Combining a REST Lexical Analysis Web Service with SPARQL for Mashup Semantic Annotation from Text

Eetu Mäkelä

Semantic Computing Research Group (SeCo),
Aalto University and University of Helsinki, Finland
`eetu.makela@aalto.fi, http://www.seco.tkk.fi/`

Abstract. Current automatic annotation systems are often monolithic, holding internal copies of both machine-learned annotation models and the reference vocabularies they use. This is problematic particularly for frequently changing references such as person and place registries, as the information in the copy quickly grows stale. In this paper, arguments and experiments are presented on the notion that sufficient accuracy and recall can both be obtained simply by combining a sufficiently capable lexical analysis web service with querying a primary SPARQL store, even in the case of often problematic highly inflected languages.

1 Introduction

The context of the current work is that as part of a national Semantic Web infrastructure for Finland [1], a service for extracting automatic semantic annotations from texts was desired. There are already many tools for such, falling into different categories based on which languages they support, the types of entitities they recognize and if they are bound to a particular reference vocabulary, or even use any vocabulary at all[1]. Unfortunately, the requirements for our service ruled out all of the existing candidates. First, the service would have to support at least Finland's two official languages of Finnish and Swedish. Second, it should allow for picking keywords from any of the many general keyword vocabularies [3] used in Finland, as well as the larger national person, place and event instance registries currently being created.

To solve this problem, a first iteration of a novel automatic indexing service ARPA was created [4], based on the Maui indexing tool [5], which could combine an arbitrary language processor, vocabulary and training corpus into a human-competitive automated indexer web service. This worked well, but caused problems in maintenance, because any update to the vocabulary, training data or lemmatizer needed a new manual packaging of the service. While this was

[1] For the purposes of this paper, a good overview of the field is given in the related work section of [2].

sufferable for the seldom-changing keyword vocabularies, it created insurmountable problems for the much larger instance registries, which constantly update to add new people, places and events.

To counteract the problem, sights were set on modularizing the system, of which good results had been previously obtained on other parts of the national infrastructure stack [6]. A particular goal was for the system to not have to maintain a local, stale copy of the vocabulary used, but to be able to e.g. query the master SPARQL endpoint of the instance registries for matches.

2 Requirements for Modular Semantic Annotation

Generally, semantic annotation can be thought of as being composed of two phases [2,5]. First, in a phrase spotting or candidate selection phase, possible annotations are extracted from the text. Then, in a disambiguation and selection phase, the candidates are compared and some are selected, others discarded.

For disambiguation, it turns out that the simple algorithm of *always selecting the concept that already appears most often in annotations* nearly matches the accuracy of more complex methods. In [7] for example, differences in accuracy ranged only from 0.002 to 0.024 for 7 different languages tested. Thus, there didn't seem to be enough added benefit in teaching and re-teaching the naive Bayes -based Maui classifier for concept selection. Instead, as this is such an easy to implement measure and benefits from access to an up-to-date version of the dataset being added to, it was decided that this functionality would not be implemented at all, with the task given over to the end user system.

As for the candidation selection phase, what is required depends on the language. For weakly inflected languages such as English, Swedish or Dutch, where word forms are seldom modified to respond to grammatical structure, even a completely language ignorant naive approach functions well. As an example for Dutch, merely *selecting any exact phrase matching a concept in the vocabulary* resulted in virtually equivalent recall to an implementation utilizing NLP processing (55.01% vs 55.53% in [7]). Thus, for automatic indexing of Swedish, adequate functionality could have been obtained merely by enumerating all n-grams in the incoming text, and then issuing label match queries to the master up-to-date SPARQL endpoints of the vocabularies and instance registries.

However, for highly inflected languages such as Russian or Hungarian, language ignorant recall in phrase spotting is considerable lower (30.62% and 34.07% respectively in [7]). Unfortunately, Finnish is a highly inflected language, where e.g. the noun for shop, "kauppa", can appear in a total of 2,253 different forms depending on the sentence[2]. For good recall in such languages it is essential to utilize lemmatization, whereby each word is transformed into its base form [4].

Thus it was decided that our new automatic indexing service would be composed of two components: first, a lexical analysis service would lemmatize the text into baseforms, and then a simple querying component would use the resulting n-grams to query a (SPARQL) web service for matching concepts.

[2] `http://www.ling.helsinki.fi/~fkarlsso/genkau2.html`

3 The Need for Morphological Analysis and Inflected Form Generation

Decoupling the vocabulary from the lemmatization service caused some new problems, however. In the earlier Maui implementation, where the vocabularies were loaded into an internal model, the indexer could also lemmatize the terms in the vocabulary for easy matching. With the vocabulary outside the indexer, this was no longer possible.

While in most cases, words in the reference vocabularies to be used are already in their nominal base form, there are still enough notable exceptions to cause problems. First, the most important Finnish vocabulary, the National Finnish General Thesaurus YSA, contains nouns in their plural form instead of singular (e.g. "presidentit" [presidents] instead of "presidentti" [president]). Second, for applications needing to index also verbs, they are often in their nominative form (e.g. both YSA and Wikipedia contain "lentäminen" [flying] instead of "lentää" [to fly]). Finally and most importantly for compound phrases, not all words turn into their baseforms. For example, the Foreign ministry of Finland has a base form (and a Wikipedia page) of "Suomen ulkoministeriö" instead of the form "Suomi ulkoministeriö", which a naive lemmatization algorithm would turn out.

Because of this, it was deemed that the lexical analysis service should also 1) support deeper morphological analysis of the text in order to be able to flexibly handle compound phrases and 2) support the generation of any inflected form instead of just the baseform to handle any quirks of the target vocabularies.

Luckily, there existed ready tools for this, in the form of the Helsinki Finite State Transducer toolkit [8] and accompanying transducers for multiple languages interesting from the Finnish standpoint (including for example the Finnish minority language of Sami). All that was needed was to package these tools together in the form of an easy to use web service.

4 The SeCo Lexical Analysis Service

The end result produced by the packaging is the SeCo Lexical Analysis Web Service at `http://demo.seco.tkk.fi/las/`, with source code available at `http://github.com/jiemakel/seco-lexicalanalysis-play`. All in all, the service is comprised of five functionalities:

1. Language recognition for a total of 95 languages, based on three sources: 1) the langdetect library [9], 2) own custom code and 3) finite state transducers from the HFST [8], Omorfi [10] and Giellatekno [11] projects.
2. Lemmatization for a total of 20 languages, utilizing again the finite state transducers from the HFST, Omorfi and Giellatekno projects, with a fallback to Snowball [12] stemmers.
3. More complete morphological analysis is available for the 14 languages fully supported by the finite state transducers.
4. Inflected form generation is likewise available for the same 14 languages.

5. Hyphenation based on finite state transducers is available for 46 languages

All functionalities are available as RPC-style web services, supporting both the HTTP and WebSocket protocols. All services are additionally CORS-enabled and return results in JSON for easy integration into HTML5 web applications. Further details as well as live examples are available at the service itself.

5 ARPA automatic annotation service

As after the lexical processing the actual querying for semantic annotations is relatively simple, a demonstration of this was implemented as a static HTML5 Javascript application at `http://demo.seco.tkk.fi/sarpa/`.

The application is comprised of a text field and a series of controls by which it is possible to change parameters of the lexical analysis process, as well as specify an arbitrary SPARQL endpoint and query for fetching candidate annotations. There are also four different complete pre-configured examples to select from demonstrating the various options and functionalities. One targets the Finnish edition of DBPedia while another targets the YSA thesaurus. The remaining two target the public SPARQL endpoint of DBPedia with different query restrictions.

As a complete example, consider the analysis[3] of the Finnish sentence "Erkki Tuomiojan mukaan Suomen ulkoministeriön tietomurtoa käsiteltiin tasavallan presidentin Sauli Niinistön kanssa heti tämän lennettyä Helsinkiin" (According to Erkki Tuomioja, the data system break-in at the Ministry for Foreign Affairs of Finland was talked over with president Sauli Niinistö as soon as he had flown to Helsinki). Run with a configuration targeting the SPARQL endpoint of the Finnish edition of DBPedia, this results in finding the pages for "Erkki Tuomioja", "Suomen ulkoministeriö" (the Ministry of Foreign Affairs of Finland), "tietomurto" (data system break-in), "presidentti" (president), "Sauli Niinistö", "lentäminen" (flying) and "Helsinki". Notable here is that first part of the compound word "Suomen ulkoministeriö" is still inflected, and the verb "lentäminen" is in its nominative form, not the base form. On the other hand, against the General Finnish Thesaurus, the concepts "tietomurto", "presidentit" (notice the plural form version of the word) and "lentäminen" (again notice the nominative form of the verb) are found.

6 Conclusions

Based on the analysis and experience presented here, it is easily possible to create lightweight automatic semantic annotation systems just by combining a lexical analysis service with a standard vocabulary query interface. In addition, from analysis and prior experiments it seems that such systems can approach the accuracy and recall levels provided by more complex annotators, although this requires further experimentation to conclusively decide.

[3] `http://j.mp/1cBiBvL`

In addition to proving the basic premise, this work also highlighted some dirty details that must be taken into account when attempting the creation of such a system for highly inflected languages. In this work, these were surmounted for the automatic indexing of Finnish material by making use of more thorough morphological analysis of the text, as well as inflected form generation.

References

1. Hyvönen, E.: Developing and Using a National Cross-Domain Semantic Web Infrastructure. In: Semantic Computing. John Wiley & Sons, Inc. (2010) 421–438
2. Mendes, P.N., Jakob, M., García-Silva, A., Bizer, C.: Dbpedia spotlight: shedding light on the web of documents. In Ghidini, C., Ngomo, A.C.N., Lindstaedt, S.N., Pellegrini, T., eds.: I-SEMANTICS. ACM International Conference Proceeding Series, ACM (2011) 1–8
3. Frosterus, M., Tuominen, J., Pessala, S., Seppälä, K., Hyvönen, E.: Linked open ontology cloud koko - managing a system of cross-domain lightweight ontologies. In Cimiano, P., Fernández, M., Lopez, V., Schlobach, S., Völker, J., eds.: ESWC (Satellite Events). Volume 7955 of Lecture Notes in Computer Science., Springer (2013) 296–297
4. Sinkkilä, R., Suominen, O., Hyvönen, E.: Automatic semantic subject indexing of web documents in highly inflected languages. In Antoniou, G., Grobelnik, M., Simperl, E.P.B., Parsia, B., Plexousakis, D., Leenheer, P.D., Pan, J.Z., eds.: ESWC (1). Volume 6643 of Lecture Notes in Computer Science., Springer (2011) 215–229
5. Medelyan, O.: Human-competitive automatic topic indexing. PhD thesis, The University of Waikato (2009)
6. Mäkelä, E., Viljanen, K., Alm, O., Tuominen, J., Valkeapää, O., Kauppinen, T., Kurki, J., Sinkkilä, R., Kansala, T., Lindroos, R., Suominen, O., Ruotsalo, T., Hyvönen, E.: Enabling the semantic web with ready-to-use web widgets. In Nixon, L.J.B., Cuel, R., Bergamini, C., eds.: FIRST. Volume 293 of CEUR Workshop Proceedings., CEUR-WS.org (2007) 56–69
7. Daiber, J., Jakob, M., Hokamp, C., Mendes, P.N.: Improving efficiency and accuracy in multilingual entity extraction. In Sabou, M., Blomqvist, E., Noia, T.D., Sack, H., Pellegrini, T., eds.: I-SEMANTICS, ACM (2013) 121–124
8. Lindén, K., Axelson, E., Hardwick, S., Pirinen, T.A., Silfverberg, M.: Hfst – framework for compiling and applying morphologies. In Mahlow, C., Piotrowski, M., eds.: SFCM. Volume 100 of Communications in Computer and Information Science., Springer (2011) 67–85
9. Shuyo, N.: Language detection library for java. `http://code.google.com/p/language-detection/` (2010)
10. Pirinen, T.A.: Modularisation of finnish finite-state language description – towards wide collaboration in open source development of morphological analyser. In: Proceedings of Nodalida. Volume 18 of NEALT proceedings. (2011)
11. Moshagen, S.N., Pirinen, T.A., Trosterud, T.: Building an open-source development infrastructure for language technology projects. In: Proceedings of the 19th Nordic Conference of Computational Linguistics (NODALIDA 2013). Volume 16 of NEALT Proceedings Series. (May 22–24 2013)
12. Porter, M.F.: Snowball: A language for stemming algorithms. `http://snowball.tartarus.org/texts/introduction.html` (October 2001)