Granska API – an Online API for Grammar Checking and Other NLP Services

Jonas Sjöbergh
Theoretical Computer Science, KTH
Stockholm, Sweden
jsh@kth.se

Viggo Kann
Theoretical Computer Science, KTH
Stockholm, Sweden
viggo@nada.kth.se

Abstract

We present an online API to access many Natural Language Processing services developed at KTH. The services work on Swedish text. They include tokenization, part-of-speech tagging, shallow parsing, compound word analysis, word inflection, lemmatization, spelling error detection and correction, grammar checking, and more. The services can be accessed in several ways, including a RESTful interface, direct socket communication, and pre-made Web forms. The services are open to anyone. The source code is also freely available making it possible to setup another server or run the tools locally.

1 Introduction

Several Natural Language Processing (NLP) tools for analysis of Swedish text have been developed at KTH (Kann, 2010). Most of the tools were developed in projects focused on grammar checking, so the tools also have a focus on tools for grammar checking, but many low level language analysis tools useful in other applications are also included.

The source code for the tools has been freely available since the tools were developed, and anyone is free to install and use them locally. Now we have also made an online API available. It can be used to access the tools running as services on a server at KTH. These services are also open for anyone to use. They can be used by a user by hand and by programs using the services to do some analysis they need.

We have also built an example application that uses the services to create a graphical text exploration environment.

2 Available Services

The available services can be divided into three types of services: low-level or pre-processing NLP tools that can be used to build more advanced NLP services, tools to help when developing and evaluating other tools, and high-level NLP services that are directly useful to end users. It is still possible to build new tools on top of the high-level services.

The low-level services in the Granska API are:

Tokenization The Granska tokenizer tokenizes text into words and sentences. It is integrated in the Granska tagger and Granska grammar checker below, but it is also possible to build a stand-alone tokenizer.

PoS Tagging The Granska tagger (Carlberger and Kann, 1999) does part-of-speech tagging of Swedish text. It is a Hidden Markov Model tagger trained on the SUC corpus (Ejerhed et al., 1992) using a slightly modified version of the SUC tag set. The tagger is integrated in the Granska grammar checker but can also be a stand-alone application.

PoS Tagging without context Taggstava (Kann, 2010) is a tagger that assigns part-of-speech tags to words without using context information. It uses inflection rules for Swedish to determine what inflected form a word could be. No disambiguation is done for ambiguous words. Taggstava uses the same rules and reference data as Stava below.

Shallow Parsing The GTA parser (Knutsson et al., 2003) does shallow parsing of Swedish text based on hand written rules. It identifies clause boundaries and phrases. The internal structures of phrases are identified, e.g. a noun phrase being part of a prepositional phrase, but a full tree for the whole sentence is not built.

GTA is built to be robust to noisy data (i.e. text with many errors) since it is built for and used in the grammar checker Granska below, which is expected to run on texts with possibly very many errors in them.

For convenience, there are also two services that return subsets of the GTA information, one that returns only clause boundaries, and one for the phrase structure.

Compound Word Analysis SärStava (Sjöbergh and Kann, 2004) is a tool that gives the most likely interpretation of a compound word, or all possible interpretations, using the Stava compound word analysis methods. It uses statistical data and some heuristics to decide which interpretation is most likely for ambiguous compounds but does not use the context of the word.

Word Inflection The Granska Inflector inflects Swedish words and can generate lists of possible inflections.

Lemmatization This service uses the Granska tagger to find the lemma form of words.

Word-Tag-Lemma Since several other services expect the input to be triples of word, part-of-speech tag, and lemma form of the word, for convenience there is also a service that takes plain text and provides word-tag-lemma triples.

There is currently only one service in the development and evaluation tools category. Other tools for evaluating NLP tools are available to run locally, but have not been made available as online services yet. The available tool is:

Realistic Spelling Error Generation Missplel (Bigert et al., 2003) is a tool that automatically inserts spelling errors in texts. Different types of errors can be simulated, for example keyboard mistypes where a neighboring key is pressed or sound-alike errors where the writer may not know the correct spelling of a known word.

Missplel can be used to automatically evaluate the robustness of other NLP systems by showing how the performance degrades when there are errors in the text. For example, running a parser on a test text and then running it on the same text with added errors. Ideally, the parser should produce similar output the second time, since the "intended" meaning of the text is the same.

The high-level services available are:

Spelling Error Detection and Correction Stava (Domeij et al., 1994) is a very powerful spelling correction tool for Swedish that finds spelling errors and suggests corrections. Stava handles the very productive compounding in Swedish (it is very common to create new compound words) using rules for how compounds can and cannot be created in Swedish. The compound analysis can also be accessed separately, as mentioned above.

Grammar Checking using Rules The Granska (Domeij et al., 2000) system detects grammatical errors in Swedish text based on manually written error detection rules. The rule language (Knutsson et al., 2001) is quite powerful and the rule writer has access to all the information provided by the tools mentioned above. Rules can for example be written to allow suspicious things if they cross a phrase boundary, or to change the inflected form of a suspicious word to a form more suitable to the surrounding context using the inflector above.

Extra rules can be added for each call. These can be used to detect types of errors not covered by the standard rules or to influence the behavior of Granska (e.g. by adding more parsing rules). Simple example rule:

```
altcorr@kong{
 X(wordcl=dt),
 Y(wordcl=nn & num!=X.num)
  -->
 corr(X.form(num:=Y.num))
 corr(Y.form(num:=X.num))
 action(scrutinizing)}
```

This rule finds places where a determiner (word class is "dt") is followed by a noun, but they have different numerus (i.e. agreement errors). It then suggests two possible corrections, changing the numerus of the determiner or of the noun. The suggested corrections are generated with the inflector above.

Grammar Checking using PoS n-grams ProbCheck (Bigert and Knutsson, 2002) detects grammatical errors in text using statistical analysis of part-of-speech n-grams,

Goal: get the most likely interpretation of the compound “språkteknologi”.
API call: `https://skrutten.csc.kth.se/granskaapi/compound/best/språkteknologi`
Output: `språkteknologi språk|teknologi`

Goal: get all possible interpretations of the compound “språkteknologi”, in JSON.
API call: `https://skrutten.csc.kth.se/granskaapi/compound/json/all/språkteknologi`
Output: `[{"word":"språkteknologi","parts":["språk|teknologi","språk|tekno|logi"]}]`

Goal: get phrase structure in the sentence “GTA kan analysera svensk text.”.
API call: `http://skrutten.csc.kth.se/granskaapi/chunk?text=GTA+kan+analysera+svensk+text+.`
Output: `GTA NPB, kan VCB, analysera VCI, svensk APMINB|NPB, text NPI, .  0`

Figure 1: Example API calls and the corresponding outputs

based on n-gram statistics from correct text. It also uses the GTA parser above, since phrase and clause boundaries can cause very rare PoS n-grams even in correct text and thus lead to false alarms. ProbCheck usually runs integrated in Granska but running only ProbCheck is also possible.

ProbCheck was created in a project focused on helping second language learners. Learners of a language make many unpredictable errors that it can be hard to write error detection rules for. There are also generally a lot of errors, and thus not much correct text as context to base the rules on.

Grammar Checking using Machine Learning SnålGranska (Sjöbergh and Knutsson, 2005) detects grammatical errors using machine learning trained on texts with synthetic errors added. By itself it does not perform as well as Granska, but it does detect errors that Granska does not detect, and it is possible to use both systems together to get improved coverage (Bigert et al., 2004).

3 Ways to Access the Services

All the services mentioned in the previous section can be accessed online[1]. There are simple Web forms where you can enter words or text by hand (or by copy-paste) and see what the tools can do.

There is also a RESTful API to access the services. This allows typing in requests in the URL bar of a Web browser by hand, but is mainly intended for other programs to automatically use the services for some processing they may need. Most services can send back the reply in either plain text form, HTML, JSON, or XML. Figure 1 shows example API calls and the corresponding outputs.

[1]https://skrutten.csc.kth.se/granskaapi/

It is also possible to access the services using socket communication. When communicating with the services directly using a socket, most services will only return the raw output of the original tool (for example not provide the result in JSON).

Each service will display a Web page with information regarding how to call the service if no input is given. An example Web form that uses the service is shown, and this can be used as a reference to see what input is expected. Normally a few example words or sentences are also provided to give a quick overview of what the output can be expected to look like.

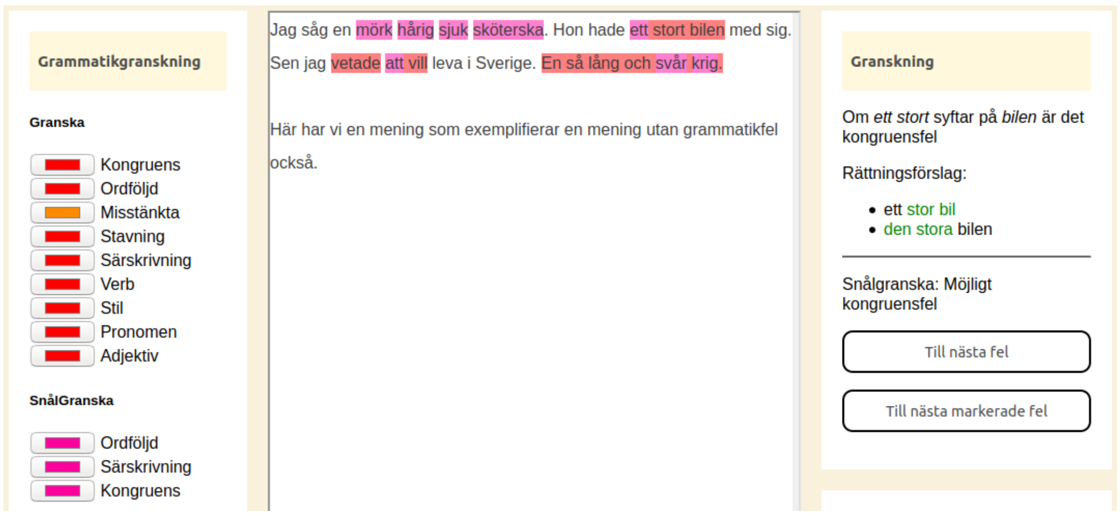
The API allows building new tools based on the services, creating new interfaces to the services, or integrating the services into existing tools (e.g. an editor or word processor). If an online service is not suitable, for example for a system that is expected to run offline, the source code for all the tools is also freely available. This also makes it possible to install any tool and run it locally, or to install tools and set up a new server that can provide the same services.

4 Example Application Built on the Services

We have created an example application using many of the services described above. FörhandsGranska[2] is a graphical text exploration tool. It can mark writing errors in different colors and suggest corrections, working as an editor with built in spelling and grammar checking tools.

It can also add linguistic markup, coloring words based on their part-of-speech, underline different types of phrases in different colors, or show clause boundaries. It also shows all inflections of a word, the compound analysis of compound words, and more. In this way, it can be used as a linguistic exploration tool or language learning

[2]https://skrutten.csc.kth.se/fhg/


Grammatikgranskning

Granska

- 🟥 Kongruens
- 🟥 Ordföljd
- 🟧 Misständkta
- 🟥 Stavning
- 🟥 Särskrivning
- 🟥 Verb
- 🟥 Stil
- 🟥 Pronomen
- 🟥 Adjektiv

SnålGranska

- 🟪 Ordföljd
- 🟪 Särskrivning
- 🟪 Kongruens

Jag såg en mörk hårig sjuk sköterska. Hon hade ett stort bilen med sig. Sen jag vetade att vill leva i Sverige. En så lång och svår krig.

Här har vi en mening som exemplifierar en mening utan grammatikfel också.

Granskning

Om *ett stort* syftar på *bilen* är det kongruensfel

Rättningsförslag:

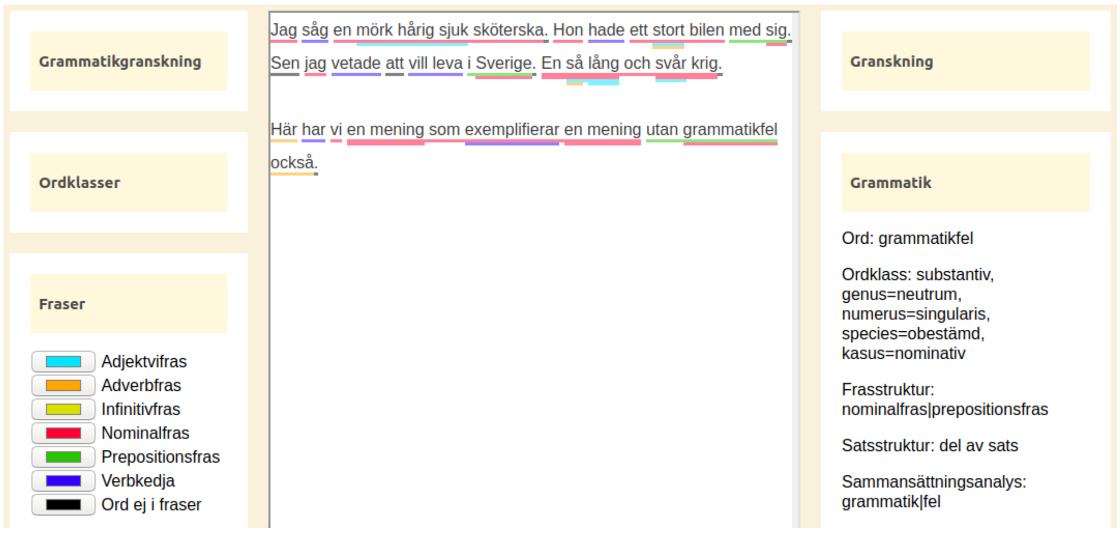
- ett stor bil
- den stora bilen

Snålgranska: Möjligt kongruensfel

Till nästa fel

Till nästa markerade fel

Figure 2: FörhandsGranska, built on top of the API services. Here showing grammar checking support, letting the user use suggested corrections through simple clicks.


Grammatikgranskning

Ordklasser

Fraser

- 🟦 Adjektivfras
- 🟧 Adverbfras
- 🟨 Infinitivfras
- 🟥 Nominalfras
- 🟩 Prepositionsfras
- 🟦 Verbkedja
- ⬛ Ord ej i fraser

Jag såg en mörk hårig sjuk sköterska. Hon hade ett stort bilen med sig. Sen jag vetade att vill leva i Sverige. En så lång och svår krig.

Här har vi en mening som exemplifierar en mening utan grammatikfel också.

Granskning

Grammatik

Ord: grammatikfel

Ordklass: substantiv, genus=neutrum, numerus=singularis, species=obestämd, kasus=nominativ

Frasstruktur: nominalfras|prepositionsfras

Satsstruktur: del av sats

Sammansättningsanalys: grammatik|fel

Figure 3: FörhandsGranska, linguistic analysis: PoS, phrase structure, compound analysis, inflections, etc.

tool. Adding more rules in the Granska rule language is also supported. Two example screenshots of FörhandsGranska are shown in Figures 2 and 3.

FörhandsGranska is written in JavaScript and is mostly a graphical interface that calls the services of the Granska API for any language analysis.

5 Related Work

There are other NLP APIs, both online and APIs for using tools locally. Most APIs are for English but APIs for other languages are also available.

For Swedish, the Sparv corpus annotation pipeline (Borin et al., 2016) has an online API. It supports tokenization, lemmatization, part-of-speech tagging, compound analysis, dependency parsing, named entity recognition, and more. Sparv also supports languages other than Swedish.

The SVENSK project (Gambäck and Olsson, 2000) collected NLP tools for Swedish, including part-of-speech tagging, parsing, text classification, and more. Resources from different sources were integrated into one consistent framework using GATE (Cunningham et al., 1996).

6 Conclusions

We provide an online API to access NLP services for Swedish text. The services are freely available online, with several ways to access them. The source code is also freely available, allowing users to set up their own servers or run the tools locally. The tools can be used by hand or integrated in other programs.

References

Johnny Bigert, Linus Ericson, and Antoine Solis. 2003. Missplel and AutoEval: Two generic tools for automatic evaluation. In *Proceedings of Nodalida 2003*, Reykjavik, Iceland.

Johnny Bigert, Viggo Kann, Ola Knutsson, and Jonas Sjöbergh. 2004. Grammar checking for Swedish second language learners. In Peter Juel Henrichsen, editor, *CALL for the Nordic Languages*, pages 33–47. Samfundslitteratur.

Johnny Bigert and Ola Knutsson. 2002. Robust error detection: A hybrid approach combining unsupervised error detection and linguistic knowledge. In *Proceedings of Romand 2002, Robust Methods in Analysis of Natural Language Data*, pages 10–19, Frascati, Italy.

Lars Borin, Markus Forsberg, Martin Hammarstedt, Dan Rosén, Roland Schäfer, and Anne Schumacher. 2016. Sparv: Språkbanken's corpus annotation pipeline infrastructure. In *Proceedings of SLTC 2016*, Umeå, Sweden.

Johan Carlberger and Viggo Kann. 1999. Implementing an efficient part-of-speech tagger. *Software – Practice and Experience*, 29(9):815–832.

Hamish Cunningham, Yorick Wilks, and Robert J. Gaizauskas. 1996. GATE – a general architecture for text engineering. In *COLING 1996 Volume 2: The 16th International Conference on Computational Linguistics*.

Richard Domeij, Ola Knutsson, Johan Carlberger, and Viggo Kann. 2000. Granska – an efficient hybrid system for Swedish grammar checking. In *Proceedings of Nodalida '99*, pages 49–56, Trondheim, Norway.

Rickard Domeij, Joachim Hollman, and Viggo Kann. 1994. Detection of spelling errors in Swedish not using a word list en clair. *Journal of Quantitative Linguistics*, 1:195–201.

Eva Ejerhed, Gunnel Källgren, Ola Wennstedt, and Magnus Åström. 1992. The linguistic annotation system of the Stockholm-Umeå Corpus project. Technical report, Department of General Linguistics, University of Umeå (DGL-UUM-R-33), Umeå, Sweden.

Björn Gambäck and Fredrik Olsson. 2000. Experiences of language engineering algorithm reuse. In *Proceedings of the Second International Conference on Language Resources and Evaluation (LREC'00)*, Athens, Greece.

Viggo Kann. 2010. KTHs morfologiska och lexikografiska verktyg och resurser (Morphological and lexicographical tools and resources from KTH). *LexicoNordica*, 17:99–117. QC 20120126.

Ola Knutsson, Johnny Bigert, and Viggo Kann. 2003. A robust shallow parser for Swedish. In *Proceedings of Nodalida 2003*, Reykjavik, Iceland.

Ola Knutsson, Johan Carlberger, and Viggo Kann. 2001. An object-oriented rule language for high-level text processing. In *NoDaLiDa'01 - 13th Nordic Conference on Computational Linguistics*, Uppsala, Sweden.

Jonas Sjöbergh and Viggo Kann. 2004. Finding the correct interpretation of Swedish compounds a statistical approach. In *Proceedings of LREC-2004*, pages 899–902, Lisbon, Portugal.

Jonas Sjöbergh and Ola Knutsson. 2005. Faking errors to avoid making errors: Very weakly supervised learning for error detection in writing. In *Proceedings of RANLP 2005*, pages 506–512, Borovets, Bulgaria.