Automatic linguistic annotation of historical language: ToTrTaLe and XIX century Slovene

Tomaž Erjavec
Department of Knowledge Technologies,
Jožef Stefan Institute
Jamova cesta 39, 1000 Ljubljana
Slovenia
tomaz.erjavec@ijs.si

Abstract

The paper describes a tool developed to process historical (Slovene) text, which annotates words in a TEI encoded corpus with their modern-day equivalents, morphosyntactic tags and lemmas. Such a tool is useful for developing historical corpora of highly-inflecting languages, enabling full text search in digital libraries of historical texts, for modernising such texts for today's readers and making it simpler to correct OCR transcriptions.

1 Introduction

Basic processing of written language, in particular tokenisation, tagging and lemmatisation, is useful in a number of applications, such as enabling full-text search, corpus-linguistic studies, and adding further layers of annotation. Support for lemmatisation and morphosyntactic tagging is well-advanced for modern-day languages, however, the situation is very different for historical language varieties, where much less – if any – resources exist to train high-quality taggers and lemmatisers. Historical texts also bring with them a number of challenges not present with modern language:

- due to the low print quality, optical character recognition (OCR) produces much worse results than for modern day texts; currently, such texts must be hand-corrected to arrive at acceptable quality levels;
- full-text search is difficult, as the texts are not lemmatised and use different orthographic conventions and archaic spellings, typically not familiar to non-specialists;
- comprehension can also be limited, esp. when the text uses an alphabet different from the contemporary norm.

This paper describes a tool to help alleviate the above problems. The tool implements a pipeline, where it first tokenises the text and then attempts to transcribe the archaic words to their modern day equivalents. For here on, the text is tagged and lemmatised using the models for modern Slovene. Such an approach is not new, as it straightforwardly follows from a situation where good language models are available for contemporary language, but not for its historical variants.

The focus of the research in such cases is on the mapping from historical words to modern ones, and such approaches have already been attempted for other languages, e.g. for English (Rayson et al. 2007), German (Pilz et al. 2008), Spanish (Sánchez-Marco et al. 2010) and Icelandic (Rögnvaldsson and Helgadóttir, 2008). These studies have mostly concentrated on mapping historical variants to modern words or evaluating PoS tagging accuracy and have dealt with Germanic and Romance languages. This paper discusses the complete annotation process, including lemmatisation, and treats a Slavic language, which has substantially different morphology; in Slovene, words belong to complex inflectional paradigms, which makes tagging and lemmatisation models quite complex, esp. for unknown words.

The paper also discusses structural annotations supported by the tool, which takes as input a document encoded according to (a subset of) the Text Encoding Initiative Guidelines, TEI P5 (Burnard and Bauman, 2007) and also produces output in this format.

An example of the tool input fragment and the corresponding output is given in Figure 1.

2 The ToTrTaLe tool

The annotation tool implements a pipeline architecture and is essentially a wrapper program that calls a number of further processing modules. The tool is based on the ToTaLe tool (Erjavec et al., 2005), which performs Tokenisation, Tagging and Lemmatisation on modern text; as the present tool extends this with Transcription, it is called ToTrTaLe, and comprises the following modules:

1. extracting processing chunks from source TEI
2. tokenisation
3. extracting text to be annotated
4. transcription to modern word-forms
5. part-of-speech tagging
6. lemmatisation
7. TEI output

While the tool and its modules make some language specific assumption, they are rather broad, such as that text tokens are (typically) separated by space; otherwise, the tool relies on external language resources, so it could be made to work with most European languages, although it is especially suited for the highly-inflecting ones.

The tool is written in Perl and is reasonably fast, i.e. it processes about 100k words per minute on a Linux server. The greatest speed bottleneck is the tool start-up, mostly the result of the lemmatisation module, which for Slovene contains thousands of rules and exceptions. In the rest of this section we present the modules of ToTrTaLe, esp. as they relate to processing of historical language.

2.1 Extracting chunks

In the first step, the top-level elements of the TEI file that contain text to be processed in one chunk are identified and passed on for linguistic processing. This step serves two purposes. Certain TEI elements, in particular the <teiHeader>, which contains the meta-data of the document, should not be analysed but simply passed on to the output (except for recording the fact that the text has been linguistically annotated). Second, the processors in certain stages keep the text and annotations in memory. As a TEI document can be arbitrarily large the available physical memory can be exhausted, leading to severe slow-down or even out-of-memory errors. It is therefore possible to specify which elements (such as <body> or <div>) should be treated as chunks to be processed in one annotation run.

2.2 The tokenisation module

The multilingual tokenisation module `mlToken`[1] is written in Perl and in addition to splitting the input string into tokens has also the following features:

- assigns to each token its token type, e.g. XML tag, sentence final punctuation, digit, abbreviation, URL, etc.
- preserves (subject to a flag) white-space, so that the input can be reconstituted from the output.

The tokeniser can be fine-tuned by putting punctuation into various classes (e.g. word-breaking vs. non-breaking) and also uses several language-dependent resource files, in particular a list of abbreviations (“words” ending in period, which is a part of the token and does not necessarily end a sentence), list of multi-word units (tokens consisting of several space-separated “words”) and a list of (right or left) clitics, i.e. cases where one “word” should be treated as several tokens. These resource files are esp. important in the context of processing historical language, as it often happens that words that used to be written apart and now written together or vice-versa. Such words are put in the appropriate resource file, so that their tokenisation is normalised. Examples of multi-word and split tokens are given in Figure 1.

2.3 Text extraction

A TEI encoded text can contain a fair amount of markup, which we, as much as possible, aim to preserve in the output. However, most of the markup should be ignored by the annotation modules, or, in certain cases, even the content of an element should be ignored; this goes esp. for markup found in text-critical editions of historical texts. For example, the top and bottom of the page can contain a running header, page number and catch-words (marked up in <fw> “forme work” elements), which should typically not be annotated as they are not linguistically interesting and would furthermore break the continuity of the text. The text might also contain editorial corrections (marked up as <choice> <sic>*mistyped text*</sic> <corr>*corrected text*</corr> </choice>), where, arguably, only the corrected text should be taken

[1] mlToken was written in 2005 by Camelia Ignat, then working at the EU Joint Research Centre in Ispra, Italy.

into account in the linguistic annotation. This module extracts the text that should be passed on to the annotation modules, where the elements to be ignored are specified in a resource file.

This solution does take care of most situations encountered so far in our corpora[2] but is not completely general. As discussed in Bennet et al. (2010), there are many cases where adding token (and sentence) tags to existing markup breaks XML well-formedness or TEI validity, such as sentences crossing structural boundaries or word-internal TEI markup.

A general "solution" to the problem is stand-off markup, where the annotated text is kept separate from the source TEI, but that merely postpones the problem of how to treat the two as a unit. And while TEI does offer solutions to such problems, implementing processing of arbitrary TEI in-place markup would, however, require much further research. So ToTrTaLe adds the linguistic mark-up in-place, but does so correctly only for a restricted, although still useful, set of TEI element configurations.

2.4 Transcription

The transcription of archaic word-forms to their modern day equivalents is the core module which distinguishes our processing of historical language as opposed to its contemporary form. The transcription process relies on three resources:

- a lexicon of modern-day word-forms;
- a lexicon of historical word-forms, with associated modern-day equivalent word-form(s);[3]
- a set of transcription patterns.

In processing historical texts, the word-form tokens are first normalised, i.e. de-capitalised and diacritic marks over vowels removed; the latter is most likely Slovene specific, as modern-day Slovene, unlike the language of the 19th century, does not use vowel diacritics.

[2] The notable exception is <lb/>, line break, which, given the large font size and small pages, often occurs in the middle of a word in historical texts. We move such line breaks in the source documents to the start of the word and mark their displacement in lb/@n.

[3] The two lexica have in fact a somewhat more complicated structure. For example, many archaic words do not have a proper modern day equivalent; for these, the lexicon gives the word in its modern spelling but also its modern near synonyms.

To determine the modern-day word-form, the historical lexicon is checked first. If the normalized word-form is an entry of the historical lexicon, the equivalent modern-day word-form has also been identified; if not, it is checked against the modern-day lexicon. This order of searching the lexica is important, as the modern lexicon can contain word-forms which have an incorrect meaning in the context of historical texts, so the historical lexicon also serves to block such meanings.

If neither lexicon contains the word, the transcription patterns are tried. Many historical spelling variants can be traced to a set of rewrite rules or "patterns" that locally explain the difference between the contemporary and the historical spelling. For Slovene, a very prominent pattern is e.g. *r→er* as exemplified by the pair *brž→berž*, where the left side represents the modern and the right the historical spelling.

Such patterns are operationalized by the finite-state "Variant aware approximate matching" tool Vaam, (Gotscharek et al. 2009; Reffle, 2011), which takes as input a historical word-form, the set of patters, and a modern-day lexicon and efficiently returns the modern-day word-forms that can be computed from the archaic one by applying one or more patterns. The output list is ranked, preferring candidates where a small number of pattern applications is needed for the rewrite operation.[4]

It should be noted that the above process of transcription is non-deterministic. While this rarely happens in practice, the historical word-form can have several modern-day equivalents. More importantly, the Vaam module will typically return several possible alternative modernisations, of which only one is correct for the specific use of the word in context. We currently make use of frequency based heuristics to determine the "best" transcription, but more advanced models are possible, which would postpone the decision of the best candidate until the tagging and lemmatization has been performed.

We currently use a set of about 100 transcription patterns, which were obtained by corpus inspection, using a dedicated concordancer.

[4] Vaam also supports approximate matching based on edit distance, useful for identifying (and correcting) OCR errors; we have, however, not yet made use of this functionality.

2.5 Tagging

For tagging words in the text with their context disambiguated morphosyntactic annotations we use TnT (Brants, 2000), a fast and robust tri-gram tagger. The tagger has been trained on jos1M, the 1 million word JOS corpus of contemporary Slovene (Erjavec and Krek, 2008), and is also given a large background lexicon extracted from the 600 million word FidaPLUS reference corpus of contemporary Slovene (Arhar and Gorjanc, 2007).

2.6 Lemmatisation

Automatic lemmatisation is a core application for many language processing tasks. In inflectionally rich languages assigning the correct lemma (base form) to each word in a running text is not trivial, as, for instance, Slovene adjectives inflect for gender, number and case (3x3x6) with a complex configuration of endings and stem modifications.

For our lemmatiser we use CLOG (Manandhar et al., 1998, Erjavec and Džeroski, 2004), which implements a machine learning approach to the automatic lemmatisation of (unknown) words. CLOG learns on the basis of input examples (pairs word-form/lemma, where each morphosyntactic tag is learnt separately) a first-order decision list, essentially a sequence of if-then-else clauses, where the defined operation is string concatenation. The learnt structures are Prolog programs but in order to minimise interface issues we made a converter from the Prolog program into one in Perl.

An interesting feature of CLOG is that it does not succeed in lemmatising just any word-form. With historical texts it almost invariably fails in lemmatising truly archaic words, making it a good selector for new entries in the historical lexicon.

The lemmatiser was trained on a lexicon extracted from the jos1M corpus, and the lemmatisation of contemporary language is quite accurate, with 92% on unknown words. However, as mentioned, the learnt model, given that there are 2,000 separate classes, is quite large: the Perl rules have about 2MB, which makes loading the lemmatiser slow.

2.7 TEI output

The final stage of processing is packing the original file with the added annotations into a valid TEI document. This is achieved by combining Perl processing with XSLT scripts. The last step in the processing is the validation of the resulting XML file against a TEI schema expressed in Relax NG. A validation failure indicates that the input document breaks some (possibly implicit) mark-up assumptions – in this case either the input document must be fixed, or, if the encoding choices were valid, the program should be extended to deal also with such cases.

3 Conclusions

The paper gave an overview of the ToTrTaLe tool, which performs basic linguistic annotation on TEI encoded historical texts. Some future work on the tool has already been mentioned, in particular exploring ways of flexibly connecting transcription to tagging and lemmatisation, as well as supporting more complex TEI encoded structures.

While the tool itself is largely language independent, it does need substantial language resources to operationalize it for a language. Specific for historical language processing are a corpus of transcribed historical texts, a lexicon of historical word forms and a pattern set. The paper did not discuss these language resources, although it is here that most work will be invested in the future.

The corpus we have used so far for Slovene lexicon building comes from the AHLib digital library (Prunč, 2007; Erjavec 2005), which contains 2 million words of 19th century texts; we now plan to extend this with older material, predominantly from the 18th century.

The on-going process of creating the Slovene historical lexicon is described in Erjavec et al., (2010), while the model of a TEI encoded lexicon containing not only historical word-forms, but also all the other lexical items needed to feed the tool (such as multi-word units) is presented in Erjavec et al. (2011). As we extend the corpus, we will also obtain new words, which will be automatically annotated with ToTrTaLe and then manually corrected, feeding into the lexicon building process.

For the patterns, the extension of the corpus will no doubt show the need to extend also the pattern set. Most likely this will be done by corpus inspection, via a dedicated concordancer, although alternative methods of pattern identification are possible. In particular, once when a substantial list of pairs historical word-form / contemporary word-form becomes available, automatic methods can be used to derive a list of patterns, ranked by how productive they are (Pilz et al., 2008; Oravecz et al. 2010).

Acknowledgements

The author thanks the anonymous reviewers for their useful comments and suggestions. The work presented in this paper has been supported by the EU IMPACT project “Improving Access to Text” and the Google Digital Humanities Research Award “Language models for historical Slovenian”.

References

Paul Bennett, Martin Durrell, Silke Scheible, and Richard J. Whitt, 2010. Annotating a historical corpus of German: A case study. *Proceedings of the LREC 2010 workshop on Language Resources and Language Technology Standards*. Valletta, Malta, 18 May 2010. 64-68.

Lou Burnard and Syd Bauman, 2007. *Guidelines for Electronic Text Encoding and Interchange (TEI P5)*. Text Encoding Initiative Consortium. Oxford, 2007. http://www.tei-c.org/release/doc/tei-p5-doc/

Tomaž Erjavec. 2007. Architecture for Editing Complex Digital Documents. Proceedings of the Conference on Digital Information and Heritage. Zagreb. pp. 105-114.

Tomaž Erjavec and Sašo Džeroski. 2004. Machine Learning of Language Structure: Lemmatising Unknown Slovene Words. *Applied Artificial Intelligence*, 18(1):17–41.

Tomaž Erjavec, Simon Krek, 2008. The JOS morphosyntactically tagged corpus of Slovene. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation, LREC’08*, Paris, ELRA.

Tomaž Erjavec, Camelia Ignat, Bruno Pouliquen, and Ralf Steinberger. Massive Multi-Lingual Corpus Compilation: Acquis Communautaire and ToTaLe. In Proceedings of the 2nd Language & Technology Conference, April 21-23, 2005, Poznan, Poland. 2005, pp. 32-36.

Tomaž Erjavec, Christoph Ringlstetter, Maja Žorga, and Annette Gotscharek, 2010. Towards a Lexicon of XIXth Century Slovene. In Proceedings of the Seventh Language Technologies Conference, October 14th-15th, 2010, Ljubljana, Slovenia. Jožef Stefan Institute.

Tomaž Erjavec, Christoph Ringlstetter, Maja Žorga, and Annette Gotscharek, (submitted). A lexicon for processing archaic language: the case of XIXth century Slovene. ESSLLI Workshop on Lexical Resources workshop, WoLeR’11. Ljubljana, Slovenia.

Annette Gotscharek, Andreas Neumann, Ulrich Reffle, Christoph Ringlstetter and Klaus U. Schulz. 2009. Enabling Information Retrieval on Historical Document Collections - the Role of Matching Procedures and Special Lexica. Proceedings of the ACM SIGIR 2009 Workshop on Analytics for Noisy Unstructured Text Data (AND09), Barcelona.

Suresh Manandhar, Sašo Džeroski and Tomaž Erjavec 1998. Learning Multilingual Morphology with CLOG. In Proceedings of Inductive Logic Programming; 8th International Workshop ILP-98 (Lecture Notes in Artificial Intelligence 1446) (pp. 135-144). Springer-Verlag, Berlin.

Csaba Oravecz, Bálint Sass and Eszter Simon. 2010. Semi-automatic Normalization of Old Hungarian Codices. Proceedings of the ECAI 2010 Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities (LaTeCH 2010), August 16, 2010, Lisbon, Portugal.

Thomas Pilz, Andrea Ernst-Gerlach, Sebastian Kempken, Paul Rayson and Dawn Archer, 2008. The Identification of Spelling Variants in English and German Historical Texts: Manual or Automatic? *Literary and Linguistic Computing*, 23/1, pp. 65-72.

Erich Prunč. 2007. Deutsch-slowenische/kroatische Übersetzung 1848-1918 [German-Slovene/Croatian translation, 1848-1918]. Ein Werkstättenbericht. Wiener Slavistisches Jahrbuch 53/2007. Austrian Academy of Sciences Press, Vienna. pp. 163-176.

Paul Rayson, Dawn Archer, Alistair Baron, Jonathan Culpeper, and Nicolas Smith, 2007. Tagging the Bard: Evaluating the accuracy of a modern POS tagger on Early Modern English corpora. In Proceedings of Corpus Linguistics 2007. University of Birmingham, UK.

Ulrich Reffle, Efficiently generating correction suggestions for garbled tokens of historical language, Journal of Natural Language Engineering, Special Issue on Finite State Methods and Models in Natural Language Processing, 2011.

Eiríkur Rögnvaldsson and Sigrún Helgadóttir, 2008. Morphological tagging of Old Norse texts and its use in studying syntactic variation and change. In Proceedings of the LREC 2008 Workshop on Language Technology for Cultural Heritage Data (LaTeCH 2008). ELRA, Paris.

Cristina Sánchez-Marco, Gemma Boleda, Josep Maria Fontana and Judith Domingo. 2010. Annotation and Representation of a Diachronic Corpus of Spanish. Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10). ELRA, Paris.

TEI input fragment:

```
  <p xml:id="p.401">Nekiga bogatiga kneza z nja <lb/>
    <pb n="93" facs="#FPG00012.097" xml:id="pb.97"/>
    družino, ki v mes nemore  <lb n="3"/>
    <gap/>
  </p>
```

ToTrTaLe output:

```
<p xml:id="p.401">
  <s>
    <w subtype="lexicon" nform="nekiga" mform="nekega" lemma="nek" ctag="Pi-msg">Nekiga</w>
    <c> </c>
    <w subtype="pattern" pattern="[ega@←iga@]" mform="bogatega" lemma="bogat"
        ctag="Agpmsg">bogatiga</w>
    <c> </c>
    <w lemma="knez" ctag="Npmsg">kneza</w>
    <c> </c>
    <w lemma="z" ctag="Si">z</w>
    <c> </c>
    <w subtype="lexicon" mform="njegova" lemma="njegov" ctag="Ps3fsnsm">nja</w>
    <c> </c>
    <lb/>
    <pb n="93" facs="#FPG00012.097" xml:id="pb.97"/>
    <w lemma="družina" ctag="Ncfsa">družino</w>
    <pc ctag=",">,</pc>
    <c> </c>
    <w lemma="ki" ctag="Cs">ki</w>
    <c> </c>
    <w type="multiw" subtype="pattern" pattern="[@v←@v_]" mform="vmes" lemma="vmes" ctag="Rgp"
         n="mw_jeGx2">v</w>
    <c> </c>
    <w type="multiw" subtype="pattern" pattern="[@v←@v_]" mform="vmes" lemma="vmes" ctag="Rgp"
        n="mw_jeGx2">mes</w>
    <c> </c>
    <w type="split" mform="ne_more" lemma="ne_moči" ctag="Q_Vmpr3s">nemore</w>
    <c>  </c>
    <lb n="3"/>
    <gap/>
  </s>
</p>
```

Figure 1. An example of ToTrTaLe input paragraph and the equivalent output.
Paragraphs, page and line breaks are preserved, and the program adds elements for words, punctuation symbols and white-space. Both punctuation and words are assigned a corpus tag and lemma, and, where different from the default, the type and subtype of the word, its normalised and modernised form, and possibly the used pattern(s). In cases of multi-words, each part is given its own word tag, which have identical analyses and are joined together by the unique value of @n; this approach allows also modelling discontinuous multi-word units, such as separable verbs in Germanic languages. Split words forms, on the other hand, are modelled by one word token, but with a portmanteau analysis.