A FrameNet-based Semantic Role Labeler for Swedish

Richard Johansson and **Pierre Nugues**
Department of Computer Science, LTH
Lund University, Sweden
{richard, pierre}@cs.lth.se

Abstract

We present a FrameNet-based semantic role labeling system for Swedish text. As training data for the system, we used an annotated corpus that we produced by transferring FrameNet annotation from the English side to the Swedish side in a parallel corpus. In addition, we describe two frame element bracketing algorithms that are suitable when no robust constituent parsers are available.

We evaluated the system on a part of the FrameNet example corpus that we translated manually, and obtained an accuracy score of 0.75 on the classification of pre-segmented frame elements, and precision and recall scores of 0.67 and 0.47 for the complete task.

1 Introduction

Semantic role labeling (SRL), the process of automatically identifying arguments of a predicate in a sentence and assigning them semantic roles, has received much attention during the recent years. SRL systems have been used in a number of projects in Information Extraction and Question Answering, and are believed to be applicable in other domains as well.

Building SRL systems for English has been studied widely (Gildea and Jurafsky, 2002; Litkowski, 2004), *inter alia*. However, all these works rely on corpora that have been produced at the cost of a large effort by human annotators. For instance, the current FrameNet corpus (Baker et al., 1998) consists of 130,000 manually annotated sentences. For smaller languages such as Swedish, such corpora are not available.

In this work, we describe a FrameNet-based semantic role labeler for Swedish text. Since there was no existing training corpus available — no FrameNet-annotated Swedish corpus of substantial size exists — we used an English-Swedish parallel corpus whose English part was annotated with semantic roles using the FrameNet annotation scheme. We then applied a *cross-language transfer* to derive an annotated Swedish part. To evaluate the performance of the Swedish SRL system, we applied it to a small portion of the FrameNet example corpus that we translated manually.

1.1 FrameNet: an Introduction

FrameNet (Baker et al., 1998) is a lexical database that describes English words using Frame Semantics (Fillmore, 1976). In this framework, predicates (or in FrameNet terminology, *target words*) and their arguments are linked by means of *semantic frames*. A frame can intuitively be thought of as a template that defines a set of slots, *frame elements* (FEs), that represent parts of the conceptual structure and typically correspond to prototypical participants or properties.

Figure 1 shows an example sentence annotated with FrameNet information. In this example, the target word *statements* belongs to ("evokes") the frame STATEMENT. Two constituents that fill slots of the frame (SPEAKER and TOPIC) are annotated as well.

As usual in these cases, [both parties]$_{\text{SPEAKER}}$ agreed to make no further **statements** [on the matter]$_{\text{TOPIC}}$.

Figure 1: A sentence from the FrameNet example corpus.

The initial versions of FrameNet were focused on describing situations and events, i.e. typically verbs and their nominalizations. Currently, however, FrameNet defines frames for a wider range of semantic relations that can be thought of as predicate/argument structures, including descriptions of events, states, properties, and objects.

FrameNet consists of the following main parts:

- An *ontology* consisting of a set of frames, frame elements for each frame, and relations (such as inheritance and causative-of) between frames.
- A list of *lexical units*, that is word forms paired with their corresponding frames. The frame is used to distinguish between different senses of the word, although the treatment of polysemy in FrameNet is relatively coarse-grained.
- A collection of *example sentences* that provide lexical evidence for the frames and the corresponding lexical units. Although this corpus is not intended to be representative, it is typically used as a training corpus when contructing automatic FrameNet labelers.

1.2 Related Work

Since training data is often a scarce resource for most languages other than English, a wide range of methods have been proposed to reduce the need for manual annotation. Many of these have relied on existing resources for English and a transfer method based on word alignment in a parallel corpus to automatically create an annotated corpus in a new language. Although these data are typically quite noisy, they have been used to train automatic systems.

For the particular case of transfer of FrameNet annotation, there have been a few projects that have studied transfer methods and evaluated the quality of the automatically produced corpus. Johansson and Nugues (2005) applied the word-based methods of Yarowsky et al. (2001) and obtained promising results. Another recent effort (**?**) demonstrates that deeper linguistic information, such as parse trees in the source and target language, is very beneficial for the process of FrameNet annotation transfer.

A rather different method to construct bilingual semantic role annotation is the approach taken by BiFrameNet (Fung and Chen, 2004). In that work, annotated structures in a new language (in that case Chinese) are produced by mining for similar structures rather than projecting them via parallel corpora.

2 Automatic Annotation of a Swedish Training Corpus

2.1 Training an English Semantic Role Labeler

We selected the 150 most frequent frames in FrameNet and applied the Collins parser (**?**) to the example sentences for these frames. We built a conventional FrameNet parser for English using 100,000 of these sentences as a training set and 8,000 as a development set. The classifiers were based on Support Vector Machines that we trained using LIBSVM (Chang and Lin, 2001) with the Gaussian kernel. When testing the system, we did not assume that the frame was known a priori. We used the available semantic roles for all senses of the target word as features for the classifier.

On a test set from FrameNet, we estimated that the system had a precision of 0.71 and a recall of 0.65 using a strict scoring method. The result is slightly lower than the best systems at Senseval-3 (Litkowski, 2004), possibly because we used a larger set of frames, and we did not assume that the frame was known a priori.

2.2 Transferring the Annotation

We produced a Swedish-language corpus annotated with FrameNet information by applying the SRL system to the English side of Europarl (Koehn, 2005), which is a parallel corpus that is derived from the proceedings of the European Parliament. We projected the bracketing of the target words and the frame elements onto the Swedish side of the corpus by using the Giza++ word aligner (Och and Ney, 2003). Each word on the English side was mapped by the aligner onto a (possibly empty) set of words on the Swedish side. We used the maximal span method to infer the bracketing on the Swedish side, which means that the span of a projected entity was set to the range from the leftmost projected token to the rightmost. Figure 2 shows an example of this process.

To make the brackets conform to the FrameNet annotation practices, we applied a small set of heuristics. The FrameNet conventions specify that linking words such as prepositions and subordinating conjunctions should be included in the brack-

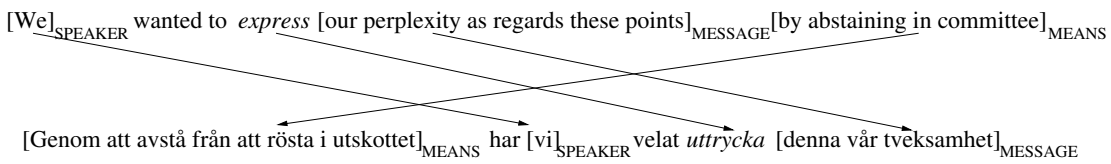

Figure 2: Example of projection of FrameNet annotation.

eting. However, since constructions are not isomorphic in the sentence pair, a linking word on the target side may be missed by the projection method since it is not present on the source side. For example, the sentence *the doctor was answering an emergency phone call* is translated into Swedish as *doktorn svarade* ***på*** *ett larmsamtal*, which uses a construction with a preposition *på* 'to/at/on' that has no counterpart in the English sentence. The heuristics that we used are specific for Swedish, although they would probably be very similar for any other language that uses a similar set of prepositions and connectives, i.e. most European languages.

We used the following heuristics:

- When there was only a linking word (preposition, subordinating conjunction, or infinitive marker) between the FE and the target word, it was merged with the FE.
- When a Swedish FE was preceded by a linking word, and the English FE starts with such a word, it was merged with the FE.
- We used a chunker and adjusted the FE brackets to include only complete chunks.
- When a Swedish FE crossed the target word, we used only the part of the FE that was on the right side of the target.

In addition, some bad annotation was discarded because we obviously could not use sentences where no counterpart for the target word could be found. Additionally, we used only the sentences where the target word was mapped to a noun, verb, or an adjective on the Swedish side.

Because of homonymy and polysemy problems, applying a SRL system without knowing target words and frames a priori necessarily introduces noise into the automatically created training corpus. There are two kinds of word sense ambiguity that are problematic in this case: the "internal" ambiguity, or the fact that there may be more than one frame for a given target word; and the "external" ambiguity, where frequently occurring word senses are not listed in FrameNet. To sidestep the problem of internal ambiguity, we used the available semantic roles for all senses of the target word as features for the classifier (as described above). Solving the problem of external ambiguity was outside the scope of this work.

Some potential target words had to be ignored since their sense ambiguity was too difficult to overcome. This category includes auxiliaries such as *be* and *have*, as well as verbs such as *take* and *make*, which frequently appear as support verbs for nominal predicates.

2.3 Motivation

Although the meaning of the two sentences in a sentence pair in a parallel corpus should be roughly the same, a fundamental question is whether it is meaningful to project semantic markup of text across languages. Equivalent words in two different languages sometimes exhibit subtle but significant semantic differences. However, we believe that a transfer makes sense, since the nature of FrameNet is rather coarse-grained. Even though the words that evoke a frame may not have exact counterparts, it is probable that the frame itself has.

For the projection method to be meaningful, we must make the following assumptions:

- The complete frame ontology in the English FrameNet is meaningful in Swedish as well, and each frame has the same set of semantic roles and the same relations to other frames.
- When a target word evokes a certain frame in English, it has a counterpart in Swedish that evokes the same frame.
- Some of the FEs on the English side have counterparts with the same semantic roles on the Swedish side.

In addition, we made the (obviously simplistic)

assumption that the contiguous entities we project are also contiguous on the target side.

These assumptions may all be put into question. Above all, the second assumption will fail in many cases because the translations are not literal, which means that the sentences in the pair may express slightly different information. The third assumption may be invalid if the information expressed is realized by radically different constructions, which means that an argument may belong to another predicate or change its semantic role on the Swedish side. **?**) avoid this problem by using heuristics based on a target-language FrameNet to select sentences that are close in meaning. Since we have no such resource to rely on, we are forced to accept that this problem introduces a certain amount of noise into the automatically annotated corpus.

3 Training a Swedish SRL System

Using the transferred FrameNet annotation, we trained a SRL system for Swedish text. Like most previous systems, it consists of two parts: a FE bracketer and a classifier that assigns semantic roles to FEs. Both parts are implemented as SVM classifiers trained using LIBSVM. The semantic role classifier is rather conventional and is not described in this paper.

To construct the features used by the classifiers, we used the following tools:

- An HMM-based POS tagger,
- A rule-based chunker,
- A rule-based time expression detector,
- Two clause identifiers, of which one is rule-based and one is statistical,
- The MALTPARSER dependency parser (**?**), trained on a 100,000-word Swedish treebank.

We constructed shallow parse trees using the clause trees and the chunks. Dependency and shallow parse trees for a fragment of a sentence from our test corpus are shown in Figures 3 and 4, respectively. This sentence, which was translated from an English sentence that read *the doctor was answering an emergency phone call*, comes from the English FrameNet example corpus.

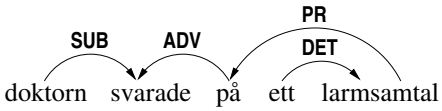

Figure 3: Example dependency parse tree.

[[doktorn]$_{NG\_nom}$ [svarade]$_{VG\_fin}$ [på]$_{PP}$ [ett larmsamtal]$_{NG\_nom}$]$_{Clause}$

Figure 4: Example shallow parse tree.

3.1 Frame Element Bracketing Methods

We created two redundancy-based FE bracketing algorithms based on binary classification of chunks as starting or ending the FE. This is somewhat similar to the chunk-based system described by Pradhan et al. (2005a), which uses a segmentation strategy based on IOB2 bracketing. However, our system still exploits the dependency parse tree during classification.

We first tried the conventional approach to the problem of FE bracketing: applying a parser to the sentence, and classifying each node in the parse tree as being an FE or not. We used a dependency parser since there is no constituent-based parser available for Swedish. This proved unsuccessful because the spans of the dependency subtrees frequently were incompatible with the spans defined by the FrameNet annotations. This was especially the case for non-verbal target words and when the head of the argument was above the target word in the dependency tree. To be usable, this approach would require some sort of transformation, possibly a conversion into a phrase-structure tree, to be applied to the dependency trees to align the spans with the FEs. Preliminary investigations were unsuccessful, and we left this to future work.

We believe that the methods we developed are more suitable in our case, since they base their decisions on several parse trees (in our case, two clause-chunk trees and one dependency tree). This redundancy is valuable because the dependency parsing model was trained on a treebank of just 100,000 words, which makes it less robust than Collins' or Charniak's parsers for English. In addition, the methods do not implicitly rely on the common assumption that every FE has a counterpart in a parse tree. Recent work in semantic role labeling, see for example Pradhan et al. (2005b), has focused on combining the results of SRL systems based on different types of syntax. Still, all

systems exploiting recursive parse trees are based on binary classification of nodes as being an argument or not.

The training sets used to train the final classifiers consisted of one million training instances for the start classifier, 500,000 for the end classifier, and 272,000 for the role classifier. The features used by the classifiers are described in Subsection 3.2, and the performance of the two FE bracketing algorithms compared in Subsection 4.2.

3.1.1 Greedy start-end

The first FE bracketing algorithm, the *greedy start-end* method, proceeds through the sequence of chunks in one pass from left to right. For each chunk opening bracket, a binary classifier decides if an FE starts there or not. Similarly, another binary classifier tests chunk end brackets for ends of FEs. To ensure compliance to the FrameNet annotation standard (bracket matching, and no FE crossing the target word), the algorithm inserts additional end brackets where appropriate. Pseudocode is given in Algorithm 1.

Algorithm 1 Greedy Bracketing

Input: A list L of chunks and a target word t
  Binary classifiers `starts` and `ends`
Output: The sets S and E of start and end brackets
  Split L into the sublists L_{before}, L_{target}, and L_{after}, which correspond to the parts of the list that is before, at, and after the target word, respectively.
  Initialize `chunk-open` to FALSE
  for L_{sub} **in** $\{L_{\text{before}}, L_{\text{target}}, L_{\text{after}}\}$ **do**
    for c **in** L_{sub} **do**
      if `starts`(c) **then**
        if `chunk-open` **then**
          Add an end bracket before c to E
        end if
        `chunk-open` $\leftarrow$ TRUE
        Add a start bracket before c to S
      end if
      if `chunk-open` $\wedge$ (`ends`$(c) \vee c$ is final in L_{sub}) **then**
        `chunk-open` $\leftarrow$ FALSE
        Add an end bracket after c to E
      end if
    end for
  end for

Figure 5 shows an example of this algorithm, applied to the example fragment. The small brackets correspond to chunk boundaries, and the large brackets to FE boundaries that the algorithm inserts. In the example, the algorithm inserts an end bracket after the word *doktorn* 'the doctor', since no end bracket was found before the target word *svarade* 'was answering'.

3.1.2 Globally optimized start-end

The second algorithm, the *globally optimized start-end* method, maximizes a global probability score over each sentence. For each chunk opening and closing bracket, probability models assign

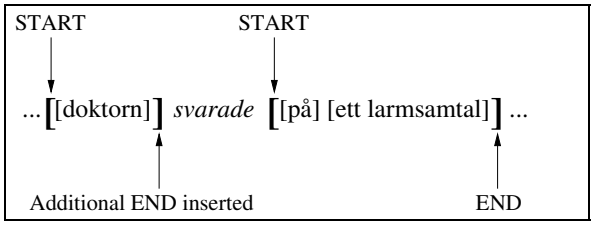

Figure 5: Illustration of the greedy start-end method.

the probability of an FE starting (or ending, respectively) at that chunk. The probabilities are estimated using the built-in sigmoid fitting methods of LIBSVM. Making the somewhat unrealistic assumption of independence of the brackets, the global probability score to maximize is defined as the product of all start and end probabilities. We added a set of constraints to ensure that the segmentation conforms to the FrameNet annotation standard. The constrained optimization problem is then solved using the JACOP finite domain constraint solver (Kuchcinski, 2003). We believe that an n-best beam search method would produce similar results. The pseudocode for the method can be seen in Algorithm 2. The definitions of the predicates `no-nesting` and `no-crossing`, which should be obvious, are omitted.

Algorithm 2 Globally Optimized Bracketing

Input: A list L of chunks and a target word t
  Probability models $\hat{P}_{\text{starts}}$ and $\hat{P}_{\text{ends}}$
Output: The sets $S_{\max}$ and $E_{\max}$ of start and end brackets

$$\begin{aligned}
\texttt{legal}(S,E) \quad &\leftarrow \quad |S| = |E| \\
&\wedge \quad \max(E) > \max(S) \wedge \min(S) < \min(E) \\
&\wedge \quad \texttt{no-nesting}(S,E) \wedge \texttt{no-crossing}(t,S,E) \\
\texttt{score}(S,E) \quad &\leftarrow \quad \textstyle\prod_{c\in S} \hat{P}_{\text{starts}}(c) \cdot \prod_{c\in L\setminus S}(1-\hat{P}_{\text{starts}}(c)) \\
&\quad\quad \cdot \textstyle\prod_{c\in E} \hat{P}_{\text{ends}}(c) \cdot \prod_{c\in L\setminus E}(1-\hat{P}_{\text{ends}}(c))
\end{aligned}$$

$$(S_{\max}, E_{\max}) \leftarrow \operatorname{argmax}_{\{\texttt{legal}(S,E)\}} \texttt{score}(S,E)$$

Figure 6 shows an example of the globally optimized start-end method. In the example, the global probability score is maximized by a bracketing that is illegal because the FE starting at *doktorn* is not closed before the target ($0.8 \cdot 0.6 \cdot 0.6 \cdot 0.7 \cdot 0.8 \cdot 0.7 = 0.11$). The solution of the constrained problem is a bracketing that contains an end bracket before the target ($0.8 \cdot 0.4 \cdot 0.6 \cdot 0.7 \cdot 0.8 \cdot 0.7 = 0.075$)

3.2 Features Used by the Classifiers

Table 1 summarizes the feature sets used by the greedy start-end (GSE), optimized start-end (OSE), and semantic role classification (SRC).

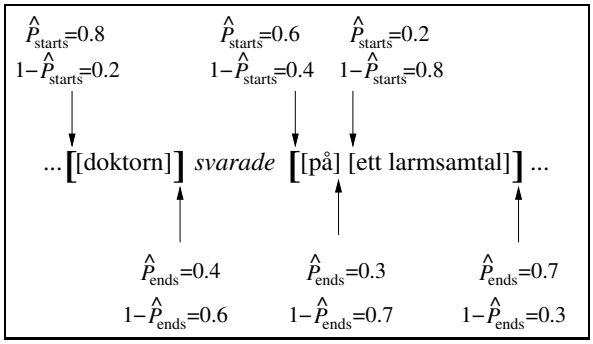

Figure 6: Illustration of the globally optimized start-end method.

	GSE	OSE	SRC
Target lemma	+	+	+
Target POS	+	+	+
Voice	+	+	+
Allowed role labels	+	+	+
Position	+	+	+
Head word (HW)	+	+	+
Head POS	+	+	+
Phrase/chunk type (PT)	+	+	+
HW/POS/PT, ±2 chunk window	+	+	-
Dep-tree & shallow path →target	+	+	+
Starting paths →target	+	+	-
Ending paths →target	+	+	-
Path →start	+	-	-

Table 1: Features used by the classifiers.

3.2.1 Conventional Features

Most of the features that we use have been used by almost every system since the first well-known description (Gildea and Jurafsky, 2002). The following of them are used by all classifiers:

- *Target word (predicate) lemma and POS*
- *Voice* (when the target word is a verb)
- *Position* (before or after the target)
- *Head word and POS*
- *Phrase or chunk type*

In addition, all classifiers use the set of allowed semantic role labels as a set of boolean features. This is needed to constrain the output to a label that is allowed by FrameNet for the current frame. In addition, this feature has proven useful for the FE bracketing classifiers to distinguish between event-type and object-type frames. For event-type frames, dependencies are often long-distance, while for object-type frames, they are typically restricted to chunks very near the target word. The part of speech of the target word alone is not enough to distinguish these two classes, since many nouns belong to event-type frames.

For the phrase/chunk type feature, we use slightly different values for the bracketing case and the role assignment case: for bracketing, the value of this feature is simply the type of the current chunk; for classification, it is the type of the largest chunk or clause that starts at the leftmost token of the FE. For prepositional phrases, the preposition is attached to the phrase type (for example, the second FE in the example fragment starts with the preposition *på* 'at/on', which causes the value of the phrase type feature to be PP-*på*).

3.2.2 Chunk Context Features

Similarly to the chunk-based PropBank argument bracketer described by Pradhan et al. (2005a), the start-end methods use the head word, head POS, and chunk type of chunks in a window of size 2 on both sides of the current chunk to classify it as being the start or end of an FE.

3.2.3 Parse Tree Path Features

Parse tree path features have been shown to be very important for argument bracketing in several studies. All classifiers used here use a set of such features:

- *Dependency tree path from the head to the target word.* In the example text, the first chunk (consisting of the word *doktorn*), has the value SUB-↑ for this feature. This means that to go from the head of the chunk to the target in the dependency graph (Figure 3), you traverse a SUB (subject) link upwards. Similarly, the last chunk (*ett larmsamtal*) has the value PR-↑-ADV-↑.

- *Shallow path from the chunk containing the head to the target word.* For the same chunks as above, these values are both NG_nom-↑-Clause-↓-VG_fin, which means that to traverse the shallow parse tree (Figure 4) from the chunk to the target, you start with a NG_nom node, go upwards to a Clause node, and finally down to the VG_fin node.

The start-end classifiers additionally use the full set of paths (dependency and shallow paths) to the target word from each node starting (or ending, respectively) at the current chunk, and the greedy end classifier also uses the path from the current chunk to the start chunk.

4 Evaluation of the System

4.1 Evaluation Corpus

To evaluate the system, we manually translated 150 sentences from the FrameNet example corpus. These sentences were selected randomly from the English development set. Some sentences were removed, typically because we found the annotation dubious or the meaning of the sentence difficult to comprehend precisely. The translation was mostly straightforward. Because of the extensive use of compounding in Swedish, some frame elements were merged with target words.

4.2 Comparison of FE Bracketing Methods

We compared the performance of the two methods for FE bracketing on the test set. Because of limited time, we used smaller training sets than for the full evaluation below (100,000 training instances for all classifiers). Table 2 shows the result of this comparison.

	Greedy	Optimized
Precision	0.70	0.76
Recall	0.50	0.44
$F_{\beta=1}$	0.58	0.55

Table 2: Comparison of FE bracketing methods.

As we can see from the Table 2, the globally optimized start-end method increased the precision somewhat, but decreased the recall and made the overall F-measure lower. We therefore used the greedy start-end method for our final evaluation that is described in the next section.

4.3 Final System Performance

We applied the Swedish semantic role labeler to the translated sentences and evaluated the result. We used the conventional experimental setting where the frame and the target word were given in advance. The results, with approximate 95% confidence intervals included, are presented in Table 3. The figures are precision and recall for the full task, classification accuracy of pre-segmented arguments, precision and recall for the bracketing task, full task precision and recall using the Senseval-3 scoring metrics, and finally the proportion of full sentences whose FEs were correctly bracketed and classified. The Senseval-3 method uses a more lenient scoring scheme that counts a FE as correctly identified if it overlaps with the gold standard FE and has the correct label. Although the strict measures are more interesting, we include these figures for comparison with the systems participating in the Senseval-3 Restricted task (Litkowski, 2004).

We include baseline scores for the argument bracketing and classification tasks, respectively. The bracketing baseline method considers non-punctuation subtrees dependent of the target word. When the target word is a verb, the baseline puts FE brackets around the words included in each of these subtrees[1]. When the target is a noun, we also bracket the target word token itself, and when it is an adjective, we additionally bracket its parent token. As a baseline for the argument classification task, every argument is assigned the most frequent semantic role in the frame. As can be seen from the table, all scores except the argument bracketing recall are well above the baselines.

Precision (Strict scoring method)	0.67 ± 0.064
Recall	0.47 ± 0.057
Argument Classification Accuracy	0.75 ± 0.050
Baseline	0.41 ± 0.056
Argument Bracketing Precision	0.80 ± 0.055
Baseline Precision	0.50 ± 0.055
Argument Bracketing Recall	0.57 ± 0.057
Baseline Recall	0.55 ± 0.057
Precision (Senseval-3 scoring method)	0.77 ± 0.057
Overlap	0.75 ± 0.039
Recall	0.55 ± 0.057
Complete Sentence Accuracy	0.29 ± 0.073

Table 3: Results on the Swedish test set with approximate 95% confidence intervals.

Although the performance figures are better than the baselines, they are still lower than for most English systems (although higher than some of the systems at Senseval-3). We believe that the main reason for the performance is the quality of the data that were used to train the system, since the results are consistent with the hypothesis that the quality of the transferred data was roughly equal to the performance of the English system multiplied by the figures for the transfer method (Johansson and Nugues, 2005). In that experiment, the transfer method had a precision of 0.84, a recall of 0.81, and an F-measure of 0.82. If we assume that the transfer performance is similar for Swedish, we arrive at a precision of $0.71 \cdot 0.84 = 0.60$, a recall of $0.65 \cdot 0.81 = 0.53$,

[1] This is possible because MALTPARSER produces projective trees, i.e. the words in each subtree form a contiguous substring of the sentence.

and an F-measure of 0.56. For the F-measure, 0.55 for the system and 0.56 for the product, the figures match closely. For the precision, the system performance (0.67) is significantly higher than the product (0.60), which suggests that the SVM learning method handles the noisy training set rather well for this task. The recall (0.47) is lower than the corresponding product (0.53), but the difference is not statistically significant at the 95% level. These figures suggest that the main effort towards improving the system should be spent on improving the training data.

5 Conclusion

We have described the design and implementation of a Swedish FrameNet-based SRL system that was trained using a corpus that was annotated using cross-language transfer from English to Swedish. With no manual effort except for translating sentences for evaluation, we were able to reach promising results. To our knowledge, the system is the first SRL system for Swedish in literature. We believe that the methods described could be applied to any language, as long as there exists a parallel corpus where one of the languages is English. However, the relatively close relationship between English and Swedish probably made the task comparatively easy in our case.

As we can see, the figures (especially the FE bracketing recall) leave room for improvement for the system to be useful in a fully automatic setting. Apart from the noisy training set, probable reasons for this include the lower robustness of the Swedish parsers compared to those available for English. In addition, we have noticed that the European Parliament corpus is somewhat biased. For instance, a very large proportion of the target words evoke the STATEMENT or DISCUSSION frames, but there are very few instances of the BEING_WET and MAKING_FACES frames. While training, we tried to balance the selection somewhat, but applying the projection methods on other types of parallel corpora (such as novels available in both languages) may produce a better training corpus.

References

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley FrameNet Project. In *Proceedings of COLING-ACL'98*, pages 86–90, Montréal, Canada.

Chih-Chung Chang and Chih-Jen Lin, 2001. *LIBSVM: a library for support vector machines.*

Charles J. Fillmore. 1976. Frame semantics and the nature of language. *Annals of the New York Academy of Sciences: Conference on the Origin and Development of Language*, 280:20–32.

Pascale Fung and Benfeng Chen. 2004. BiFrameNet: Bilingual frame semantics resource construction by cross-lingual induction. In *Proceedings of COLING-2004*.

Daniel Gildea and Daniel Jurafsky. 2002. Automatic labeling of semantic roles. *Computational Linguistics*, 28(3):245–288.

Richard Johansson and Pierre Nugues. 2005. Using parallel corpora for automatic transfer of FrameNet annotation. In *Proceedings of the 1st ROMANCE FrameNet Workshop*, Cluj-Napoca, Romania, 26-28 July.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *MT Summit 2005*.

Krzysztof Kuchcinski. 2003. Constraints-driven scheduling and resource assignment. *ACM Transactions on Design Automation of Electronic Systems*, 8(3):355–383.

Ken Litkowski. 2004. Senseval-3 task: Automatic labeling of semantic roles. In *Senseval-3: Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, pages 9–12.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Sameer Pradhan, Kadri Hacioglu, Valerie Krugler, Wayne Ward, James Martin, and Dan Jurafsky. 2005a. Support vector learning for semantic argument classification. *Machine Learning*, 60(1):11–39.

Sameer Pradhan, Wayne Ward, Kadri Hacioglu, James Martin, and Daniel Jurafsky. 2005b. Semantic role labeling using different syntactic views. In *Proceedings of ACL-2005*.

David Yarowsky, Grace Ngai, and Richard Wicentowski. 2001. Inducing multilingual text analysis tools via robust projection across aligned corpora. In *Proceedings of HLT 2001*.