.958**	1:47:23	3:11:29
EN-UD-Test	0.922	0.945	**0.972**	0:00:14	0:11:08
EN-UD-Train	0.920	0.941	**0.972**	0:01:18	1:14:54
DE-Tiger	0.804	**0.940**	0.925	0:12:12	0:53:07
DE-UD-Test	0.841	0.915	**0.946**	0:00:44	0:13:34
DE-UD-Train	0.783	0.910	**0.944**	0:05:08	3:02:56
FR-UD-Test	0.902	0.881	**0.961**	0:00:12	0:03:00
FR-UD-Train	0.882	0.896	**0.973**	0:04:10	1:31:28
ES-Test	0.904	0.779	**0.936**	0:00:18	0:01:47
ES-Train	0.884	0.797	**0.954**	0:02:41	0:45:50
ES-Ancora-Test	**0.993**	0.806	0.950	0:00:26	0:05:30
ES-Ancora-Train	**0.993**	0.807	0.953	0:07:21	0:58:06

Table 1: Performance of TreeTagger (TT) and our Dictionary Lemmatizer (DL) and Dictionary Lemmatizer trained on the UD training set (DL-TR) on different corpora. Figures are accuracy of lemma (ignoring case) for tokens matching the corpus token boundaries, times are in HH:MM:SS.

Corpora	HFST	Lemma Dicts
BNC-EN	0.89	0.924
UD-Test-EN	0.905	0.933
UD-Train-EN	0.90	0.928
Tiger-DE	0.812	0.853
UD-Test-DE	0.827	0.827
UD-Train-DE	0.817	0.827
UD-Test-FR	0.878	0.799
UD-Train-FR	0.894	0.819

Table 2: Performance of DictLemmatizer (HFST only and Wiktionary lemma dictionary only) on different corpora.

Download. Philadelphia: Linguistic Data Consortium.

Sabine Brants, Stefanie Dipper, Peter Eisenberg, Silvia Hansen-Schirra, Esther König, Wolfgang Lezius, Christian Rohrer, George Smith, and Hans Uszkoreit. 2004. Tiger: Linguistic interpretation of a german corpus. *Research on Language and Computation* 2(4):597–620. https://doi.org/10.1007/s11168-004-7431-3.

Amedeo Cappelli and Lorenzo Moretti. 1983. *Aspetti della rappresentazione della conoscenza in linguistica computazionale*, volume 5. Pacini.

Jeremy H. Clear. 1993. The digital word. MIT Press, Cambridge, MA, USA, chapter The British National Corpus, pages 163–187. http://dl.acm.org/citation.cfm?id=166403.166418.

BNC Consortium. 2007. The british national corpus, version 3 (bnc xml edition). Distributed by Oxford University Computing Services on behalf of the BNC Consortium. http://www.natcorp.ox.ac.uk/. http://www.natcorp.ox.ac.uk/.

Hamish Cunningham, Diana Maynard, Kalina Bontcheva, and Valentin Tablan. 2002. GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. In *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02)*.

Hamish Cunningham, Diana Maynard, Kalina Bontcheva, Valentin Tablan, Niraj Aswani, Ian Roberts, Genevieve Gorrell, Adam Funk, Angus Roberts, Danica Damljanovic, Thomas Heitz, Mark A. Greenwood, Horacio Saggion, Johann Petrak, Yaoyong Li, and Wim Peters. 2011. *Text Processing with GATE (Version 6)*.

Wolfgang Lezius, Reinhard Rapp, and Manfred Wettler. 1998. A freely available morphological analyzer, disambiguator and context sensitive lemmatizer for german. In *Proceedings of the 17th international conference on Computational linguistics*. Association for Computational Linguistics, pages 743–748.

Krister Lindén, Erik Axelson, Sam Hardwick, Miikka Silfverberg, and Tommi Pirinen. 2011. HFST–framework for compiling and applying morphologies pages 67–85.

Ryan McDonald, Joakim Nivre, Yvonne Quirmbach-Brundage, Yoav Goldberg, Dipanjan Das, Kuzman Ganchev, Keith Hall, Slav Petrov, Hao Zhang, Oscar Tackstrom, Claudia Bedini, Nuria Bertomeu Castello, and Jungmee. 2013. Universal dependency annotation for multilingual parsing. In *Lee Proceedings of ACL 2013*.

Praharshana Perera and René Witte. 2005. A self-learning context-aware lemmatizer for german. In

Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing. Association for Computational Linguistics, pages 636–643.

Slav Petrov, Dipanjan Das, and Ryan T. McDonald. 2011. A universal part-of-speech tagset. *CoRR* abs/1104.2086. http://arxiv.org/abs/1104.2086.

Helmut Schmid. 2013. Probabilistic part-ofispeech tagging using decision trees. In *New methods in language processing*. Routledge, page 154.

Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*. Association for Computational Linguistics, Stroudsburg, PA, USA, NAACL '03, pages 173–180.