NLP Resources for a Rare Language Morphological Analyzer: Danish Case

Mykhailo Kotov (orcid.org/0000-0001-8327-5197)

Envion Software / V.N. Karazin Kharkiv National University, Kharkiv, Ukraine

mykhailo.kotov@gmail.com

Abstract. The paper discusses the characteristics and practical aspects of application of the natural language processing resources available for developing a rare language morphological analysis solution. The case under consideration reveals the pipeline design needed to prepare the grammatical resources for Danish. Being rare not only in terms of distribution, but also in the amount of natural language resources available, the Danish language represents a significant problem in terms of application of third-party tools to help solve various NLP-related issues. The paper focuses on part-of-speech tagging and lemmatization, typical but indispensable tasks at the pre-processing stage within the framework of developing a morphological analyzer as a custom NLP solution.

Keywords: morphological analyzer, lemmatization, part-of-speech tagging, Hunspell, OpenNLP, Snowball stemmer, SyntaxNet, word-list.

1 Introduction

Developing a morphological analyzer is a multi-staged complex process, whose starting point is preparation of word-list(s) with indicated grammatical meanings of entries. Such word-list(s) preparation includes the essential pre-processing measures – tokenizing, cleaning, permuting, part-of-speech (POS) tagging, lemmatizing or stemming. At the same time, this initial stage, aiming at obtaining a ―lemma-derivative-grammatical meaning" format out of an unsortedword-list rich in inflected and suppletive forms, appears to be resource-consuming and demands cost-efficient solutions and optimization.

The task of proper word-list processing becomes no less difficult in case of its implementation for a rare language. In our classification, rare is a language with not only few speakers and/or limited distribution [8], but also, what is more important, a limited number of natural language processing (NLP) and linguistic resources available.

In this paper, the focus is on Danish – a language rare in terms of availability of NLP solutions. Being a typical representative of the group of analytic Germanic languages, Danish is characterized by a decent number of inflected forms for all parts of speech, among which the postpositive usage of the definite article with nouns (e.g. *en hund – hund***en**) is of special interest. Certain word-forms are marked by suppletion (e.g. *god – bedre – bedst*). In Section 2, POS tagging resources available for Danish are analyzed and compared. Section 3 deals with the description of properties of the lemmatizing/stemming solutions. The criteria used for the analysis and comparison includethe declared quality (per token accuracy), operating system (OS) compatibility, and licensing policy. Finally, Section 4 outlines a possible approach on how to handle the word-list processing issue. In the outline, nonetheless, I do not concentrate in detail on the cleaning and permuting, as to a large extent the above can be accomplished without using any additional external solutions.

2 Part-of-Speech Tagging Solutions for the Danish Language

Being critical for almost every natural language processing system, POS tagging is still receiving a great deal of attention. Traditionally, based on the type of information in use, several approaches to implementation of POS tagging solutions are distinguished: rule-based, stochastic/probabilistic, and the combination of the two (hybrid). The first type, as evident from its name, is based on the linguistic models which ―range from a few hundreds [sic] to several thousand rules, and they usually require years of labour" [11]. The prototypical representative of the rule-based approach among POS-taggers is Brill's tagger [4]. The stochastic and hybrid approaches are becoming more popular nowadays, in particular in spheres concerning but not limited to neural networks application [1, 9].

Considering the Danish language, it is important to point to the fact that the part-of-speech resources available are to a large extent based on stochastic approaches and represented by domain leaders. These are Apache OpenNLP POS-tagger and SyntaxNet by Google. Among the rule-based taggers, it is necessary to mention Brill's adapted CST's POS tagger [5].

Apache OpenNLP project provides models for part-of-speech tagging. The OpenNLP POS tagger uses a probability model in order to predict the precise part-of-speech tag out of the tag-set. In order to limit the possible tags, one can make use of a tag dictionary aiming as well at increasing the tagging precision and runtime performance. For testing, it is advised, to try out the part-of-speech tagger via the command line tool. But the API is also available for embedding into an application. Licensed under the Apache License, OpenNLP POS tagger shows decent results when language models match the input text, and the latter is correctly decoded [3]. The tagger is OS-independent. Judging by tests [6], if pre-trained on the training part of the Danish Dependency Treebank, with part-of-speech tags converted to the Google universal tag-set, the POS-tagger can achieve the accuracy of 96.8% on the test portion of the Danish Dependency Treebank.

Another novel representative of the stochastic approach family is Google's SyntaxNet, an open-source implementation of the method discussed in [1]. SyntaxNet has been integrated in the TensorFlow framework and accompanied by ParseyMcParseface parser, ―tuned for a balance of speed, simplicity, and accuracy". For the latter, there is a set of syntactic models available. The models are pretrained on Universal Dependencies datasets. The Danish language model boasts the accuracy of 95% over all tokens, including punctuation. Despite the high quality of the performance and favorable licensing policy (available under Apache License), SyntaxNet‘s limitations are connected with OS compatibility. At present, as integrated into the TensorFlow framework, the solution is functioning under UNIX systems.

CST‘s POS-tagger, available under GNU General Public License, is represented by the adapted version of Brill‘s tagger. As stated, the distribution comprises Brill‘s original; distribution and the archive with CST‘s software adaptations (reformatting to C++ standard, better handling of capitals in headings, making the source code independent of language etc.) [12]. The tagger was trained on DSL‘s publicly accessible PAROLE Corpus. In terms of architecture, as stated in [3], CST‘s tagger is characterized by the restricted access to a web version only, which can hardly be suited for a large amount of text.

The overall comparison of the discussed solutions is represented in Table 1.

Table 1. Comparison of POS-taggers for the Danish language

Solution	**Accuracy**	**OS Compatibility**	**Licensing Policy**
OpenNLP	96.8%	any	Apache License
SyntaxNet	95%	Unix	Apache License
CST‘s	N/A	any	GPL License

3 Stemming and Lemmatization Solutions for the Danish Language

A different but equally important procedure within the pipeline for a word-list processing is returning the base word-form. Such ―dictionary" word-forms can be obtained from two similar but at the same time different in nature processes – stemming and lemmatization. As given in [10], ―Stemming usually refers to a crude heuristic process that chops off the ends of words in the hope of achieving this goal correctly most of the time, and often includes the removal of derivational affixes. Lemmatization usually refers to doing things properly with the use of a vocabulary and morphological analysis of words, normally aiming to remove inflectional endings only and to return the base or dictionary form of a word, which is known as the lemma."

The choice between the solutions is predetermined by the available resources and specification, and often can be characterized by the fact that stemming solutions are easier to get and make use of; they tend to increase recall, but hurt precision. However, lemmatizers, being more precise, but thus more ―knowledgeable" and complex in design, are under stricter limitations in terms of licensing policy, let alone the case with rare languages. The available and noteworthy solutions for the Danish language are Hunspell, CST‘s lemmatizer, and NLTK compatible Snowball stemmer.

A well-known solution for primarily spelling checking, Hunspell has proven efficient as a stemmer with varying degrees of accuracy for different languages. Available as GPL/LGPL/MPL tri-license, Hunspell, at the same time, can be used with any type of OS and employs Unicode character encoding. Hunspell delivery includes the dictionary files: one with base forms and links to declension/conjugation patterns, and the other describing the mentioned patterns to generate derived and inflected forms for lemmas.

The Snowball stemmer, or, to be more precise, stemming algorithm, together with the stop word list and Snowball compiler, represent, in case of Danish, a hardly viable alternative for Hunspell. Although the major advantage is compatibility with the Natural Language Tool Kit (NLTK) and BSD licensing policy, which eases largely the application of the stemmer.

Finally, CST‘s lemmatizer solution [7] consists of a rules set and a dictionary. The rules set is derived from the Large Computational Dictionary, STO, and the verification of the output boasts 94%-98% accuracy, depending on whether the entries have been supplied with the proper grammatical meanings or not. CST‘s lemmatizer is freely available for non-commercial applications, but special permission is required for commercial usage.

The overall comparison of the discussed stemming and lemmatizing solutions is represented in Table 2.

Table 2. Comparison of Stemmers and Lemmatizers for the Danish language

Solution	**Accuracy**	**OS Compatibility**	**Licensing Policy**
Hunspell	N/A	any	GPL/LGPL/MPL tri-license
Snowball	N/A	any	BSD
CST‘s	94-98%	any	Non-commercial use – free, special permission for commercial use

4 Word-List Processing Pipeline

As discussed earlier, the final aim of the application of multiple third-party resources is to construct a word-list of a specific form for further usage as a linguistic resource for a custom NLP solution. In this case, a base form (lemma) of a certain lexeme stands next to its derived or inflected word forms, and each and every word form, including lemma, receives a proper grammatical description. Such an outcome makes further manual processing of the linguistic information (verification with subsequent further more elaborated classification) much easier.

The suggested pipeline consists of several stages, which immediately follow one another (Fig 1.). Resulting from the preliminary Stage 0 (tokenization, cleaning, and permuting), as the input for further processing we have a list of tokens, one per line. During Stage 1, the word-list is verified for consistency, minor mistakes and omissions, resulting from Stage 0 and influencing further activities, are fixed.

The next two stages (Stage 2 and 3) are the key for subsequent classification. By applying third-party resources available we add extra features – grammatical meaning, in the form of part-of-speech tags, and base word forms for each entry. Such extra information significantly enhances our capability for correct automatic juxtaposing within the triplet lemma – derived/inflected word-form – part-of-speech tag.

It is important to mention the fact that Stages 2 and 3 can be iterated using different available resources to reach the optimum for precision and recall. Each stage can be followed by manual revision to tweak the list for the next procedure.

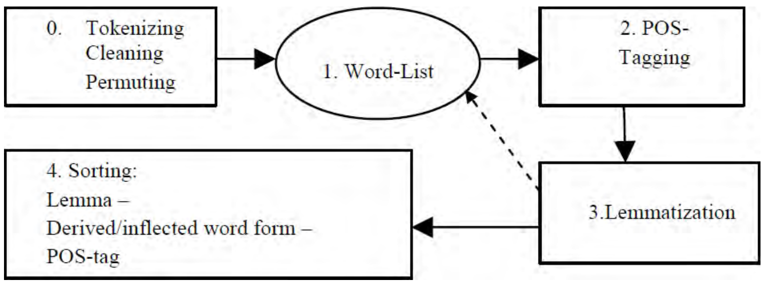

Fig. 1. Pipeline for Word-List Processing

Finally, Stage 4 presupposes sorting procedures. Taking into consideration the extra features obtained resulting from Stages 2 and 3, we build up the list in the desired format. Typically, we first group by lemma, and then by the relevant part-of-speech tag. Lemmatizers, let alone the notorious homonymy issue, may yield nothing at all, leaving an empty row, thus manual revision and verification here are indispensible. In case of stemmers, the output, depending on language specific features and, surely, a stemmer‘s accuracy, may turn out to be of significantly inferior quality, thus demanding more efforts for revision and improvements introduction.

5 Conclusion

The presented pipeline for processing the word-lists, one of the initial stages in developing a rare language (Danish) morphological analyzer, opens the way for significant reduction of manual labor. This happens due to automation of part-of-speech tagging and lemma ascribing processes.

The list of the available resources and their comparative analysis aimed to help solve the discussed above issue. The assistance consisted in both providing references to existing libraries for part-of-speech tagging, lemmatization or stemming of the Danish language, and indicating possible stumbling blocks for the resource application, e.g. operating system compatibility, licensing policy, accuracy of the solution.

The discussed pipeline is the first step in designing the framework for automation of manual labor and optimization of the available resources allocation.

References

1. Andor, D., Alberti, C., Weiss, D., Severyn, A., Presta, A., Ganchev, K., Petrov, S., and Collins, M. (2016). Globally normalized transition-based neural networks. In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics* (p. 2442–2452).
2. *Apache OpenNLP Developer Documentation*. (2017). Retrieved from https://opennlp. apache.org/documentation/1.5.2-incubating/manual/opennlp.html.
3. Asmussen, J. (2015). *Survey of POS taggers. Approaches to making words tell who they are* (Technical Report DK-CLARIN WP 2.1). Retrieved from http://korpus.dsl.dk/clarin/ corpus -doc/ pos-survey.pdf

4. Brill, E. (1995). Transformation-based error-driven learning and natural language processing: a case study in part-of-speech tagging. *Comput. Linguist.*,*21*, 543–565.
5. Hansen, D.H. (2000). *Træningogbrugaf Brill-taggerenpådansketekster* (Ontoquery Technical report). Retrieved from https://cst.dk/online/pos_tagger/Brill_tagger.pdf
6. Johannsen, A. (2014). A trainable Part-of-Speech Tagger and Dependency Parser for Danish. Available at: https://github.com/andersjo/danish_dependency_parser/blob/master/README.md
7. Jongejan, B., and Dalianis, H. (2009). Automatic training of lemmatization rules that handle morphological changes in pre-, in- and suffixes alike. In: *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*. (pp. 145–153).
8. Lewis, M.P., Simons, G.F., and Fennig, C.D. (eds.). (2013). *Ethnologue: Languages of the World*. Dallas, Texas: SIL International.
9. Ling, W., Dyer, C., Black, A.W., Trancoso, I., Fermandez, R., Amir, S., Marujo, L, and Luis, T. (2015). Finding function in form: Compositional character models for open vocabulary word representation. In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing* (pp. 1520– 1530).
10. Manning, C.D., Raghavan, P., Schütze, H. (2008). *Introduction to Information Retrieval*. New York: Cambridge University Press
11. Marquez i Villodre, L. (1999). *Part-of-speech Tagging: A Machine Learning Approach based on Decision Trees* (Doctoral dissertation). Retrieved from https://upcommons.upc.edu/ bitstream/handle/2117/93974/TLMV1de2.pdf
12. taggerXML [Computer software and its adaptations]. Retrieved from http://cst.dk/download/uk/index.html#tagger