The Dependency-Parsed FrameNet Corpus

Daniel Bauer, Hagen Fürstenau, Owen Rambow

Columbia University
New York, NY 10024, USA
bauer@cs.columbia.edu, {hagen, rambow}@ccls.columbia.edu

Abstract

When training semantic role labeling systems, the syntax of example sentences is of particular importance. Unfortunately, for the FrameNet annotated sentences, there is no standard parsed version. The integration of the automatic parse of an annotated sentence with its semantic annotation, while conceptually straightforward, is complex in practice. We present a standard dataset that is publicly available and that can be used in future research. This dataset contains parser-generated dependency structures (with POS tags and lemmas) for all FrameNet 1.5 sentences, with nodes automatically associated with FrameNet annotations.

Keywords: frame semantics, semantic resource, dependency parsing

1. Introduction

FrameNet (Fillmore et al., 2003) is a lexical resource for English, based on the theory of Frame Semantics (Fillmore, 1976). It comprises both a lexicon and a corpus of example sentences, in which certain words are identified as *frame evoking elements* (FEEs) and annotated with a semantic frame. Such frames represent prototypical situations or events, such as COMMERCIAL_TRANSACTION, and may feature one or more *frame elements* (FEs, sometimes also called *semantic roles*), such as BUYER or SELLER. In its most recent version 1.5, FrameNet contains manual annotations for more than 170,000 sentences. Figure 1 shows an example of a semantically annotated sentence. These annotations have been used to train semantic role labeling (SRL) systems, which are then able to derive frame semantic analyses, i.e., identify frames and frame elements in any given sentence.

Syntactic structure is of crucial importance to the training and application of SRL systems, which typically make use of it to learn and infer semantic structure. Indeed, starting with Gildea and Jurafsky (2002), most SRL systems have made extensive use of syntactic parses as the basis of their semantic predictions, e.g., in the form of features used in various classification algorithms (e.g. Moschitti et al. (2008), Johansson and Nugues (2007), Das and Smith (2011)). These features have been shown to improve SRL performance. In addition to characterizing syntactic realizations of predicate-argument structure, syntactic information is also necessary to extract the head word of an argument phrase. Features extracted for this head are commonly used to model the meaning of the argument and thus its compatibility with a particular role.

Unfortunately the majority of FrameNet annotations is taken from the British National Corpus, for which there exists no standard syntactic annotation. Instead, as can be seen from the example in Figure 1, FrameNet provides only shallow syntactic information, indicating parts of speech, phrase types, and limited information on grammatical functions. FrameNet annotates FEs on surface text spans, instead of syntactic constituents, and annotations do not mark the head word of an annotated FE.

Consequently, SRL work in the FrameNet paradigm often uses a syntactic parser to automatically produce syntactic analyses for the sentences in the FrameNet corpus. These parses then need to be aligned with the provided semantic annotations. For instance, Figure 2 shows the annotation from Figure 1 projected onto a parser-generated dependency structure. FEEs and FEs have been identified with nodes in the parse tree. Aligning syntactic analyses produced by a parser with the semantic annotation provided by FrameNet, while seemingly straightforward, is complicated by various issues, including parser errors, discontinuous frame elements, inconsistent FrameNet annotations, and technical issues such as character encodings. However, while this alignment task has to be performed repeatedly by many researchers and potentially has a significant impact on the performance of SRL systems, it has rarely been explained and evaluated in detail. This makes it harder than necessary to reproduce work on SRL within the Frame Semantic paradigm. In addition, differences in parser performance, syntactic representations, and alignment techniques limit the comparability of reported SRL results.

The aim of the present work is threefold:

1. In Section 3, we present in detail a simple but effective method of aligning FrameNet annotations with syntactic parses. We decided to restrict ourselves to dependency trees which have enjoyed increasing popularity in SRL research (Hacioglu, 2004; Johansson and Nugues, 2007; Surdeanu et al., 2008).

2. In Section 4, we evaluate this method on a small subset of the FrameNet corpus, for which we manually annotated the head words of FEs, and tune some of its parameters.

3. We apply the method to the entire FrameNet 1.5 corpus to produce a standard dataset that is publicly available and can be used in future research.[1] This dataset

[1] The dataset is available at `http://www1.ccls.columbia.edu/~rambow/resources/parsed_framenet/`.

	She	wrinkled	her	nose	in	disapproval	.
Frame:		Body_movement					
FE:	Agent		Body_part		Internal_cause		
POS:	PNP	VVD	DPS	NN1	PRP	NN1	PUN
PT:	NP		NP		PP		
GF:	Ext		Obj		Dep		

Figure 1: Example FrameNet annotation. FEs and FEEs are annotated over text spans. Additional information about parts of speech (POS), phrase types (PT), and grammatical functions (GF) is provided.

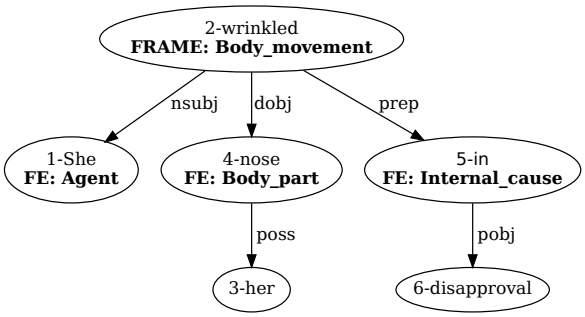

Figure 2: FrameNet annotation projected onto a dependency graph produced by an automatic parser. FEs and FEEs are assigned to nodes in the parse tree.

contains dependency parses (with POS tags and lemmas) for all FrameNet 1.5 annotations, including per-dependency-node annotation of FEEs and FEs. We also include the split into training and test data that was used to evaluate a recent state-of-the-art SRL systems for FrameNet (Das and Smith, 2011). Section 5 describes this resource in detail.

We conclude in Section 6.

2. Related Work

The basic problem of parsing FrameNet sentences and aligning the resulting parses with the semantic annotations had to be addressed by most SRL systems working with the FrameNet data. In contrast, the PropBank corpus (Palmer et al., 2005) is based on the Penn Treebank and therefore already provides hand-annotated syntax, so that a similar problem does not arise. However, it follows a different paradigm of semantic roles (Rambow et al., 2003). Furthermore, while PropBank annotates only verbs, the FrameNet lexicon also contains frame evoking nouns, adjectives, adverbs, and prepositions.

An explicit description of different algorithms for the task at hand was previously given in (Fürstenau, 2008). However, that approach was limited to annotations of verbal FEEs. The present work provides more extensive evaluation results, confirming the robustness of our approach, reports on parameter tuning experiments, and is accompanied by a publicly available resource.

3. Method

In this section, we describe our approach of merging dependency information provided by a syntactic parser and Frame Semantic annotation from the FrameNet corpus into a new resource. The resulting resource is a corpus of *semantically annotated dependency graphs*, i.e., syntactic dependency graphs in which certain nodes are marked as frame evoking elements (FEEs) or frame elements (FEs) of specific semantic frames.

While describing the various processing steps involved in creating a unified resource, we will leave certain parameters unspecified. These will be empirically optimized in the following Section 4. In Section 5, we will then describe technical details of the resulting publicly available resource.

3.1. Parsing

FrameNet contains the tokenized plain texts of each annotated sentence. We extract these sentences and parse them with the Stanford Parser (Klein and Manning, 2003)[2], setting it up to produce dependency analyses. In particular, we consider Stanford Dependencies (de Marneffe et al., 2006) in either of the two modes `basicDependencies` and `CCPropagatedDependencies`. The main difference between these are that the latter "collapses" prepositions into edge labels and does not guarantee that the dependency graph is a tree (it may not even by acyclic), both of which can allow the syntactic dependencies to better reflect the semantics of a sentence. Since there are advantages and disadvantages to either dependency representation, we experimented with both, and offer two versions of the final resource, built on either kind of syntax.

Instead of only considering the most probable parse, we let the parser output the 50 best parses of each sentence. In Section 3.3.3, we will describe how our method makes use of competing syntactic analyses. Section 4 then shows the impact of the number n of considered parses ($n \leq 50$) on the quality of the resulting resource.

3.2. Preprocessing

In preparation for the merging of syntactic parses and semantic annotation, we transform the FrameNet annotation, which is based on character indices of substrings, into token-based annotation. Since the sentences in FrameNet are already tokenized, we simply split each sentence string at white space characters and number the resulting tokens, starting at 1. Each FEE or FE in FrameNet comes annotated as a (not necessarily continuous) subset of the characters of the sentence. We convert such an annotation into a set of token numbers by considering each token whose set

[2]version 2.0.1, available at `http://nlp.stanford.edu/software/lex-parser.shtml`.

of characters is completely contained within the set of annotated characters. For example, the annotation of the FE *Body_part* in Figure 1 would be converted from the index set $\{13, \ldots, 20\}$ into the token number set $\{3, 4\}$.

3.3. Annotation Mapping and Parse Selection

For each sentence in the FrameNet corpus, we consider its frame annotation and the n best syntactic parses. Our goal is to select one of these parses, and for each annotated frame to identify one dependency graph node as the FEE and one node as the head word of each FE, as illustrated in Figure 2. We reach both of the goals at the same time by *mapping* the semantic annotations onto the n best parses one after another (starting with the best parse). While ideally the syntactic parse would be completely compatible with the semantic annotation, we find that due to various reasons the match is not perfect in every case. We therefore measure the quality of the mapping (in terms of precision and recall, as detailed below) and select the parse that is most compatible with the semantic annotation. The intuition here is that the manual semantic annotation is able to implicitly resolve syntactic ambiguities, such as the correct attachment of prepositional phrases, and thereby guide our process of parse selection. This will be confirmed in our experiments in Section 4. In the following, we now describe separately how we map FEEs and FEs onto a given dependency graph.

3.3.1. Mapping the FEE

For each annotated frame, exactly one graph node is to be marked as its FEE. In most cases, this is straightforward, since the annotation of the FEE consists of only a single token, which is represented by a single node in the dependency graph.

However, it may happen that no graph node corresponds to the annotated FEE. This is the case for frame evoking prepositions if the parser is producing dependency graphs in mode `CCPropagatedDependencies`, which removes prepositions from the dependency graph and includes (collapses) them into edge labels. We skip such frame, as there is no obvious way of representing them on the given syntactic analyses. However, they are included in the case of `basicDependencies`.

It may also happen that the annotated FEE comprises more than one dependency graph node. This usually happens in the case of particle verbs (such as *break up*) or compound nouns (such as *brand name*). In this case, we choose the node that dominates most of the others within the set of the FEE tokens. In case of a tie between such nodes, we choose the first of them in linear sentence order. This simple heuristic takes care of the frequent cases of particle verbs and compound nouns (by the dominance criterion) and is robust in most other cases.

3.3.2. Marking FEs by Finding Head Words

For each FE of each annotated frame, our goal is to identify a single dependency graph node that represents its syntactic head. Unfortunately, FrameNet does not annotate head words of FEs, but only their whole textual spans. For a given FE, we therefore consider each node of the given dependency graph in turn and compare the set of nodes (directly or indirectly) dominated by it to the set of annotated FE tokens. Ideally, we would find exactly one node (the syntactic head of the FE) that dominates exactly the annotated FE tokens. For various reasons (mostly consisting of parse errors), it is not always possible to find such a node. For example, if in Figure 2 the parser had erroneously attached the prepositional phrase *in disapproval* under the head *nose*, then there would be no node that dominates exactly the two words *her* and *nose*. We therefore search for the node that matches the given set of FE tokens *as closely as possible*. To measure this fit, we employ precision and recall of the set of dominated nodes with respect to the set of FE tokens. Specifically, we compute a weighted F_β score between precision (P) and recall (R), with the optimal value of β to be determined in Section 4:

$$F_\beta = (1 + \beta^2) \cdot \frac{P \cdot R}{\beta^2 P + R}$$

We use the following algorithm to select a graph node for a given set of FE tokens. In comparison to a naive search in arbitrary order, it is more efficient, more robust to cyclic dependency graphs, and incorporates the intuition that in case of ties certain positions of the FEE relative to the FE should be preferred:

1. If the set of FE tokens is empty, skip this FE. (This happens in cases of FEs that are not textually realized. We do not attempt to include these in the resulting resource.)
2. If the set of FE tokens has exactly one item, this is the head word.
3. Otherwise, start at the FEE node (identified before) and recursively traverse its dependents and their dependents, but never recurse into a node that has already been visited. Find the node that maximizes F_β.
4. Repeat (3), now starting from the direct predecessors[3] of the FEE node, their direct predecessors, and so on (recursively).

The head of the FE is the first node that maximizes F_β. In case of ties, the order of graph traversal therefore has the effect of preferring nodes dominated by the FEE over others, and nodes dominated by closer predecessors of the FEE over those dominated by more distant predecessors. In addition, we always prefer direct dependents of the FEE, as direct dominance should be the most common syntactic relation between a predicate and its semantic arguments.

3.3.3. Scoring Mappings and Selecting Parses

To score the mapping of the semantic annotation onto a given parse, we compute a score for each frame, averaging over all its FEs, and then compute the average of these scores over all frames of the sentence. This results in a mapping score that quantifies the compatibility between the semantic annotation of a given sentence and any given parse. We therefore select the parse with the highest score, preferring earlier ones (i.e., more probable ones according to the parser) over later ones in case of ties.

[3] There may be multiple direct predecessors since the dependency graphs are not guaranteed to be trees.

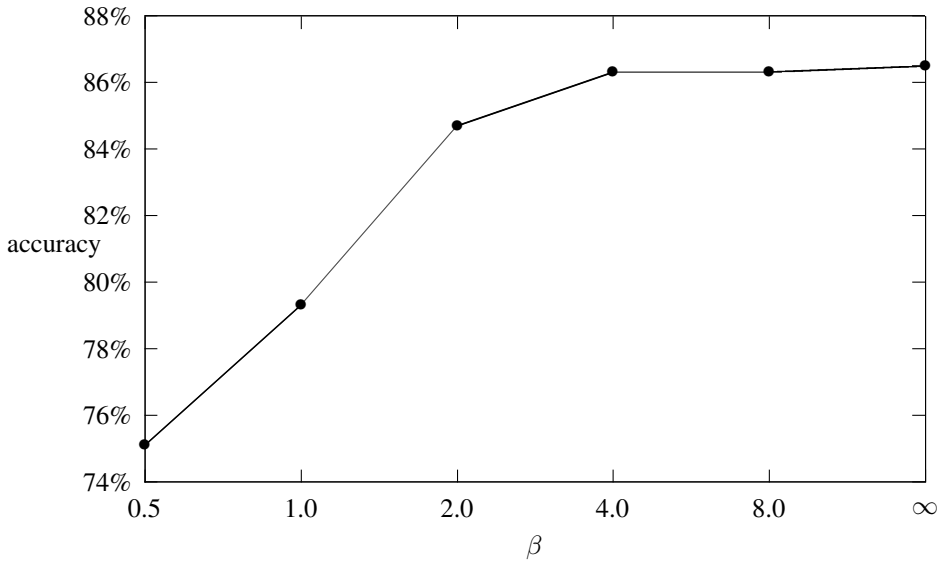

Figure 3: Accuracy of head identification compared to human annotations for different values of β.

4. Evaluation

When mapping FEs to nodes in the dependency structure, our method attempts to ensure that the surface string spanned by the subtree under a node corresponds to the phrase selected by FrameNet annotators as closely as possible. We wish to evaluate how successful our method is in identifying the head of an FE or FEE in the FrameNet corpus. As mentioned above, correctly identifying heads is crucial because the meaning of an FE can be approximated by the lexical meaning of the head word. In addition, evaluation schemes for frame semantic parsing, such as the one used in the SemEval'07 shared task (Baker et al., 2007), are based on 'semantic dependencies', so that correctly identifying the overall semantic dependency structure is more important than identifying exact text spans. It is therefore necessary that the training data contains correct heads words. We consider syntactic dependencies a good approximation to semantic dependencies.

To evaluate how well our method identifies head words, the three authors manually annotated syntactic and semantic heads for all FEEs and FEs in a set of frame annotations. We established the following guidelines for frequent cases requiring principled annotation decisions:

- For prepositional, phrases we annotated the complement of the preposition, as our evaluation was carried out on the parses produced in `CCPropagatedDependencies` mode. The same rule applied to *by*-phrases in passive constructions.
- The head of coordinations is the first conjunct. This is consistent with the Stanford Decency representation of coordinations.
- The head of proper names of individuals is their family name.
- In cases were the syntactic head differs from the semantic head, we annotated the syntactic one.

For instance, for the *Message* FE in the following example

> *[She]$_{Speaker}$ expressed a warm opinion of the piece and asked$_{\mathbf{Request}}$ [for more of her work]$_{Message}$.*

we annotated the syntactic head *more* (according to the first rule), and not the semantic head *work* (according to the last rule).

We randomly sampled 400 sentences from the lexicographic part of FrameNet 1.5, which contain one frame annotation each. Each annotator worked on 100 sentences. An additional set of 100 sentences was annotated by all three. We found inter-annotator agreement on these sentences to be high (94% average pairwise agreement). The final evaluation set contained the majority annotation for the common sentences and the 300 sentences with a single annotator.

Given the set of sentences with manual head annotations, we can evaluate the output of different versions of our method. We present two experiments, showing the effect of two different parameters: the parameter β in the F_β score used to score the fit between semantic annotation and syntactic analysis, and the number n of syntactic analyses of each sentence considered for reranking (the n-best analyses proposed by the parser). We perform these experiments using the `CCPropagatedDependencies` option of the Stanford Parser.

Figure 3 shows the influence of the parameter β, ranging from 0.5 to 8.0 on a logarithmic scale, on the accuracy of the head identification. Observing that the accuracy grows monotonically, we also show the limit case of $\beta = \infty$, i.e., choosing the head node to maximize recall. The parameter n was fixed at $n = 1$ for this experiment, i.e., we only considered the first-best parse instead of selecting the parse that produced the highest mapping score. The experiment shows that the optimal value of β is in fact $\beta = \infty$. In the creation of our resource, we therefore ignore the precision score of the match between a graph node and the

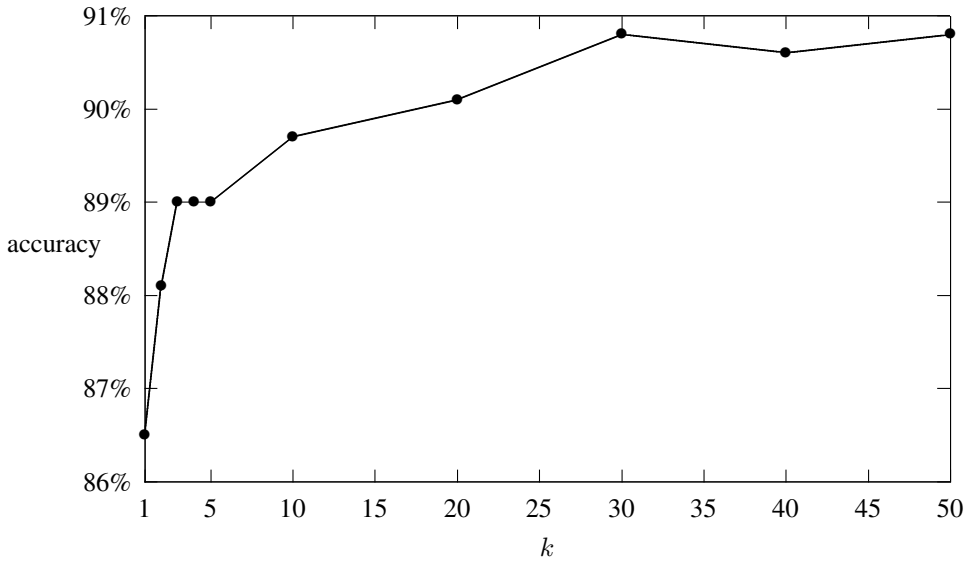

Figure 4: Accuracy of head identification compared to human annotations for different values of k ($\beta = \infty$) .

FrameNet annotation and use only recall for scoring. The high accuracy with $\beta = \infty$ shows that the parser is generally good at attaching all relevant words within an FE or FEE as descendants below the head word, but this subtree may include other irrelevant lexical material as well.
In Figure 4, we similarly show the influence of the number of alternative parses suggested by the parser on head identification accuracy. As can be seen, choosing the best-fitting parse among the n-best parses, i.e. the parse that maximizes the overall mapping score, strongly increases the accuracy of the identified head words over always choosing the most likely parse. Up to $n = 30$ accuracy increases with n, while we do not observe further improvement for a larger number of parses. In the creation of our resource, we therefore consider the 30 most likely parses proposed by the parser for each sentence.

5. Resource

We apply the method described in Section 3 to the full lexicographic corpus and the fulltext annotations of FrameNet 1.5. The result is represented in a variant of the column-based format used by the CoNLL-2008 Shared Tasks on dependency parsing (Surdeanu et al., 2008): Each word token is represented by a single line, with consecutive sentences in the same file separated by empty lines. Each line consists of at least the following 10 fields, separated by tabulator characters:

1. sequential token ID, starting at 1 for each sentence
2. word form
3. lemma of the word (according to TreeTagger[4])
4. POS tag included in FrameNet
5. POS tag according to TreeTagger

[4]available at `http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/`

6-8. three empty fields for compatibility with the CoNLL-2008 format

9. IDs of the syntactic heads
10. grammatical relations to syntactic heads

The following fields contain the semantic annotation. For each annotated frame of the sentence, there are two additional columns, the first one marking its FEE and the second one marking all the FEs. This representation makes it possible for one token to be the FEE or FE of different frames. Figure 5 shows our example sentence from Figures 1 and 2 in this format. The resource is published in two versions, with either `basicDependencies` or `CCPropagatedDependencies`.
For each sentence in the corpus, we additionally provide information on the rank of the parse (n-best) selected by our algorithm, the mapping score, and a random sentence ID (RID). The latter may function as a convenient way for future work using the provided resource to uniquely characterize a specific subset of the sentences, e.g., as training or test set in semantic role labeling. Since these IDs were randomly shuffled over the whole corpus, a random sample could be specified, e.g., as "RIDs 1–1000" and be reproduced by anyone working with our resource.
In addition to the parsed dataset itself we provide the split of the FrameNet fulltext annotations used by Das and Smith (2011) to evaluate their SRL system. This data is provided as stand-off annotations. We also include a mapping from RIDs to FrameNet sentence IDs.
Because the Stanford Parser produces output that is tokenized and normalized according to the Penn Treebank convention, our data set differs from the FrameNet plaintext. Although evaluations for FrameNet style SRL are based on semantic dependency structures (Baker et al., 2007), evaluation scripts build these structures internally from frame element text spans. To be able to evaluate SRL systems trained on our data set we provide a script to re-align parsed

ID	FORM	LEMMA	FN-POS	POS	–	–	–	HEAD	DEPREL	FRAME	FE
1	She	she	PNP	PRP	_	_	_	2	nsubj	_	Agent
2	wrinkled	wrinkel	VVD	VBD	_	_	_	_	_	Body_movement	_
3	her	her	DPS	PRP$	_	_	_	4	poss	_	_
4	nose	nose	NN1	NN	_	_	_	2	dobj	_	Body_part
5	in	in	PRP	IN	_	_	_	2	prep	_	Internal_cause
6	disapproval	disapproval	NN1	NN	_	_	_	5	pobj	_	_
7	.	.	PUN	.	_	_	_	_	_	_	_

Figure 5: The sentence from Figure 2 as it appears in the final resource (in the version with `basicDependencies`).

FrameNet with the original annotations, which generates new FrameNet fulltext annotations with our preprocessing.

6. Conclusion

This paper presented an effective method for enriching the frame annotations of FrameNet 1.5 with dependency structures generated by the Stanford Parser. Our algorithm maps each frame evoking element or frame element to a node in the dependency parse. It chooses the dependency node that maximizes the F_β-score between the tokens contained within an annotation span and the nodes within a subtree of the dependency parse. We evaluate how effectively this method selects the correct syntactic head word of each annotation span, as defined by human annotators. We find that setting $\beta = \infty$, i.e. considering only token recall, performs best. We also use the mapping algorithm for parse selection. From the set of n-best parses proposed by the dependency parser we select the parse that maximizes the average F_β score over all selected nodes. Our experiments show that the optimal value for n is 30. With these best parameters we reproduce hand-annotated syntactic heads with an accuracy of close to 91%.

The resulting resource, which also contains POS tags, lemmas, and a standard split into training and test data, is publicly available for download. We hope that Parsed FrameNet will make results obtained by different researcher more comparable, and that it will facilitate and ease new work on semantic role labeling within the FrameNet paradigm.

7. Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant No.IIS-0904361 and the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior National Business Center (DoI/NBC) contract number D11PC20153. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DoI/NBC, or the U.S. Government.

8. References

C. Baker, M. Ellsworth, and K. Erk. 2007. Semeval'07 task 19: frame semantic structure extraction. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 99–104. Association for Computational Linguistics.

Dipanjan Das and Noah A. Smith. 2011. Semi-supervised frame-semantic parsing for unknown predicates. In *Proceedings of the 49th Meeting of the Association for Computational Linguistics*, pages 1435–1444.

Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of LREC 2006*.

Charles J. Fillmore, Christopher R. Johnson, and Miriam R. L. Petruck. 2003. Background to FrameNet. *International Journal of Lexicography*, 16:235–250.

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 and Speech*, 280:20–32.

Hagen Fürstenau. 2008. Enriching frame semantic resources with dependency graphs. In *Proceedings of the 6th Language Resources and Evaluation Conference*, pages 1478–1484, Marrakech, Morocco.

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

K. Hacioglu. 2004. Semantic role labeling using dependency trees. In *Proceedings of the 20th international conference on Computational Linguistics*, page 1273. Association for Computational Linguistics.

Richard Johansson and Pierre Nugues. 2007. Lth: semantic structure extraction using nonprojective dependency trees. In *Proceedings of of SemEval*.

Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st Meeting of the Association for Computational Linguistics*, pages 423–430.

A. Moschitti, D. Pighin, and R. Basili. 2008. Tree kernels for semantic role labeling. *Computational Linguistics*, 34(2):193–224.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The Proposition Bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106.

Owen Rambow, Bonnie Dorr, Karin Kipper, Ivona Kučerová, and Martha Palmer. 2003. Automatically deriving tectogrammatical labels from other resources: A comparison of semantic labels across frameworks. *The Prague Bulletin of Mathematical Linguistics*, (79–

80):23–36.

Mihai Surdeanu, Richard Johansson, Adam Meyers, Lluís Màrquez, and Joakim Nivre. 2008. The CoNLL 2008 shared task on joint parsing of syntactic and semantic dependencies. In *Proceedings of CoNLL 2008*, pages 159–177, Manchester, England.