HFST-SweNER — A New NER Resource for Swedish

Kokkinakis, Dimitrios

European Language Resources Association (ELRA)

2014-05-26


http://hdl.handle.net/10138/136265

Downloaded from Helda, University of Helsinki institutional repository.

This is an electronic reprint of the original article.

This reprint may differ from the original in pagination and typographic detail.

Please cite the original version.
HFST-SweNER – A New NER Resource for Swedish

Dimitrios Kokkinakis§, Jyrki Niemi§, Sam Hardwick‡, Krister Lindén‡, Lars Borin§

§Språkbanken, Department of Swedish University of Gothenburg, Sweden E-mail: first.last@svenska.gu.se
‡Department of Modern Languages University of Helsinki, Finland E-mail: first.last@helsinki.fi

Abstract

Named entity recognition (NER) is a knowledge-intensive information extraction task that is used for recognizing textual mentions of entities that belong to a predefined set of categories, such as locations, organizations, and time expressions. NER is a challenging, difficult, yet essential preprocessing technology for many natural language processing applications, and particularly crucial for language understanding. NER has been actively explored in academia and in industry especially during the last years due to the advent of social media data. This paper describes the conversion, modeling and adaptation of a Swedish NER system from a hybrid environment, with integrated functionality from various processing components, to the Helsinki Finite-State Transducer Technology (HFST) platform. This new HFST-based NER (HFST-SweNER) is a full-fledged open source implementation that supports a variety of generic named entity types and consists of multiple, reusable resource layers, e.g., various n-gram-based named entity lists (gazetteers).

Keywords: named entity recognition, finite-state transducers, Swedish

1. Introduction

Named entity recognition (NER) is a fundamental natural language processing (NLP) technology that aims at the automatic resolution of lexical ambiguity on the level of so-called named entities (Sekine and Ranchhod, 2009). This is a task that is performed by locating, extracting and classifying named entities into predefined target named entity classes in unstructured text. In many cases, such named entities can be specific to a particular goal and domain (Ananiadou et al., 2004; Dozier et al., 2009; Wang 2009), but in the general case, NER systems are concerned with the identification and classification of names into a set of predefined generic classes. There are three universally accepted categories of such classes recognized, namely persons, organizations and locations. This set is usually used and supplemented with several other entity types, at least, such as date/time expressions and numerical expressions (e.g. quantities, monetary values, percentages) as well as domain-specific entities (e.g. bibliographic references, names of diseases or names of transportation means).

Most importantly, NER is, in principle, a prerequisite step for numerous advanced processing tasks, including ontology population (i.e., creating instances of particular concepts in a given ontology; Giuliano and Gliozzo, 2008), relation extraction (i.e., by creating structured knowledge bases with ontological relationships; Suchanek et al. 2006); text classification (Kumaran and Allan, 2004); question answering (i.e., many fact-based answers to questions are entities that can be detected by a NER; Mollá et al., 2007) or machine translation (Nikoulina et al., 2012). However, polysemy and synonymy, including variation of named entities (e.g. “John Smith”, “Mr Smith”, “John”), ambiguities between entities (e.g. “Ericsson” as a person vs. a company), ambiguities between entities and common words (e.g. “Ingå” as a proper noun vs. a pronoun in Swedish), nesting (e.g. “Bank of [New York]”) and several other problems, have clearly contributed to the fact that, after several decades of research in the area (Grishman and Sundheim, 1996) and the substantial efforts that have been made in the area of NER, the task is actually far from solved (Marrero et al., 2013). Nevertheless, recent NER approaches achieve high F-scores (90.8%; Ratinov and Roth, 2009) on standard data sets, such as the one developed for the CoNLL-2003 NER shared task (Tjong Kim Sang and de Meulder, 2003).

2. Named Entity Recognition for Swedish

In recent years, a small number of NER systems have also emerged in the Swedish research terrain. Kokkinakis (1998) describes a rule-based system that recognizes nine entity types, both generic ones (persons, organizations, geographical places and time sequences) and also some domain-dependent ones in the area of “drug enforcement” (money expressions, transportation and communication means, social places and narcotic substances). Dalianis and Åström (2001) present a NER system, SweNam, that employs rules, lexicons, and machine learning techniques for the recognition and classification of four entity types: persons, locations, organizations and time. Kokkinakis (2004) describes a further improvement and enhancement of the rule-based system described earlier, which comprises a number of modules organized into layers and applied sequentially, in a pipeline fashion.

Borin and Kokkinakis (2010) further describe the application and adaptation of the aforementioned NER system on Swedish classical literary works, mainly from the 19th and early 20th century. Salomonsson et al. (2012) give a description of yet another Swedish NER system that recognizes persons, locations, organizations and the
category miscellaneous (which incorporates entities such as product, myth, event, animal and work), Salomonsson et al.’s system relies on machine learning techniques, using standard features such as part-of-speech tags of the surrounding and the current token, the word tokens themselves, the preceding two named entity tags and some other Boolean features such as initial capitalization and if a word contains digits. Finally, Ek et al. (2011) describe yet another Swedish NER system that recognizes named entities (locations, names, dates, times and telephone numbers) in Short Message Services (SMS) written in Swedish, that runs on an Android telephones.

3. HFST-SweNER
The NER tagger described by Kokkinakis (2004) has been converted and adapted to the Helsinki Finite-State Transducer Technology (HFST) platform. This new HFST-based NER (HFST-SweNER) is a full-fledged open source implementation that uses the pattern-matching tool HFST Pmatch in the HFST toolkit (Lindén et al. 2011; 2013). HFST Pmatch (hereafter simply Pmatch) has been modeled after Karttunen’s Finite-State Transducer (FST) pattern matching tool, pmatch (Karttunen, 2011). An important motivation for the work was to enable linguists who are familiar with the Xerox XFST morphology development environment to migrate their skills to writing NER and to combine NER and other language processing components in a uniform, standardized, open finite-state framework. With Pmatch, we have achieved this goal. By migrating a mature NER like the Swedish one to Pmatch we verified that the Pmatch environment is competitive with regard to expressiveness, file size, compilation and run-time speed.

4. The NER’s Major Parts
The original Swedish NER system, SweNER, (Kokkinakis, 2004) is comprised of a number of modules organized into layers and applied sequentially in a pipeline fashion. The major components are lists of multiword names (approximately 4,116 bigrams and 1,504 trigram generic entities, such as US Virgin islands and New York). The lists of multiword named entities are matched directly against the text being processed since empirical evidence has shown that such n-grams are reliable and can be safely applied in the early stages of NER for nearly any document genre, since there are rarely ambiguities or conflicts between multiword entities. Entity lists can provide very high precision, but low recall; they are also at first glance intuitively quite clear, but there are a lot of grey areas caused by a plethora of linguistic phenomena that can be encountered in real texts. In many of such cases the internal and external evidence (local context) can decide an appropriate label.

Therefore, to remedy some of these problems, the next major component in the pipeline consists of sets of linguistic rules – grammars – for each type of entity recognized by the system. Each rule in such a grammar defines patterns in the entity itself and its local context (e.g., typical classes of trigger words, designators and modifiers). These linguistic rules are used after the multiword entity look up stage is terminated.

Large lists\(^1\) of single-word names, currently 114,508 entities (categorized according to the description provided in Section 4.1) are consulted at a later step. Each named entity in these lists is marked for a major type and a minor subtype which further specifies the entity in question. At each processing stage a theory revision and refinement filter reviews the annotations, in order to detect and resolve possible errors and assign new annotations based on existing ones. This can involve merging compatible annotations, modifying and even deleting partial annotations and annotated fragments. This module also applies a set of manually built templates, a kind of local discourse structures, for recognizing potentially new entities that are neither in a gazetteer nor recognized by the grammar component and for which the surrounding context provides a reliable base for the recognition, e.g., in enumerations. For instance, the illustrative pattern ‘\(<ENTITY_1> Qqq \)</ENTITY_1>’ and ‘\(<ENTITY_2> Zzz \)</ENTITY_2>*’ (where ‘Xxx’ is meant to be any token, Qqq and Zzz two previously recognized entities with labels ENTITY\(_1\) and \(_2\) of the same class) will annotate the hypothetical entity XXX with the same label as in ENTITY\(_1\) and \(_2\).

4.1 Named Entity Taxonomy
As previously discussed, the nature and type of named entities vary depending on the task under investigation or the target application. In any case, person names, location and organization names are considered ‘generic’. Several attempts have been made to introduce richer named entity hierarchies; (Fleischman and Hovy, 2002; Sekine, 2004). We balance these initiatives by implementing a rather fine-grained named entity taxonomy with eight main named entity types as well as several (52) subtypes (Johannesen et al., 2005; Kokkinakis, 2004). The eight main categories are:

1. Person (PRS): people names (forenames, surnames), animal/pet names, mythological etc.;
2. Location (LOC): functional, geographical, geo-political, astronomical, street names;
3. Organization (ORG): political, athletic, media, military, transportation, education etc.;
4. Artifact (OBJ): food/wine products, prizes, means of communication (vehicles), etc.;
5. Work&Art (WRK): printed material, names of films, novels and newspapers, sculptures, etc.;
6. Event (EVN): religious, athletic, scientific, cultural, races, championships, battles, etc.;
7. Measure/Numerical (MSR): volume, age, index, dosage, web-related, speed etc.;
8. Temporal\(^2\) (TME).

\(^1\) We tried to avoid building large entity lists in order to mitigate the risk of identifying a large number of false positives. Smaller gazetteers are often a wiser choice (Mikhuev et al., 1999).

\(^2\) Temporal expressions recognized include both relative (nästa
Figure 1 illustrates how the named entity annotations look like in the underlying format. Identified named entities in context are surrounded by the tags `<ENAMEX ... >` entity category 1-6; `<NUMEX> ... </NUMEX>` entity category 7, or `<TIMEX> ... </TIMEX>` entity category 8. The starting tag also contains two attributes, which further specify the type and subtype of each recognized named entity. Thus, `<ENAMEX TYPE='ORG' SBT='PLT'> in Figure 1 provides an example in which the recognized named entities by HFST-SweNER (a) have the main type and their subtype annotated. The manual annotated version in the gold standard (b) is given for comparison: `<NAME type=inst> Socialdemokratiska förbundet </NAME>`, kunde starta Social-Demokraten och fick medarbetare som Axel Danielsson, Fredrik Sterky och, året därpå, Hjalmar Branting <s>`. The HFST-SweNER tagger, described in sections 5 and 6, identifies and classifies named entities into exactly the same set of predefined major entity categories and their subtypes as the SweNER does.

### 5. Converting SweNER to HFST Pmatch

The main aim of this work was to convert the original SweNER recognition rules, written basically in the Flex lexical analyzer and other scripting languages, to Pmatch expressions as faithfully as possible. We retained the original NER pipeline structure: 24 recognition stages and a theory revision and refinement filter that modifies, removes and adds new tags based on existing ones. Each stage forms a finite-state transducer.

Since both Flex and Pmatch are based on recognizing the leftmost longest matches of regular expressions, we were able to automate most of the conversion. A conversion script analysed the Flex actions to split a Flex regular expression pattern into a name and its context. The theory revision and refinement filter was converted by hand, since its rules were more varied than those in the recognizers. In contrast with the original SweNER, also the gazetteers needed to be compiled to finite-state automata. To facilitate that, Pmatch has a construct for reading an external file containing a list of strings.

Rule ordering in Flex was simulated by using weights in Pmatch expressions. If several Flex rules have the same leftmost longest match, the first one is chosen, so the rules can be ordered from the most specific to the most general. Although Pmatch cannot guarantee any specific order of matching for unweighted expressions, a desired order can effectively be achieved by adding to all alternative match expressions a penalty weight that is the higher the later an expression should be considered. Using weights is in general simpler than the alternative approaches that would add more detailed context conditions or subtract the more specific regular expressions from the more general. For example, the following expressions state that capitalized words ending in `gatan` should be tagged as street names (EnamexLocStr), except for `Vintergatan 'Milky Way'`, which should remain untagged (for a brief explanation of Pmatch syntax, see Section 6):

```plaintext
define Except {Vintergatan};
define Gatan UppercaseAlphaAlpha+ EndTag(EnamexLocStr);
define TOP Except | Gatan;
```

Without the penalty weights (the numbers following the semicolons), `Vintergatan` might be incorrectly tagged as a street name.

Differences remaining between the recognition results of the original SweNER and HFST-SweNER are explained by differences in the semantics of Flex NER rules and Pmatch. In particular, the regular expression patterns in the Flex rules cover the contexts in addition to the name to be recognized, whereas Pmatch excludes

---

3 ENAMEX stands for ‘Extended NAME ExPreSSION’.
contexts from its leftmost longest match. Consequently, the leftmost longest match at a certain point in text may be found by different expressions in Flex and Pmatch.

Cyclic finite-state automata which appear as part of a complex regular expression sometimes grow considerably when the expression is determined. To reduce the size of and the compilation time required for the largest NER automata, we selectively added guards around some of the subexpressions to keep them from being determined as part of their context. This introduces some limited non-determinism during run time but keeps the local automata and their compilation time reasonably small. The guards are called Pmatch insertion statements. Using insertion statements reduced the size and compilation time by a factor of 5 on the average and by a factor of 40 in the best case. The largest improvements were in cases where the insertion statement was added to recurring sub-expressions recognizing general patterns, such as “any word”, at the beginning of a recognition rule.

6. NE Recognition with HFST Pmatch

A key feature of Pmatch, making it well-suited for NER, is the ability to efficiently add XML-style tags around substrings matching a regular expression, as in Karttunen (2011). For example, the expressions in Figure 2 mark capitalized words with EnamexOrgCrp only if they are preceded by the string `rörelseresultatet för` ‘operating profit of’:

```plaintext
define NoSpTag [? - [Whitespace|"|']> | ];
define CapWord2 UppercaseAlpha NoSpTag+;
define OrgCrpOpProfit CapWord2 ["CapWord2"]
EndTag(EnamexOrgCrp) LC((rörelseresultatet för ));
define TOP OrgCrpOpProfit;
```

Figure 2: Pmatch example.

E.g.: `rörelseresultatet för` <EnamexOrgCrp>Comp Systems</EnamexOrgCrp> 4 ...`). Pmatch considers leftmost longest matches of the expression named TOP in the input and adds the tags specified with EndTag (TagName) in TOP or in the expressions to which TOP refers. To disambiguate between matches, a regular expression may be accompanied with a context condition specifying that a match should be considered only if the left or right context of the match matches the context condition (specified in LC() or RC(), respectively). The built-in set Whitespace denotes any whitespace character and UppercaseAlpha any uppercase letter.

7. Evaluation and Gold Standard

For the evaluation we used an existing Swedish gold

standard corpus, namely the Stockholm-Umeå Corpus, SUC version 3.0 (SUC3.0; Gustafsson-Capkova and Hartmann, 2006). In SUC3.0, all named entities (34,194) have been manually annotated and could thus be used for the gold standard evaluation. Table 1 shows the entities found in SUC 3.0 and their corresponding labels before and after their conversion.

Although SUC’s entity labels are not 100% compatible with HFST-SweNER’s tagset, it is the closest we can get with respect to a gold standard corpus for automatic evaluation of the HFST-SweNER. The automatic evaluation was made with the conlleval script.5

<table>
<thead>
<tr>
<th>SUC3.0</th>
<th>HFST-SweNER</th>
</tr>
</thead>
<tbody>
<tr>
<td>person</td>
<td>15,128</td>
</tr>
<tr>
<td>place</td>
<td>8,776</td>
</tr>
<tr>
<td>inst</td>
<td>6,334</td>
</tr>
<tr>
<td>work</td>
<td>1,887</td>
</tr>
<tr>
<td>product</td>
<td>638</td>
</tr>
<tr>
<td>other</td>
<td>542</td>
</tr>
<tr>
<td>animal</td>
<td>364</td>
</tr>
<tr>
<td>myth</td>
<td>280</td>
</tr>
<tr>
<td>event</td>
<td>245</td>
</tr>
<tr>
<td>num</td>
<td>18,098</td>
</tr>
</tbody>
</table>

Table 1. Named entity tags and their occurrences in SUC3.0, and their conversion to the HFST-SweNER.

We evaluated the NER in two ways. Firstly, we conducted a functional evaluation (Section 7.2), secondly an evaluation based on all entities (with minor exceptions, see Section 7.1) found in the same gold standard (Section 7.3).

7.1 Preprocessing

In order to ease the comparison between the annotations, we harmonized the annotations between the original SUC3.0 and the ones returned by HFST-SweNER. Thus in order to make the evaluation more reliable, we performed a few preprocessing steps in SUC3.0. For convenience reasons, we converted the SUC3.0 label tags (i.e., name type=) to the ones used in HFST-SweNER (i.e., ENAMEX). The most important preprocessing step was applied for the category “person” for which SUC3.0 sometimes includes in the annotation an apposition or designator, often a common noun in lower case. These, however, seem to be used in an ad hoc manner; for instance in SUC3.0 there are annotations such as: `<name type=person>kusin Bosse</name>` “cousin Bosse” (SUC3.0-file: kr06); `<name type=person>tante Ture</name>` “That uncle Ture offered you ten percent …” (SUC3.0-file: kr06) or `<name type=person>Tant Sigrid</name>` “Aunt Sigrid” (SUC3.0-file: kr04). Therefore, these designators were moved outside of the annotation. Thus, the three previous examples were changed to: `<name type=person>kusin</name> ...’ (SUC3.0-file: kr06) or `<name type=person>tante Ture</name>...

5 For the evaluation we used the conlleval script (v. 04-01-26) by Tjong Kim Sang <http://www.cnts.ua.ac.be/conll2002/ner/bin/conlleval.txt>.
explained by the fact that SUC3.0 does not mark all occurrences of certain entities such as: ‘Naturskyddsföreningen’ ‘the Swedish Society for Nature Conservation’, Nätverket ‘The National Board of Health and Welfare’ or Naturvårdsverket ‘the Swedish Environmental Protection Agency’ or not at all other entities such as Socialnämnden ‘Social Services Committee’. The results in Table 2 are shown with and without considering whether an identified entity is part of a multi token annotation. Using the so called IOB representation, each identified named entity is prefixed by either ‘B-’ or ‘I-’. Here ‘B’ (for Begin) marks the first word in a multi token named entity (or a single entity token), while further named entity tokens receive the prefix ‘I’, for Inside. In text, tokens that are not part of named entity receive the annotation O (for Outside).

<table>
<thead>
<tr>
<th>Label</th>
<th>Precision</th>
<th>Recall</th>
<th>FB1</th>
<th>#</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRS</td>
<td>91.34%</td>
<td>91.32%</td>
<td>91.33%</td>
<td>22492</td>
</tr>
<tr>
<td>LOC</td>
<td>73.03%</td>
<td>83.78%</td>
<td>78.04%</td>
<td>10917</td>
</tr>
<tr>
<td>ORG</td>
<td>70%</td>
<td>43.17%</td>
<td>53.41%</td>
<td>5220</td>
</tr>
<tr>
<td>B-PRS</td>
<td>88.78%</td>
<td>87.45%</td>
<td>88.11%</td>
<td>15519</td>
</tr>
<tr>
<td>I-PRS</td>
<td>87.90%</td>
<td>90.89%</td>
<td>89.37%</td>
<td>6973</td>
</tr>
<tr>
<td>B-LOC</td>
<td>75.42%</td>
<td>84.69%</td>
<td>79.79%</td>
<td>9854</td>
</tr>
<tr>
<td>I-LOC</td>
<td>43.37%</td>
<td>62.21%</td>
<td>51.11%</td>
<td>1063</td>
</tr>
<tr>
<td>B-ORG</td>
<td>65.00%</td>
<td>39.37%</td>
<td>49.04%</td>
<td>3837</td>
</tr>
<tr>
<td>I-ORG</td>
<td>76.28%</td>
<td>49.53%</td>
<td>60.06%</td>
<td>1383</td>
</tr>
<tr>
<td>OBJ</td>
<td>64.14%</td>
<td>33.12%</td>
<td>43.68%</td>
<td>488</td>
</tr>
<tr>
<td>WRK</td>
<td>52.91%</td>
<td>18.56%</td>
<td>27.48%</td>
<td>1544</td>
</tr>
<tr>
<td>EVN</td>
<td>41.62%</td>
<td>66.85%</td>
<td>51.30%</td>
<td>591</td>
</tr>
<tr>
<td>B-OBI</td>
<td>55.23%</td>
<td>29.78%</td>
<td>38.70%</td>
<td>344</td>
</tr>
<tr>
<td>I-OBJ</td>
<td>78.47%</td>
<td>36.81%</td>
<td>50.11%</td>
<td>144</td>
</tr>
<tr>
<td>B-EVN</td>
<td>42.97%</td>
<td>66.12%</td>
<td>52.09%</td>
<td>377</td>
</tr>
<tr>
<td>I-EVN</td>
<td>35.98%</td>
<td>62.60%</td>
<td>45.70%</td>
<td>214</td>
</tr>
<tr>
<td>B-WRK</td>
<td>44.88%</td>
<td>17.44%</td>
<td>25.11%</td>
<td>733</td>
</tr>
<tr>
<td>I-WRK</td>
<td>46.49%</td>
<td>14.99%</td>
<td>22.67%</td>
<td>811</td>
</tr>
<tr>
<td>All</td>
<td>79.02%</td>
<td>70.56%</td>
<td>74.55%</td>
<td>15519</td>
</tr>
</tbody>
</table>

Table 2. Evaluation results for the entity types in SUC3.0; ‘#’ occurrences in SUC3.0 after the preprocessing steps (see Section 7.1)

8. Conclusions

In this paper we have described the conversion, modeling, adaptation and evaluation of a Swedish named entity recognizer to the Helsinki Finite-State Transducer Technology toolkit intended for various language software applications. Language technology in the form of named entity recognition and annotation systems can be usefully applied for building computer linguage to support various applications e.g., in the area of digital humanities. For instance, entity annotations allow a more semantically-oriented exploration of textual content in large digitized archives (Borin and Kokkinakis, 2010; Elson et al., 2010).

What can we conclude from the results of the evaluation? Apparently the results do not reflect the potential of the HFST-SweNER in a fair way. Several entity annotations in SUC3.0 are not consistent and there are various encountered examples that supported the fact

---


For instance, the low recall for the ORG class can be particularly, between the location and organization classes. For instance, the low recall for the ORG class can be explained by the fact that SUC3.0 does not mark all occurrences of certain entities such as: Naturskyddsföreningen ‘the Swedish Society for Nature Conservation’, Socialstyrelsen ‘The National Board of Health and Welfare’ or Naturlandsverket ‘the Swedish Environmental Protection Agency’ or not at all other entities such as Socialnämnden ‘Social Services Committee’. The results in Table 2 are shown with and without considering whether an identified entity is part of a multi token annotation. Using the so called IOB representation, each identified named entity is prefixed by either ‘B-‘ or ‘I-‘. Here ‘B’ (for Begin) marks the first word in a multi token named entity (or a single entity token), while further named entity tokens receive the prefix ‘I’, for Inside. In text, tokens that are not part of named entity receive the annotation O (for Outside).

Table 2. Evaluation results for the entity types in SUC3.0; ‘#’ occurrences in SUC3.0 after the preprocessing steps (see Section 7.1)

8. Conclusions

In this paper we have described the conversion, modeling, adaptation and evaluation of a Swedish named entity recognizer to the Helsinki Finite-State Transducer Technology toolkit intended for various language software applications. Language technology in the form of named entity recognition and annotation systems can be usefully applied for building computer linguage to support various applications e.g., in the area of digital humanities. For instance, entity annotations allow a more semantically-oriented exploration of textual content in large digitized archives (Borin and Kokkinakis, 2010; Elson et al., 2010).

What can we conclude from the results of the evaluation? Apparently the results do not reflect the potential of the HFST-SweNER in a fair way. Several entity annotations in SUC3.0 are not consistent and there are various encountered examples that supported the fact

---


For instance, the low recall for the ORG class can be particularly, between the location and organization classes. For instance, the low recall for the ORG class can be explained by the fact that SUC3.0 does not mark all occurrences of certain entities such as: Naturskyddsföreningen ‘the Swedish Society for Nature Conservation’, Socialstyrelsen ‘The National Board of Health and Welfare’ or Naturlandsverket ‘the Swedish Environmental Protection Agency’ or not at all other entities such as Socialnämnden ‘Social Services Committee’. The results in Table 2 are shown with and without considering whether an identified entity is part of a multi token annotation. Using the so called IOB representation, each identified named entity is prefixed by either ‘B-‘ or ‘I-‘. Here ‘B’ (for Begin) marks the first word in a multi token named entity (or a single entity token), while further named entity tokens receive the prefix ‘I’, for Inside. In text, tokens that are not part of named entity receive the annotation O (for Outside).

Table 2. Evaluation results for the entity types in SUC3.0; ‘#’ occurrences in SUC3.0 after the preprocessing steps (see Section 7.1)

8. Conclusions

In this paper we have described the conversion, modeling, adaptation and evaluation of a Swedish named entity recognizer to the Helsinki Finite-State Transducer Technology toolkit intended for various language software applications. Language technology in the form of named entity recognition and annotation systems can be usefully applied for building computer linguage to support various applications e.g., in the area of digital humanities. For instance, entity annotations allow a more semantically-oriented exploration of textual content in large digitized archives (Borin and Kokkinakis, 2010; Elson et al., 2010).

What can we conclude from the results of the evaluation? Apparently the results do not reflect the potential of the HFST-SweNER in a fair way. Several entity annotations in SUC3.0 are not consistent and there are various encountered examples that supported the fact

---

that the evaluation results should be taken with caution since there is a strong indication that these inconsistencies have a negative impact on the evaluation performance and the figures in Table 2. Consider for instance the examples: `<s id=ce02b-002> <name type=inst> Konserthusets </name> stora sal: [...] </s>` and `<s id=kr01a-035> [...] fylla <name type=other> Konserthusets </name> stora sal. </s>`, the first annotated as `inst` (organization) and the second as `other`; both, however, refer to a physical location `Concert hall`. Certain pieces of work as well as events are not marked in SUC3.0, such as Catechesationsbok `Catechization book` (SUC3.0-file jd06); Valborgsmässoafton `Walpurgis Night`; or Kristi himmelsfärd `Ascension`; while other similar kind of entities are sometime marked sometime not; e.g. (SUC3.0-file kk11) `<name type=other> Svenska flaggans dag </name>` ‘the Swedish flag day’ but not the Skånska flaggans dag `Scanian flag day’ (in SUC3.0-file af05). Other multiword entities are also annotated in an ad hoc manner, for instance, only the first part of the expression, i.e. ‘Big’, is annotated in the following example `<name type=event> Big </name> Bang-teorins `the Big Bang theory’s’` (SUC3.0-file: fh12). Therefore, we believe that the gold standard we used for the evaluation needs to be thoroughly re-checked for the entity annotations it contains. Of course, this fact does not diminish the importance of SUC3.0 as a very valuable resource for evaluating Swedish NER components.

Although a whole new NER system could be rebuilt in a machine learning framework or other rule-based framework, we decided to use Pmatch in order to verify that we are offering an environment that is competitive with the technologies used in SweNER with regard to expressiveness, compilation speed, file size and run-time speed. Note also that although we are relatively satisfied with the first three, we still need to work on the run-time speed in comparison with Flex. We believe that for even better results we need to integrate the strengths of rule-based systems with the ones provided by supervised learning, a task we have left for exploration in the near future.

9. Acknowledgements

This work was supported by the Centre for Language Technology <http://clt.gu.se/> at the University of Gothenburg and the FIN-CLARIN at the University of Helsinki <www.helsinki.fi/finclarin>. The current version of the HFST-SweNER is available from <http://clt.gu.se/clt-toolkit/>. The license for HFST is LGPL by default but some of HFST’s runtime libraries are available with an Apache license as well. The HFST-SweNER includes the uncompiled and compiled rule set with a CC-BY license which is also the license for the HFST-SweNER.

10. References


