Computer-Assisted Enlargement of Morphological Dictionaries

Jan Daciuk
Alfa Informatica, Rijksuniversiteit Groningen
e-mail: *j.daciuk@let.rug.nl*

Abstract

We present a tool that helps in enlargment of already existing morphological dictionaries. The tool works by analysing the contents of an existing (core) dictionary, finding rules that associate endings and prefixes with morphological descriptions. Those associations are then used to find predictions (or guesses) about descriptions of new words. The whole linguistic knowledge needed for the task is extracted from the core dictionary. A graphical interface greatly facilitates the task of finding and choosing the correct description.

1 Introduction

Creating a morphological dictionary for a morphologically rich language can be divided into two parts: designing the core dictionary (writing inflection/derivation rules and spelling rules), and adding new words.

[10] describe a system for building the core morphology, where much information is elicited from human informants. In [8], an algorithm for automatically acquiring morphological links between words for a domain-oriented stemmer is developed. [7] and [13] described discovery of spelling rules. [16] describe a successful, minimally supervised system for discovering the core morphology from corpora. Un unsupervised system for learning derivational morphology from inflectional lexicons (using probabilistic methods) is described in [6]. Un unsupervised system learing from corpora, and focusing mainly on regular morphology can be found in [12]. It may be tempting to think that systems like [16] are all we need to build full morphological dictionaries, as they not only discover rules, but can build a lexicon as well. However, for languages with rich flection (e.g. Slavic languages), this does not seem to be true. Usually, only few flectional forms of a lexeme are found in corpora. For example, for verbs, some declarative present and past tense forms are usually found, but this still leaves plenty of room for possible variations of imperative or participles. The result is that words cannot be added automatically - there are several possibilities to choose from, and human supervision is needed.

Surprisingly little effort has been put into automating the second part. In a US patent US05412567 granted to Xerox on May 2nd, 1995, Lauri Karttunen proposes a system that adds a new word to a dictionary when an informant provides a "model" word that has surface forms analogous to the new word. However, it is up to the human operator of the system to find the model word. In other systems for lexicon acquisition, e.g. [15], lexical rules are used for generation rather than for analysis, and the focus is on semantics, not on morphology.

Our goal is to facilitate the task of adding new words to an existing morphological dictionary. The dictionary we used was in the format of mmorph ([11]), a morphology tool from ISSCO, Geneva, Switzerland. The dictionary is not an annotated word list. It consists of some initial parts describing the alphabet, inflection or derivation rules, spelling rules, as well as the lexicon itself. For each entry in the lexicon, the entry description, the base form, and the canonical form are given. The description is a feature structure containing information that makes it possible to generate all inflected or derived forms of that entry. It may include part of speech, gender, paradigm name, etc. We assume that all or almost all spelling rules and inflection rules have already been written, and that the existing morphological dictionary provides enough examples for every rule combination so that it is possible to infer associations between endings, prefixes, and particular morphological descriptions of words that can be used to generate them. We also assume that the nature of the language is such that those associations exist. Our system proposes a list of possible descriptions to the user. The user's task is then only to select the appropriate description.

2 Finding Associations

In our experiments, we used a Polish morphological dictionary written in the format of mmorph ([11]) to find rules that associate certain endings and prefixes with the corresponding descriptions of lexemes. Such a rule should also describe how to transform the word in question to obtain its base form and its canonical form. For example, in a Polish morphological dictionary, there should be a rule that associates the ending *-ytego* and the prefix *nie-* with at least two rules:

- v[asp=imp par_v=kryc form=stem] "Δ^1" = "$\Delta_l^3\Delta_r^4$ć"
- v[asp=per par_v=kryc form=stem] "Δ^1" = "$\Delta_l^3\Delta_r^4$ć"

The first part specifies that the lexeme is a verb, the aspect is either perfective or imperfective, that the paradigm is that of type "kryc", and the form is stem. The last part specifies the transformation that must be performed on the word form to get the canonical form. Δ_l^i specifies that i characters should be deleted from the beginning of the inflected form, Δ_r^i – i characters from the end of the inflected form to get the base for the canonical form[1]. To obtain the canonical form, we append its ending – in this case it is "ć". The central part (in front of the equal sign) specifies the transformation from the canonical form to the base form. Δ^i says that i characters should be deleted from the end of the canonical form. No ending is appended.

Such rules can be discovered using a variety of techniques[2]. An obvious choice would be to use the transformation-based error-driven learning ([3], [2]). Patterns of transformations should be established, and then individual transformations learned by choosing the best scoring transformation in each step. [9] used Brill's technique on a lexicon. However, his goal was not to learn the associations between the endings and the corresponding description. He learned only the corresponding POS tags.

We decided to use a different, quicker approach, based on finite-state automata. We used the algorithm described in [4] as the basis, but we made substantial modifications. The method is based on the observation that if there is an association between endings and the corresponding descriptions, then all (regular) words should have one of a few descriptions associated with their endings. We reverse the words, and append descriptions at the end of them. If we take another word, reverse it, and match it against our collection of strings, its ending (now at the beginning) should match with endings of words with the same ending. We can retrieve those words, and extract associated descriptions.

It is possible to extract pairs (inflected form, description) from the dictionary for every inflected form generated from the description. Transformations leading to the canonical form and to the base form are encoded. *Mmorph* can use *archephonemes* to represent e.g. umlautable letters in German. If *archephonemes* are used in endings, or in the final part of roots, they are handled by the ending deletion mechanism described above. If they appear earlier in roots, another code describes their position, how many characters they replace, and spells them out. We form strings consisting of: reversed inflected form without prefix, separator, prefix, separator, encoded transformations, description. If the inflected form

[1] This is not exactly a stem, as endings for the inflected and canonical form may share some letters

[2] They can also be discovered by hand. Jan Tokarski spent years of research to compile a list of rules for Polish ([14]). Using the method described in the present paper, the same job takes a few minutes of the CPU.

does not have a prefix, then the two separators are just adjacent. Moving the prefixes from the inflected form into a special place in the string is necessary because of the pruning process later on.

Those strings are then used to construct a finite-state automaton such that the language recognized by the automaton be the set of those strings. The automaton is then pruned. Pruning ([4]) removes the stems of words, so that the (reversed) endings lead directly to the corresponding descriptions. For any state in the automaton, if all paths from that state lead to the same set of descriptions, they are replaced with a single transition from that state to the state from which all those descriptions are reachable. There are additional heuristics that help making generalization and increase recall. If a certain state has many outgoing transitions that lead to a small number of different sets of descriptions, then a new state unifying those descriptions is created, and all paths from that state to the descriptions are replaced by a single transitions to that newly created state.

Before pruning begins, each transition is associated with the number of strings that are recognized by the part of the automaton that begins with the target state of that transition. Those numbers serve as weights, as they are not removed by pruning.

We used only regular words for construction of the automaton. It is known (cf. [1], [5, pp. 140–144]) that words that occur only once in corpora better estimate unknown words than the whole dictionary. However, our dictionary is in its initial state, and it is still missing some of quite frequent words.

3 Applying Associations

In the application phase, unknown words are reversed. Each word is searched for in the automaton. Starting from the initial state, transitions labeled with consecutive letters of the reversed word are followed, until such a state is reached that there is no transition labeled with the next letter of the reversed word. The recognized part of the unknown word roughly corresponds to its ending. All descriptions reachable from that state are printed. Weights on transitions are used for establishing the order between descriptions. "Heavier" descriptions, i.e. those that were associated with a larger number of words, are printed first. This process may produce descriptions that cannot generate the inflected form that was used to infer them. Therefore, a filter is used that removes all such descriptions. Additionally, if the same description appears in more than one word, it is ranked higher than others, regardless of their weights. The underlying assumption is that the correct description should appear in all inflected forms of the lexeme, but various forms can have various additional (incorrect) descriptions.

A graphical user interface (GUI) written in Tcl/Tk is used to assist the user in selecting the right description (fig. 1). The user chooses a word from the `Word form` pane. A list of possible descriptions is displayed in the `Descriptions` pane. The user selects a description and saves it. Saving removes from the `Word form` pane all word forms generated by the description, only the current form, or nothing, depending on the state of a radio button. If the user is not sure what the description generates, he or she can press `Mmorph` button to obtain a list of generated forms in the `Mmorph output` pane. To compare two descriptions, the user selects them both and presses the `Mmorph` button. The differences are shown in the `Mmorph output` pane. It is possible to manually correct the descriptions.

Sometimes it is difficult to find the right description. In that case, the user can correct the contents of the `Mmorph output` pane and press the `mAtch mmorph` button. The tool will try to find a matching description.

The new entries are saved in a text file. They can be incorporated into the initial dictionary file (mmorph format) either by hand, or using a simple perl script.

4 Results

For such tasks, the standard measures are recall, precision, and coverage. However, to compute them, we would have to have the complete dictionary. Our dictionary is just being built. For each word, we were interested in a single analysis even if more correct

Figure 1: A snapshot of the Tcl/Tk interface

analyzes were possible for that word. Our Polish dictionary contains about 1800 lexemes that generate about 60 000 inflected word forms. We evaluated over 750 unknown word forms. For the forms that were not processed, there were either missing paradigms, or missing spelling rules, so there existed no correct descriptions for them. 750 word forms were processed. For 62.53% of them, the correct description was among those proposed by the tool. For 44.53% of them, the first description proposed by the tool was the correct one. The tool proposed 2.08 descriptions on average for a word. 62.53% is a low number, but it was calculated for each individual unknown inflected form. However, the correct description can be associated with another form of the same lexeme. 89.73% of unknown inflected forms had the correct description among guesses, but not necessarily attached to every one of them. Guesses for the remaining inflected forms were not totally correct, but in most cases the part of speech was right, and often the difference between the correct and incorrect description was a value of one feature. Such imperfection can easily be corrected using menus in the Tcl/Tk interface.

5 Conclusions

We have presented the first tool for facilitating the process of adding new lexemes to an existing morphological dictionary. It allowed us to add new words much faster than it would be possible without it. In the current system, the user launches a script that does the whole job of extracting information from a morphological dictionary in mmorph format. The only thing the user has to specify is whether the language makes use of prefixes or infixes, and what archephonemes (if any) are to be found in roots or stems not close to their ends. No additional knowledge is required, except for the ability to recognize correct and incorrect forms. The tool is available in source form from `http://www.pg.gda.pl/~jandac/fsa.html`. It is free for non-commercial purposes.

The tool works with mmorph, but it should be relatively simple to coerce it to use another format. One of the reviewers suggested using it with agglutinative languages. In our (limited) understanding of those languages it would mean that a series of endings would have to be analyzed. This translates to repetitive use of the guessing automaton. While this should be relatively easy – the major obstacle would be to code information when to stop, preparation of data for the automaton would have to be quite different, and apparently more difficult.

6 Acknowledgements

This research was carried out within the framework of the PIONIER Project *Algorithms for Linguistic Processing*, funded by NWO (Dutch Organization for Scientific Research) and the University of Groningen. We would like to thank anonymous reviewers for help-

ful comments that contributed to improvements in the paper.

References

[1] Harald Baayen and Richard Sproat. Estimating lexical priors for low-frequency morphologically ambiguous forms. *Computational Linguistics*, 22(2):155–166, June 1996.

[2] Eric Brill. *A Corpus-Based Approach to Language Learning*. PhD thesis, Department of Computer and Information Science, University of Pennsylvania, USA, 1993.

[3] Eric Brill. Transformation-based error-driven learning and natural language processing: A case study in part-of-speech tagging. *Computational Linguistics*, 21(4):543–565, December 1995.

[4] Jan Daciuk. Treatment of unknown words. In *proceedings of Workshop on Implementing Automata WIA'99*, pages IX–1 – IX–9, Potsdam, Germany, July 1999.

[5] Evangelos Dermatas and George Kokkinakis. Automatic stochastic tagging of natural language texts. *Computational Linguistics*, 21(2):137–163, June 1995.

[6] Éric Gaussier. Unsupervised learning of derivational morphology from flectional lexicons. In *ACL '99 Workshop: Unsupervised Learning in Natural Language Processing*, Univ. of Maryland, 1999.

[7] Daniel Gildea and Daniel Jurafsky. Learning bias and phonological-rule induction. *Computational Linguistics*, 22(4):497–530, December 1996.

[8] Christian Jacquemin. Guessing morphology from terms and corpora. In *Proceedings of 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'97)*, Philadephia, USA, July 1997.

[9] Andrei Mikheev. Automatic rule induction for unknown-word guessing. *Computational Linguistics*, 23(3):405–423, September 1997.

[10] Kemal Oflazer and Sergei Nirenburg. Practical bootstraping of morphological analyzers. In *Proceedings of Computational Natural Language Learning (CoNLL-99) Workshop at EACL'99*, Bergen, Norway, 1999.

[11] Dominique Petitpierre and Graham Russell. MMORPH – the multext morphology program version 2.3. Delivrable 2.3.1, ISSCO, University of Geneva, Route des Acacias, CH-1227 Carouge, Switzerland, October 1995.

[12] Patrick Schone and Daniel Jurafsky. Knowledge-free induction of inflectional morphologies. In *Proceedings of Language Technologies 2001: Second Conference of the North American Chapter of the Association for Computational Linguistics*, Pittsburgh, Pennsylvania, USA, June 2001.

[13] Pieter Theron and Ian Cloete. Automatic acquisition of two-level morphological rules. In *Fifth Conference on Applied Natural Language Processing*, pages 103–110, Washington, DC, USA, April 1997. Association for Computational Linguistics, Association for Computational Linguistics.

[14] Jan Tokarski. *Schematyczny indeks a tergo polskich form wyrazowych.* Wydawnictwo Naukowe PWN, 1993.

[15] Evelyne Viegas, Boyan Onyshkevych, Victor Raskin, and Sergei Nirenburg. From *submit* to *submitted* via *submission*: On lexical rules in large-scale lexicon acquisition. In *Proceedings of ACL'96*, Santa Cruz, California, USA, June 1996.

[16] David Yarowsky and Richard Wicentowski. Minimally supervised morphological analysis by multimodal alignment. In K. Vijay-Shanker and Chang-Ning Huang, editors, *Proceedings of the 38th Meeting of the Association for Computational Linguistics*, pages 207–216, Hong Kong, October 2000.