Test data and testing of spelling checkers

Sjur Moshagen
UiT Norgga árktalaš universitehta
Uppsala, 13. november 2014

Content

Overview

- Many ways of testing spellers:
  - Counting red squiggles and suggestions
  - Automatically running corpora through the spell checker
- Presenting a project today:
  - manually collected error corpus
  - manually marked up for spelling errors
  - converted to a common xml format
  - used in an automated test bench

Overview 2

- Measures automatically:
  - precision, recall and accuracy
  - suggestion quality
- It does also facilitate research on spelling errors
- Supports five different speller engines
- Open source and portable

Test data

- To test one needs test data
- To do fast and repeated testing one needs automatisation
- To do automated testing one needs test data in a systematic and consistent format
- Thus, the Divvun group started building a test corpus that used a simple markup for identifying spelling errors, as part of the work to develop proofing tools for the Sámi languages
- This markup was further enhanced by Lene Antonsen in her work on L2 spelling errors, and later L1 errors

The Nordplus project

Goals:

- To build a corpus of running text containing at least 1000 spelling errors
- One such corpus each for Greenlandic, Icelandic, North, South and Julev Sámi
- Further modularise and improve portability of the speller test bench
- Test at least one speller for each language against the collected corpora
- Rebuild the test result browser
- To use the test results of the tested spellers to suggest improvements to the developers, and as a basis for further discussion and research on proofing tools

The error markup format

- The error markup format was based on the earlier work
- It differentiates between six markup types (≈error types) - the separator symbol is listed in parenthesis behind each type, its use will be explained on the following slides
  - unclassified error (§)
  - non-word error ($)
  - real-word error (¢)
  - morphosyntactic error (£)
  - syntactic error (¥)
  - lexical error (€)
  - foreign text & noise (∞)

Error markup syntax

- The syntax can be illustrated as follows:

... text (error)$(classification|correction) more text ...

- It is possible to nest the markup:

... text (de e girde (haleda)$(vowc,a-á|háleda))¥(wo|de girde e háleda) more text ...

- The last example illustrates the use of the separator symbol: it both separates the error from the correction, and informs the markup parser of what type of error is being marked up, to help in parsing and constraining the markup for easier detection of markup errors.
- Parentheses are used to define the scope of an error

More error markup syntax

- As we just saw, the markup is useful for much more than just marking up speller errors
- At the moment we only make use of non-word errors (\$) plus unclassified errors (§), which we treat as non-word spelling errors.
- During corpus markup, other errors have been marked up as well, to the extent possible within the given time constraints, but this markup is not checked, and there are probably many other types of errors that are not identified and marked up
- Still, the error corpora we have built should be quite valuable also for other usages than to test spelling checkers

Error markup in files and conversion to xml

- The error markup is manually added to copies of the original files
- Our corpus tools convert the markup and other textual features to an xml document. The error markup ends up like the following xml:

```
<errorort correct="låvdagoaden" errorinfo="vowlat,å-a">låvdågoaden</errorort>
```

- More information about the error markup syntax, and information on how errors are classified, can be found on `http://divvun.no/doc/proof/index.html`

Corpus sizes

Language	Total size	Nº of misspellings	% misspellings
Greenlandic (KAL)	75 508	1 904	2,52
Icelandic (ISL)	163 172	1 062	0,65
Julev Sámi (SMJ)	35 792	1 866	5,21
North Sámi (SME)	227 156	11 181	4,92
South Sámi (SMA)	181 701	16 390	9,02

Collecting and marking up the texts required 2-3 man months each for ISL, KAL and SMJ. SMA and SME could build on a lot of earlier work, and the invested time isn't directly comparable with the other languages.

Corpus processing tools

- Our corpus infrastructure contains one tool to convert documents in different formats to xml: `convert2xml`
- ... as well as a tool to extract different projections of the xml files for use in different contexts: `ccat`
- `ccat` has a number of features related to processing and printing of error markup
- Both tools are written in Python

Speller testing

- With proper test data available, we can test the available spellers
- For that we use our speller test bench

Architecture

- Configuration and portability: `Autotools`
- Multilingual support through the templating infrastructure by Divvun & Giellatekno
- Modular and extendable speller integration
- Modular processing & conversion of speller output

Running tests

- `./autogen.sh`
- `./configure`
- `make`
- The test bench can either use corpus data in our corpus repositories, or it can take as input a single, specified document
- The output is an xml file that can be inspected as is
- The xml file is also made available in a test result browser available on the net

Basic test results

Lang./speller	Prec.	Recall	Accuracy	Corr.1	Corr.1-5
ISL/Hunspell	12,13%	72,37%	96,51%	56,84%	67,66%
ISL/Púki	61,45%	61,45%	99,45%	39,64%	69,82%
KAL	15,45%	84,33%	87,48%	20,57%	22,37%
SMA/Word	83,05%	91,91%	94,63%	71,75%	79,98%
SME/Word	74,75%	86,62%	97,60%	61,82%	79,85%
SME/Hunspell	63,63%	84,43%	96,81%	60,45%	77,88%
SMJ/Word	69,10%	91,35%	97,01%	58,57%	72,63%

Browsing the test results

- An older version of the browser is available at `http://divvun.no/doc/proof/index.html`
- A new version is under development, both more flexible and more restricted
- More flexible:
  - Search for error types
  - Display suggestion quality data for specific error types
  - Easier comparison of spellers
- More limited:
  - Only True Positives are listed and available for inspection in the browser
  - This is to avoid the most obvious temptations to improve the speller by fixing the «errors» reported by the test. But: such fixes destroys the gold standard! Thus, we will not display that data.

Extending the test bench

- Since the test bench is modular, it is easy to extend it to support new speller engines
- Two steps are required:
  - A wrapper around the speller engine
  - A parser for the output produced by the speller
- The wrapper can be a single shell command, or it can be a complex script driving a graphical word processor
- Command-line spellers are very easily integrated
- Graphical word processors require much more work with less reliable results, and should be avoided if possible

Future work

- Add support for running on Windows and test MS Office spellers using VB (required for testing the MS spellers for e.g. Norwegian, Swedish and Danish)
- Add support for more speller engines
- Improve the test result browser
- Create error markup automatically from either pairs of files with unproofed and proofread text, or from e.g. MS Word's change tracking features
- Collect and mark up texts for some of the majority languages, in cooperation with the language councils
- Test more spellers for more languages
- Make a releasable version for others to use

Conclusion

- A nice set of error corpora for five languages
- An extendable test bench for spelling checkers
- A set of conventions for marking up errors in the original files
- A set of tools for converting this markup to xml and further process this xml
- Everything is open source, even parts of the test data
- Having everything open is problematic for the quality of future test results - it is very easy to cheat and thus destroy the gold standard
- The new test result browser makes better speller comparison
- The test bench could help in the general effort to improve reproducability for language technology research

Thanks

- Thanks to Nordplus Sprog for financing the project
- Thanks to Børre Gaup, Elin Neshamar, Hulda Óladóttir, Inga Lill Sigga Mikkelsen, Maja Kappfjell, Thomas Omma and Tomi Pieski for participating
- Thanks to Friðrik Skúlason for letting us evaluate the Icelandic speller Púki, and to Tino Didriksen for providing a command-line interface to the Greenlandic speller
- Thank you for listening!