A Web-based Tool for the Integrated Annotation of Semantic and Syntactic Structures

Richard Eckart de Castilho[†◇] and **Éva Mújdricza-Maydt**[*◇] and **Seid Muhie Yimam**[‡] and **Silvana Hartmann**[†] and **Iryna Gurevych**[†◇] and **Anette Frank**[*◇] and **Chris Biemann**[‡◇]

[†]**Ubiquitous Knowledge Processing Lab**
Department of Computer Science
Technische Universität Darmstadt

[‡]**FG Language Technology**
Department of Computer Science
Technische Universität Darmstadt

[*]**Department of Computational Linguistics**
Heidelberg University

[◇]**Research Training Group AIPHES**
Heidelberg University and
Technische Universität Darmstadt

Abstract

We introduce the third major release of WebAnno, a generic web-based annotation tool for distributed teams. New features in this release focus on semantic annotation tasks (e.g. semantic role labelling or event annotation) and allow the tight integration of semantic annotations with syntactic annotations. In particular, we introduce the concept of *slot features*, a novel *constraint mechanism* that allows modelling the interaction between semantic and syntactic annotations, as well as a new annotation user interface. The new features were developed and used in an annotation project for semantic roles on German texts. The paper briefly introduces this project and reports on experiences performing annotations with the new tool. On a comparative evaluation, our tool reaches significant speedups over WebAnno 2 for a semantic annotation task.

1 Introduction

As natural language processing pushes towards natural language understanding, i.e. the ability for a machine to process the meaning of language rather than just its structure, there is a growing need for corpora analysed on the semantic level. Semantic structures pose different requirements for annotation tools than morphological or syntactic annotation. For example, the rich sets of semantic categories used in tasks such as semantic role labelling (SRL) or event annotation require special support to avoid the choice becoming a burden to the annotator. Also, due to the interaction of semantics with other levels of linguistic analysis, particularly syntax, it is desirable to work with generic annotation tools that simultaneously support multiple levels of annotation.

Generic web-based annotation tools do not sufficiently support the annotation of semantic structures with their rich tagsets. Also, specialised annotation tools, in particular for SRL, are not flexibly adaptable to other annotation schemes, and many of them are technologically outdated.

WebAnno 3[1] is the third major release of the web-based annotation tool WebAnno (Yimam et al., 2013; Yimam et al., 2014) introducing new functionalities enabling the annotation of semantic structures:

1. **Slot-features** allow the appropriate modelling of predicate-argument structures for SRL. We also support the following additional semantic annotation types: participants and circumstances for event annotation, *n*-ary relations for relation extraction, and slot-filling tasks for information extraction.
2. **Constraints** help annotators by performing a context-sensitive filtering of the rich semantic tagsets. For example, the sense of a semantic predicate determines available argument roles. Such a filtering is necessary to avoid loosing valuable time by having annotators search through a large number of tags or to manually type in tags. Constraint rules can be defined manually or they can be automatically

[1] https://webanno.github.io/webanno/

Proceedings of the Workshop on Language Technology Resources and Tools for Digital Humanities (LT4DH), pages 76–84, Osaka, Japan, December 11-17 2016.

generated, e.g. from machine-readable lexical resources. To our knowledge, there is no other web-based annotation tool offering a comparable functionality.

3. **An improved annotation interface** for a streamlined annotation process using a permanently visible sidebar instead of a pop-up dialog for editing annotations and their features.

These new functionalities integrate well with the existing functionalities in WebAnno 2, in particular its support for the annotation of syntactic structures, thus enabling semantic annotation in coordination with syntactic annotation. To our knowledge, WebAnno 3 is presently the only web-based and team-oriented annotation tool to support both, the annotation of semantic as well as syntactic structures. This includes, but is not limited to the tools mentioned in Section 2.

WebAnno 3 was developed and implemented in close coordination with users in the context of an annotation project (cf. Mújdricza-Maydt et al. (2016)) for word sense disambiguation (WSD) and SRL on German texts and driven by its practical requirements. SRL is the task of identifying semantic predicates, their arguments, and assigning roles to these arguments. It is a difficult task usually performed by experts. Examples of well-known SRL schemes motivated by different linguistic theories are FrameNet (Baker et al., 1998), PropBank (Palmer et al., 2005), and VerbNet (Kipper Schuler, 2005). SRL annotation is typically based on syntactic structures obtained from treebanks, such as the constituent-based Penn Treebank (for PropBank annotation), or the German TIGER treebank for FrameNet-style annotation (Burchardt et al., 2009). An argument is typically identified by the span of its syntactic head or syntactic constituent. For some annotation schemes (e.g. FrameNet), the task also includes WSD. In this case, the sense label typically determines the available argument slots. The example below shows an annotation using FrameNet; the predicate *ask* receives the frame label *Questioning* (corresponding to its word sense) and its arguments are annotated as *Addressee*, *Speaker*, *Message*, and *Iterations*:

(1) Fred$_{Role:Addressee}$ didn't answer.
While I$_{Role:Speaker}$ had asked$_{Frame:Questioning}$ this question$_{Role:Message}$ twice$_{Role:Iterations}$ before.

Note that there are multiple interdependencies between annotations involved here. E.g. the available sense labels for the semantic predicate depend on its lemma. Also, the available argument roles depend on the sense label. We further describe the annotation project and the application of WebAnno 3 within the project in Section 4. While joint WSD and SRL annotations are conveniently supported and facilitated using constraints, they can also be performed separately and independently of one another.

2 Related Work

We briefly review presently available annotation tools that could be used for annotating semantic structures. Since we aim to support geographically distributed annotation teams, we consider recent generic web-based annotation tools. Additionally, we examine tools from earlier semantic annotation projects that are specialised for SRL but not web-based.

2.1 Web-based Annotation Tools

Anafora by Chen and Styler (2013) is a recent web-based annotation tool for event-like structures. Specifically, it supports the annotation of spans and *n*-ary relations. Spans are anchored on text while relations exist independently from the text and consist of slots that can be filled with spans. Annotations are visualised using a coloured text background. Selecting a relation highlights the participating spans by placing boxes around them. Anafora is not suited for annotation tasks that require an alignment of the semantic structures with syntactic structures such as constituent or dependency parse trees.

brat by Stenetorp et al. (2012) is another web-based annotation tool with a focus on collaborative annotation. The tool supports spans and *n*-ary relations (also called *events*). Annotations are visualised as boxes and arcs above the text. Multiple annotators can simultaneously work on the same annotations instead of being isolated from each other. However, this removes the ability to calculate inter-annotator agreement. All annotation actions are performed through a pop-up dialog, which necessitates many actions even for simple annotations. While in principle the support for semantic annotation in brat through

the *n*-ary relations is good, there is no support for guiding the user through rich semantic tagsets, e.g. by showing only applicable tags based on the annotation's context.

WebAnno 2 (Yimam et al., 2014), for which we present an updated version here, is a generic and flexible annotation tool for distributed teams. To visualise the text and annotations, it uses the JavaScript-based annotation visualisation from brat.

The annotation of semantic structures can only be realised with great difficulty in WebAnno 2. It requires the creation of a custom span layer to represent semantic predicates and arguments. An additional custom relation layer is required to connect the arguments to the predicate. The distinction between predicates and arguments needs to be made through a feature on the span layer (e.g. *semanticType=pred|arg*), because the use of two distinct span layers for predicates and their arguments is not possible. The reason is that WebAnno 2 only allows relations to be created between annotations on the same layer. With such a setup, annotators have to take great care not to create invalid semantic structures, e.g. by linking predicates to other predicates or arguments to other arguments while the annotation guidelines only allow links between predicates and arguments. This complicates and slows down the annotation process and requires post-hoc consistency checks. Finally, WebAnno 2 offers no provisions to guide annotators through rich semantic tagsets. Due to the complex setup, annotating semantic structures also requires too many user interactions (*clicks*), making it also a very tedious task. This problem is aggravated by an annotation dialog popping up for each action.

2.2 Semantic Role Annotation Tools

SALTO by Burchardt et al. (2006) supports the annotation of constituency treebanks with FrameNet categories. Once a frame for a predicate has been selected, applicable roles from FrameNet can be assigned to nodes in the parse tree by drag-and-drop. SALTO supports discontinuous annotations, multi-token annotations, and cross-sentence annotations. It also offers basic team management functionalities including workload assignment and curation. However, annotators cannot correct mistakes in the underlying treebank because the parse tree is not editable. This is problematic for automatically preprocessed input. The final release of SALTO was in 2012.

Jubilee and Cornerstone by Choi et al. (2010) are tools for annotating PropBank. Jubilee supports the annotation of PropBank instances, while its sister tool Cornerstone allows editing the frameset XML files that provide the annotation scheme to Jubilee. The user interface (UI) of Jubilee displays a treebank view and allows annotating nodes in the parse tree with frameset senses and roles. Jubilee supports annotation and adjudication of annotations in small teams. It is a Java application that stores all data on the file system. Thus, the annotation team needs to be able to access a common file system, which does not meet our needs for a distributed team. Both tools appear to be no longer being developed since 2014.

2.3 Requirements of Semantic Annotation

The annotation of semantic structures imposes two main requirements on annotation tools: 1) the flexibility to support multiple layers of annotation including syntactic and semantic layers using freely configurable annotation schemes and 2) the ability to handle large, interdependent tagsets.

Flexible multi-layer annotation. While the usage-driven design of dedicated SRL annotation tools allows for a very efficient annotation, users face a serious lack of flexibility when trying to combine different annotation schemes (e.g. GermaNet senses (Hamp and Feldweg, 1997) and VerbNet roles), or when trying to use data preprocessed in different ways (e.g. for a crowdsourcing approach, automatically pre-annotating predicate and argument spans can be helpful, while experts may find pre-annotated dependency relations beneficial). This is not supported by current web-based annotation tools.

Handling rich annotation schemes. Tools need to specifically support rich semantic annotation schemes—like FrameNet—with interdependent labels (i.e. sense labels determine available argument roles). Manually typing sense and role labels is error-prone, and selecting them from a long list is cumbersome for the annotator. The tool must guide the annotator, e.g. by filtering or reordering possible sense labels based on the lemma and do the same for role labels based on the selected sense. E.g. SALTO only offers frames in the spectrum of the lemma under annotation.

3 Semantic Annotation using WebAnno 3

To meet the requirements described above, WebAnno 3 introduces support for arbitrary semantic role labelling schemes by geographically distributed teams. The great challenge in building WebAnno 3 was to identify a way of implementing support for semantic annotation not only without adversely affecting existing features of WebAnno 2, but also by capitalising on the existing features. The data model for annotations used by WebAnno is based on *feature structures*, i.e. typed key-value pairs. First, our tool adds a new type of features for use in these structures, namely *slot features*, that enable SRL and other slot-filling annotation tasks. Next, we introduce *constraint rules* to allow the modelling of tag interdependencies. Constraint rules can be used not only to provide context-sensitive assistance for semantic annotations, but in general for all kinds of annotations supported by WebAnno 3, including those inherited from WebAnno 2. Finally, an improved and streamlined user interface was introduced to enable annotation to be performed without disruptive pop-up dialogs.

3.1 Semantic Annotation using Slot Features

WebAnno provides a flexible configuration system for annotation layers supporting spans and relations. However, as outlined in Section 2.1, the provided features are insufficient for semantic annotation.

WebAnno 3 introduces a new feature type into the feature-structure-based system—the so-called *slot features*—to support semantic annotations like SRL. Slot features can be added to span layers and allow one span annotation (*slot owner*) to *link* to one or more other span annotations (*slot fillers*). Each link is qualified with a *role*, which may optionally be restricted by a tagset. Slot features can be configured to allow arbitrary slot fillers or only such pertaining to a specific layer. Figure 1 illustrates the annotation of word sense and semantic roles for *ask* as introduced in Example (1).

Declaration. We model the semantic predicate using a span layer *SemPred* with a slot feature called *arguments*. A second span layer—*SemArg*—represents the slot fillers.[2]

Interaction. To use the slot feature, the user first creates the slot-owning annotation, here a *SemPred* annotation. Then, the user selects a role to add a new slot to the feature *arguments*. Selecting a span then automatically creates the slot filler annotation (*SemArg*) and completes the link.

Visualisation. The links between slot owner and slot fillers are visualised as arcs in the colour of the slot owner. A special dashed style is used to distinguish them from other types of relations.

Agreement. With the introduction of slot features, we completely reimplemented the calculation of inter-annotator agreement. A single feature of a layer can be selected for agreement calculation (i.e. the word sense feature or the semantic argument feature). The first step in the calculation of agreement identifies which annotations actually need to be compared to each other. To address this, we first derive *coordinates* from the annotated data (Table 1). For regular span features, a coordinate is the tuple of <layer, feature, char offsets(begin-end)> and for relation features <layer, feature, char offsets(srcBegin-srcEnd, tgtBegin-tgtEnd)>. Then, agreement is calculated by comparing the labels assigned by each user for each coordinate.

We implemented two modes of calculating agreement for slot features. The first mode *role-as-label* calculates agreement based on the slot's *role*. Consider *asked* as the semantic predicate and *Fred* as a slot filler. Two annotators agree if they assigned the same role to *Fred*, e.g. *Addressee*. It is a disagreement if one annotator assigns the role *Speaker* to Fred and another assigns the role *Addressee* (cf. Table 1 d/e). In this mode, the slot filler is part of the annotation coordinates (cf. Table 1 column *extra*).

In the second mode, *target-as-label*, agreement is calculated based on the slot filler. Two annotators agree if they use the same slot filler for a slot with a given role. So consider again *asked* as the semantic predicate and *Addressee* as the role to be filled. In this case, there is an agreement if two annotators choose *Fred* as a slot filler and a disagreement if one annotator fills the slot with *Fred* and the other one with *I*. In this mode, the role label is part of the annotation coordinates (cf. Table 1 column *extra*).

Missing annotations are very common with slot features. For example, cases d) and e) in Table 1 differ in their coordinates, namely in the *extra* column which is used to discriminate between slots, but not in

[2]The layer names proposed here are meant as examples motivated by our use case. In WebAnno, the name of the layers and their features can in general be freely defined by the annotation project's creator.

Feature type	User	Coordinates				Label
		layer	feature	char offsets	extra	
a) string on span	A	Lemma	value	32-37 (*asked*)		ask
b) string on relation*	A	Dependency	DependencyType	26-27 (*I*), 32-37 (*asked*)		nsubj
c) slot (role-as-label)	A	SemPred	arguments	32-37 (*asked*)	0-4 (*Fred*)	Addressee
d) slot (target-as-label)	A	SemPred	arguments	32-37 (*asked*)	Addressee	0-4 (*Fred*)
e) slot (target-as-label)*	B	SemPred	arguments	32-37 (*asked*)	Speaker	0-4 (*Fred*)

Table 1: Examples of coordinates and labels for different layer and feature types in Figure 1 (except *).

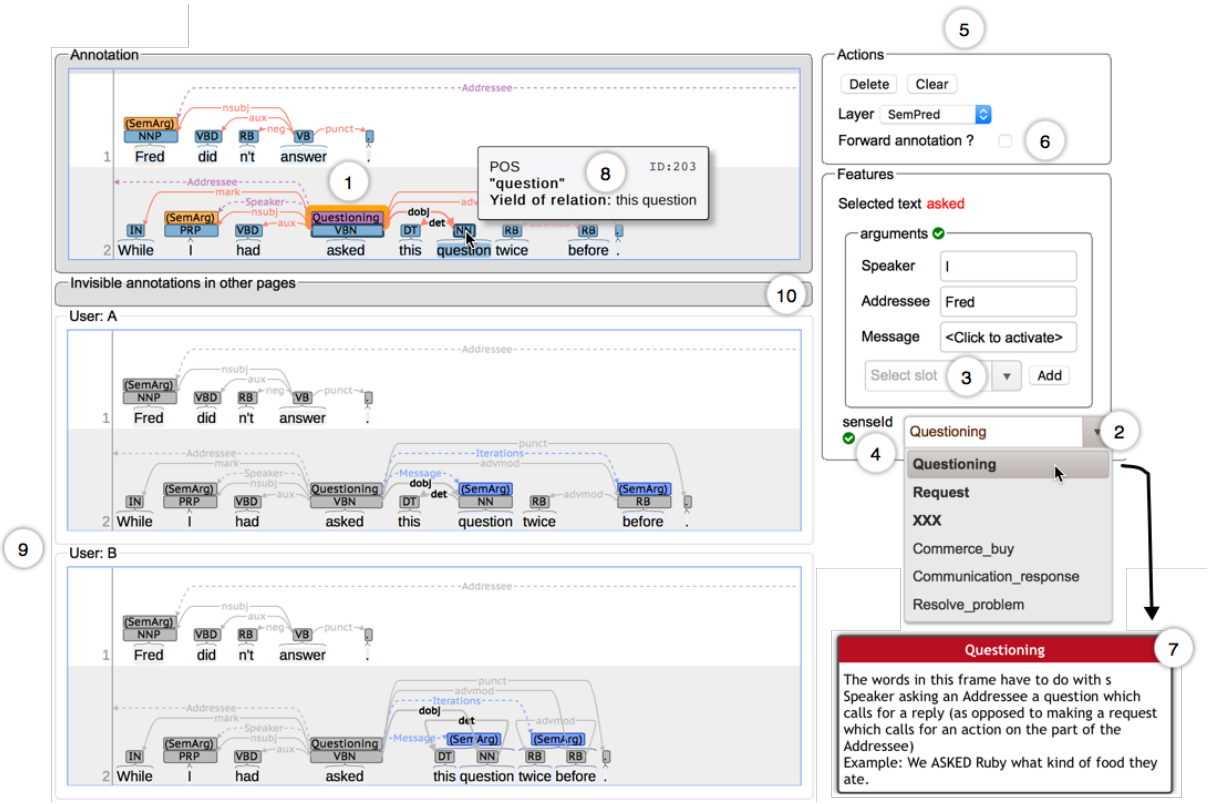

Figure 1: Cross-sentence annotation/curation with slot features and constraint rules.

their label. This makes them two entirely different annotations which the respective other user simply did not annotate rather than a case of disagreement. To cope with such missing annotations, Krippendorff's Alpha (Krippendorff, 1970) was added to the list of supported agreement measures. This measure is conceptually able to handle missing annotations.

The level of detail on agreement calculation provided by WebAnno 2 proved to be insufficient. It is often not clear how the agreement values related to the actual annotations. Thus, while reimplementing the agreement calculation to support for slot features into the agreement mechanism, we also greatly enhanced the level of detail in the UI to explain which data was (not) used for calculation and why it may not have been possible to calculate agreement. Finally, we added the ability to export raw agreement data as CSV files for detailed inspection and external agreement calculation.

3.2 Handling Rich Tagsets via Constraints

Semantic annotation tasks typically operate with a very large set of labels. While a typical part-of-speech tagset contains around 50 tags, FrameNet contains over 800 frame labels. We currently observe over 1,000 role labels and more than 8,000 frame-role combinations in the FrameNet lexicon. However, given a specific lemma, there is only a small set of applicable frame labels that it evokes (e.g. *ask* can evoke the frames *Questioning* and *Request*).

We introduce a generic mechanism that allows formalising such interdependencies as constraint rules.

Listing 1: Example FrameNet-style constraints

```
SemPred {
  // Rule 1
  @Lemma.value = "ask" -> senseId = "Questioning" | senseId = "Request" | senseId = "XXX";
  // .. other lemmata
  // Rule 2
  senseId = "Questioning" ->
    // core roles
    arguments.role = "Addressee" (!) | arguments.role = "Message" (!) | arguments.role = "Speaker" (!) |
    // non-core roles
    arguments.role = "Time" | arguments.role = "Iterations";
  // .. other senses
}

Dependency {
  governor.pos.value = "NN" & dependent.pos.value = "DET" -> dependencyType = "DET";
}
```

Listing 2: Grammar for constraint rules

```
// Basic structure
<file>          ::= <import>* | <scope>*
<scope>         ::= <shortLayerName> "{" <ruleset> "}"
<ruleset>       ::= <rule>*
<import>        ::= "import" <qualifiedLayerName>
                    "as" <shortLayerName>
<rule>          ::= <conds> "->" <actions> ";"
// Conditions
<conds>         ::= <cond> | <cond> "&" <conds>
<cond>          ::= <path> "=" <value>
<path>          ::= <featureName> | <step> "." <path>
<step>          ::= <featureName> | <layerSelector>
<layerSelector> ::= <layerOperator>? <shortLayerName>
<layerOperator> ::= "@" // select annotation in layer X

// Restrictions
<actions>       ::= <action> |
                    <action> "|" <actions>
<action>        ::= <actionPath> "=" <value>
                    ( "(" <flags> ")" )
<actionPath>    ::= <featureName> |
                    <actionPath> "." <featureName>
<flags>         ::= "!" // core role
```

Rules are typically defined over annotations and their labels, but may also access the annotated text. A rule consists of a left-hand side expressing the conditions under which the rule applies and a right-hand side expressing the effect of the rule. A logical conjunction of multiple conditions can be specified. If multiple effects for a rule are specified, they are considered to be alternatives. The mechanism does not limit the annotator's choice. It only reorders the list of labels the users can choose from, showing first those labels covered by constraints in a bold font, then the rest of the labels. While constraint rules can be written manually, they can also be generated automatically from machine-readable schemes. We provide suitable conversions from FrameNet and GermaNet (cf. Section 4).

Listing 1 shows example constraint rules for a FrameNet-like annotation style. These apply to annotations on the layer *SemPred*. Rule 1 reorders the choice of sense IDs for the *SemPred* annotation: if a *Lemma* annotation with value *ask* exists at the same character offsets as the *SemPred* (Figure 1 ①), then the sense IDs *Questioning*, *Request*, and *XXX* (other/unknown) should be offered as top labels in the sense ID dropdown selection (Figure 1 ②). Similar rules for other lemmata would normally follow.

Rule 2 in Listing 1 illustrates constraints on the roles of slot features. If the *senseId Questioning* is chosen, *Addressee*, *Message*, *Speaker*, *Time*, and *Iterations* are offered as top labels in the argument role dropdown box (Figure 1 ③). Slots for core roles (marked with "*!*") are added automatically while slots for other roles need to be added manually.

The constraint mechanism is generic. It uses a path-like notation to navigate along features pointing to other annotations (e.g. *governor.pos.value*), as illustrated in the dependency rule. Within conditions, a path component may also be *@shortLayerName*, meaning "*select all annotations of the specified layer at the same offsets as the current annotation*". We use this in Listing 1 to access information from the *Lemma* layer. Listing 2 shows the full constraint rule grammar in a BNF-like notation.

An indicator (Figure 1 ④) next to a feature in the editor panel shows the status of constraints. The indicator is *green* if there are matching rules (e.g. there is a *Lemma* annotation with the value *ask* and there is a rule with a condition *@Lemma.value="ask"*). It is *orange* if there are rules that affect the feature but none matches (e.g. there is at least one rule with a condition on the value of a *Lemma* but none that matches on the specific given value for the present annotation). Finally, it is *red* if there are rules, but the feature is not restricted by a tagset. In the latter case, the rules have no effect since without a tagset,

there are no labels to be reordered.

3.3 Redesigned Annotation User Interface

The annotation user interfaces of WebAnno 2 and brat use a pop-up dialog whenever an annotation is created or modified. This was perceived as a major impediment to efficient annotation by our annotation team, not only because the dialog obscures the underlying annotation interface, but mainly because the dialog needs to be closed by an explicit user action. Hence, we implemented a dialog-less annotation user interface, similar to the one in the Anafora tool. Along with the new interface, additional user-expressed needs for semantic annotation were addressed. These include better support for keyboard-based annotation, the ability to create zero-length annotations to represent slot fillers not realised in the text, the display of descriptive tooltips explaining semantic tags, the indication of tokens transitively covered by dependency relations, and the ability for the curation of cross-sentence annotations.

Dialog-less annotation. The UI elements used for editing the annotation feature values have been moved from a dialog (cf. brat) to a permanently visible detail panel on the side of the screen (Figure 1 ⑤). Selecting a span of text or drawing an arc now immediately creates an annotation on the active layer and loads its features in the detail panel. Changes to the feature values are immediately reflected in the annotation view. For tasks that require repeatedly annotating the same feature values (e.g. annotating persons in a named entity task), the option *save feature* can be enabled for each feature individually. New annotations are then created with the same feature values as the previously selected annotation.

Keyboard-based annotation. Keyboard-based annotation can significantly speed up the annotation process. For annotation layers with a single tagset-controlled feature (e.g. part-of-speech), we introduce a new keyboard-based *forward annotation* mode (Figure 1 ⑥). In this mode, pressing a key selects the first tag starting with the respective character and pressing it again selects the next. As annotators memorise the number of key presses for the tags, we expect their efficiency to improve. Once a tag is chosen, pressing *space* automatically selects the next token for annotation.

Zero-width annotations. Zero-width span annotations can now be created (at arbitrary positions in the text), e.g. to instantiate semantic predicates or arguments that are not realised in the text.

Tooltips. Tag descriptions are now displayed as tooltips (Figure 1 ⑦) during annotation. In particular for rich semantic tagsets, this is an important improvement. The tag descriptions are part of the tagset definitions which can be edited via the user interface in the project settings. Alternatively, they can be externally generated and imported as JSON files. For the purpose of our annotation project, we automatically derived tagset files from VerbNet and GermaNet.

Relation yield highlighting. Moving the mouse cursor over a span with outgoing relations now displays a tooltip showing the *yield* of the relation (Figure 1 ⑧). E.g. for dependency relations, the yield represents all the tokens subsumed by the respective dependency node. This feature facilitates the coordination of syntactic and semantic annotations when using syntactic heads as slot fillers. In the annotation project, it was determined that dependency-based annotation, enhanced with highlighting of the dependency relation yield proved superior to span-based annotation, which is more error prone.

Cross-sentence curation. We extended the curation component of WebAnno to support the adjudication of multiple annotations for individual sentences. This was necessary in order to curate cross-sentence semantic relations, e.g. to use parts of a previous sentence as slot fillers. An example of such a case is provided in Figure 1 ⑨. Curators can now define the number of sentences to visualise and curate at a time. Additionally, an indicator is displayed for any relation annotations that are not rendered because one of their endpoints is realised outside of the window of currently displayed sentences (Figure 1 ⑩).

4 Experiences and Evaluation

4.1 Annotation Study

Our new tool was tested in a large-scale semantic annotation study on German standard as well as non-standard texts (cf. Mújdricza-Maydt et al. (2016)). An annotation team performed a combination of word sense and semantic role annotation based on GermaNet senses and VerbNet-style role labels. As basis for the annotation, the new slot features were used to model predicate-argument structures. Compared to

Doc	a) WebAnno 2	b) WebAnno 3 no constraints	c) WebAnno 3 constraints
1	14:58	10:07	08:57
2	12:21	10:29	08:16
3	05:19	04:16	03:47
4	10:40	08:22	07:22
All	43:18	33:14	28:22

Table 2: Annotation times in minutes:seconds for WebAnno 2 and 3 with and without constraints.

earlier experiences with predicate-argument annotation using relation layers in WebAnno 2, the annotation team reported improved comfort and efficiency with the new interface. Having the annotation detail panel visible all the time and no longer being forced to constantly switch between annotation dialog and annotation view was perceived as an important factor for this improvement. This study also made extensive use of constraint rules and relied heavily on the new tooltips explaining senses and roles to the annotators. In a pilot study for the project, different strategies of selecting slot fillers were examined. The strategy that was eventually selected relied on pre-annotated dependency relations and required the annotator to select the syntactic head of a phrase to be used as a slot filler (e.g. *house* instead of *the house*). The new ability to display the *yield* of the dependency relations proved to be instrumental in this task to locate the correct head words.

To facilitate SRL annotation projects for others, we provide in addition to the tool itself 1) the setup for our annotation study as a demo project; 2) configuration files for different predicate sense and semantic role labeling frameworks (VerbNet, FrameNet, and PropBank), including their respective tagsets and constraint rules.

4.2 Comparative Evaluation

After the development has been finalised, we conducted a comparative evaluation study in order to measure whether our efforts in tool engineering translate into annotation time speedups, which directly influences annotation cost. For this, we chose the following setup: an annotator was presented with a static display of gold standard predicate-argument annotations like the ones shown in Figure 1. In this way, we minimise fluctuations caused by cognitive load for choosing the right labels, simulating a maximally trained annotator. The annotator had ample experience with both the annotation tools, but not with the specific settings. Thus, for each setting, the annotator became familiar with the setup on a small training document. We conducted this annotation in three different settings: a) using WebAnno 2 (version 2.3.1), modelling the annotation task as outlined in Section 2.1, b) using WebAnno 3 without constraints and c) using WebAnno 3 with constraints for rules that re-order semantic predicates according to the lemma value, as exemplified in (Figure 1 ④). To avoid interpersonal differences in annotation speed, the same annotator worked on all three settings. The data set on which time was measured included four documents with a total of 64 semantic predicates, filling a total of 115 slots with overall 94 semantic arguments (*SemArg* in Figure 1). Note that the same semantic argument can fill slots of different predicates.

Table 2 shows the times in minutes and seconds measured for the four documents. The improvements of the user interface in setting b) already results in a speedup of about 23% less time compared to WebAnno 2. In combination with constraints, a speedup of over 34% is reached in setting c), meaning that the same annotator can produce 50% more annotations in the same time. The annotator reported that she needed to perform considerably fewer interactions in setting c).

5 Conclusion

We have introduced WebAnno 3,[3] the third major release of WebAnno. The new version introduces major improvements that significantly go beyond the state-of-the-art: they enable the annotation of semantic structure and the handling of rich semantic tagsets. The new version was developed in close coordination with an associated annotation project performing combined WSD and SRL annotation on German texts. Moreover, the tool can be used for many non-linguistic annotation tasks in various domains,

[3] https://webanno.github.io/webanno/

e.g. the annotation of metaphors and ambiguities (Digital Humanities), skills and qualifications (Human Resources), sentiment in product reviews (Advertisement/Marketing), etc. A comparative evaluation showed considerable speedups over previous tools and validates both, our adaptation to the user interface and the incorporation of the constraint language to facilitate the annotation of large, interdependent tagsets.

Acknowledgments

The work presented in this paper was funded by a German BMBF grant to the CLARIN-D project, by the German Research Foundation as part of the Research Training Group Adaptive Preparation of Information from Heterogeneous Sources (AIPHES) under grant No. GRK 1994/1, and by the Volkswagen Foundation as part of the Lichtenberg-Professorship Program under grant No. I/82806. We also thank Aakash Sharma and Sarah Holschneider for their active support in the development of WebAnno, the documentation of the tool, and for conducting the comparative evaluation study.

References

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley FrameNet project. In *Proc. of COLING-ACL 1998*, pages 86–90, Montreal, Canada.

Aljoscha Burchardt, Katrin Erk, Anette Frank, Andreas Kowalski, and Sebastian Padó. 2006. SALTO – A Versatile Multi-Level Annotation Tool. In *Proc. of LREC 2006*, pages 517–520, Genoa, Italy.

Aljoscha Burchardt, Katrin Erk, Anette Frank, Andrea Kowalski, Sebastian Padó, and Manfred Pinkal. 2009. Using FrameNet for the Semantic Analysis of German: Annotation, Representation and Automation. In Hans C. Boas, editor, *Multilingual FrameNets – Practice and Applications*, pages 209–244. Mouton de Gruyter, Berlin, Germany.

Wei-Te Chen and Will Styler. 2013. Anafora: A web-based general purpose annotation tool. In *Proc. of the NAACL HLT 2013 Demonstration Session*, pages 14–19, Atlanta, GA, USA.

Jinho Choi, Claire Bonial, and Martha Palmer. 2010. Multilingual PropBank annotation tools: Cornerstone and Jubilee. In *Proc. of the NAACL HLT 2010 Demonstration Session*, pages 13–16, Los Angeles, CA, USA.

Birgit Hamp and Helmut Feldweg. 1997. GermaNet - a lexical-semantic net for German. In *Automatic Information Extraction and Building of Lexical Semantic Resources for NLP Applications*, pages 9–15, Madrid, Spain.

Karin Kipper Schuler. 2005. *VerbNet: A Broad-coverage, Comprehensive Verb Lexicon*. Ph.D. thesis, University of Pennsylvania, Philadelphia, PA, USA.

Klaus Krippendorff. 1970. Estimating the reliability, systematic error and random error of interval data. *Educational. and Psych. Measurement*, 30(1):61–70.

Éva Mújdricza-Maydt, Silvana Hartmann, Iryna Gurevych, and Anette Frank. 2016. Combining Semantic Annotation of Word Sense & Semantic Roles: A Novel Annotation Scheme for VerbNet Roles on German Language Data. In *Proc. of LREC 2016*, pages 3031–3038.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The Proposition Bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–105.

Pontus Stenetorp, Sampo Pyysalo, Goran Topić, Tomoko Ohta, Sophia Ananiadou, and Jun'ichi Tsujii. 2012. brat: a web-based tool for NLP-assisted text annotation. In *Proc. of EACL 2012*, pages 102–107, Avignon, France.

Seid Muhie Yimam, Iryna Gurevych, Richard Eckart de Castilho, and Chris Biemann. 2013. WebAnno: A flexible, web-based and visually supported system for distributed annotations. In *Proc. of ACL 2013: System Demonstrations*, pages 1–6, Sofia, Bulgaria.

Seid Muhie Yimam, Richard Eckart de Castilho, Iryna Gurevych, and Chris Biemann. 2014. Automatic annotation suggestions and custom annotation layers in WebAnno. In *Proc. of ACL 2014: System Demonstrations*, pages 91–96, Baltimore, MD, USA.