DoTAT: A Domain-oriented Text Annotation Tool

Anonymous ACL submission

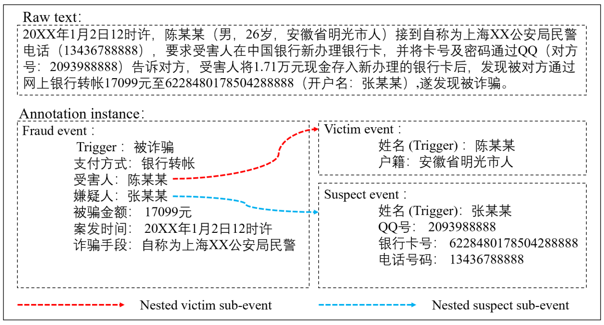
Abstract

We propose DoTAT, a domain-oriented text annotation tool. The tool designs and implements functions heavily in need in domain-oriented information extraction. Firstly, the tool supports a multi-person collaborative process with automatically merging and review, which can greatly improve the annotation accuracy. Secondly, the tool provides annotation of event, nested event and nested entity, which are frequently required in domain-related text structuring tasks. Finally, DoTAT provides visualized annotation specification definition, automatic batch annotation and iterative annotation to improve annotation efficiency. Experiments on the ACE2005 dataset show that DoTAT can reduce the event annotation time by 19.7% compared with existing annotation tools. The accuracy without review is 84.09%, 1.35% higher than Brat and 2.59% higher than Webanno. The accuracy of DoTAT even reaches 93.76% with review. The demonstration video can be accessed from https://ecust-nlp-docker.oss-cn-shanghai.aliyuncs.com/dotat_demo.mp4. A live demo website is available at https://github.com/FXLP/MarkTool.

1 Introduction

A high-quality corpus is a prerequisite in supervised machine learning, especially for neural network-based model. Currently more and more domain-oriented information extraction tasks are proposed, therefore annotation tools should be redesigned to meet the requirements.

- **Multiple specifications support** There are many document types in each domain, and the schema of the target structured data are different. Therefore normally different annotation specification is defined for each document type. For example, in the medical domain, document types many include discharge summary, admission record, examination report, and operation record. In the public security domain, document types include fraud, theft, robbery, disputes and so on.



Raw text:

20XX年1月2日12时许，陈某某（男，26岁，安徽省明光市人）接到自称为上海XX公安局民警电话（13436788888），要求受害人在中国银行新办理银行卡，并将卡号及密码通过QQ（对方号：2093988888）告诉对方，受害人将1.71万元现金存入新办理的银行卡后，发现被对方通过网上银行转帐17099元至6228480178504288888（开户名：张某某）,遂发现被诈骗。

Annotation instance:

Fraud event：
Trigger： 被诈骗
支付方式：银行转帐
受害人：陈某某
嫌疑人：张某某
被骗金额： 17099元
案发时间： 20XX年1月2日12时许
诈骗手段：自称为上海XX公安局民警

Victim event：
姓名 (Trigger)： 陈某某
户籍：安徽省明光市人

Suspect event：
姓名 (Trigger)：张某某
QQ号： 2093988888
银行卡号： 6228480178504288888
电话号码： 13436788888

Figure 1: An example of nested events in the public security domain.

- **Nested event** Domain-oriented information extraction tasks often require event and nested event annotation. As shown in Figure 1, two separate sub-events (“victim” and “suspect”) are nested in the top “fraud” event. Traditionally event is defined as n-tuples and the trigger word is a verb, such as fraud event in Figure 1. In this paper, we take n-tuples of all forms as events, the trigger word can be a noun as a subject, and the arguments may be the attributes of the subject. For example, subject “victim” has multiple attribute-value pairs in Figure 1.

- **Multi-person support with merging and reviewing** Single-person annotation often leads to missing and wrong annotation due to human errors, the ambiguity of the words, or particular language phenomenon not covered by the specifications. In later experiment in Section 5.2, the accuracy may be less than 60% for new annotators. When there are multiple annotation specifications in domain-oriented annotation tasks, more errors may appears since specifications vary and more annotators are required. Therefore, Multi-person collaborative annotation is required to improve the annotation quality. Furthermore the divergence between multiple annotators should be detected and the improved result can be achieved by automatically merging and human reviewing.

Among the existing text annotation tools, only Brat (Stenetorp et al., 2012) and Webanno (Yimam et al., 2013) support event annotation. However, the two do not design event annotation as a core function and do not contain enough features for specification management and quality improvement. To address the challenges above, we propose DoTAT, a domain-oriented text annotation tool for complex event annotation tasks. Besides the ordinary function such as visualized entity and relation annotation, its main features are as follows:

- **Visualized annotation specifications definition** The annotation specifications are defined by a visual interface instead of manual configuration so that administrators can easily define multiple specifications and annotators can dynamically select the specification to match their documents.

- **Merge and review** It provides pairwise consistency checking and automatic merging of content annotated by multiple people. The reviewer can also manually edit the merged content.

- **Iterative annotation** Annotators can re-load previous exported result file for further annotation. The function is frequently used in the situation that new version of a domain specification is designed and existing annotation file should be reused and revised. The above three features forms the basis of DoTAT annotation process and help to improve the quality of the annotation.

- **Nested event and nested entity** The tool not only supports nested event but also supports nested entity. Nested Entity means that one entity is inside another entity.

- **Automatic batch annotation** The tool provides automatic batch annotation by text matching based on regular expressions and dictionaries.

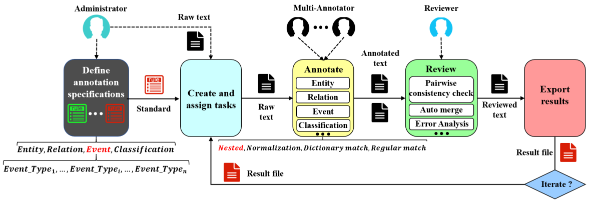


Figure 2: Typical workflow using DoTAT.

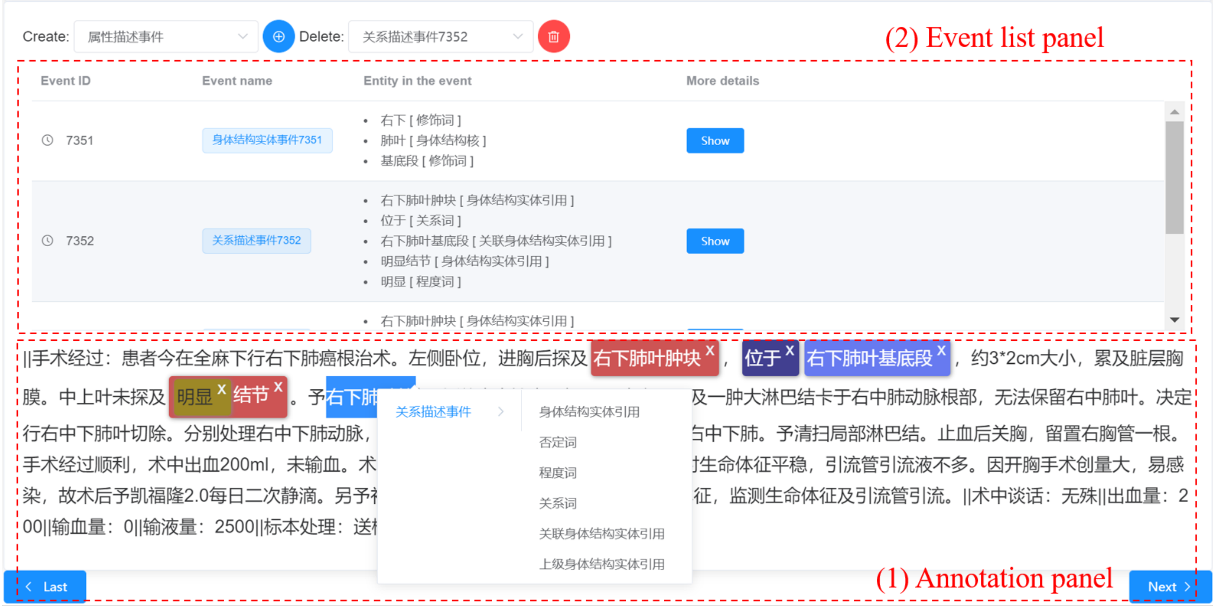
In the following section, we summarize annotation tools. Section 3 describes the overall workflow of DoTAT and its functions. Section 4 introduces the implementation of DoTAT. Section 5 illustrates the comparative experiment. Section 6 shows the case study in the medical and public security domains. Section 7 concludes this paper and gives further directions.

2 Related Work

There are various text annotation tools for different scenarios, but most of them do not support event annotation, including Knowtator (Ogren, 2006), WordFreak (Morton and LaCivita, 2003), Anafora (Chen and Styler, 2013), Atomic (Druskat et al., 2014), GATE Teamware (Bontcheva et al., 2013), Doccano and YEDDA (Yang et al., 2018). Each tool has their own special features, e.g., WordFreak supports constituent parse structure and dependent annotations as well as ACE named-entity and coreference annotation. Doccano and YEDDA support the use of shortcut keys for entity annotation, and YEDDA can perform batch annotation through the command line.

Currently only Brat (Stenetorp et al., 2012) and Webanno (Yimam et al., 2013) support event annotation. However, it is difficult for them to annotate nested event. The method used by Brat and WebAnno for event annotation is to connect multiple entities through directed arcs. If the number of entities is numerous or the distance between entities is far, abundant arcs and intersections will appear on the whole page, resulting in an inferior visualization effect. Except for WordFreak, Anafora and Atomic, most tools declare to support multi-person collaborative annotation. GATE Teamware provides the adjudication interfaces to compare annotations. However, only Webanno provides the curation with automatic merging function.

Compared to these tools, event annotation in



Event ID	Event name	Entity in the event	More details
7351	身体结构实体事件7351	右下 [修饰词] 肺叶 [身体结构核] 基底段 [修饰词]	Show
7352	关系描述事件7352	右下肺叶肿块 [身体结构实体引用] 位于 [关系词] 右下肺叶基底段 [关联身体结构实体引用] 明显结节 [身体结构实体引用] 明显 [程度词]	Show
		右下肺叶肿块 [身体结构实体引用]	

||手术经过：患者今在全麻下行右下肺癌根治术。左侧卧位，进胸后探及右下肺叶肿块，位于右下肺叶基底段，约3*2cm大小，累及脏层胸
膜。中上叶未探及明显结节。予右下肺 及一肿大淋巴结卡于右中肺动脉根部，无法保留右中肺叶。决定
行右中下肺叶切除。分别处理右中下肺动脉， 右中下肺。予清扫局部淋巴结。止血后关胸，留置右胸管一根。
手术经过顺利，术中出血200ml，未输血。术 生命体征平稳，引流管引流液不多。因开胸手术创量大，易感
染，故术后予凯福隆2.0每日二次静滴。另予 征，监测生命体征及引流管引流。||术中谈话：无殊||出血量：2
00||输血量：0||输液量：2500||标本处理：送

Figure 3: The event annotation of DoTAT. Top: event list panel, bottom: annotation panel.

DoTAT is much easier to use. Furthermore DoTAT designs an iterative process from specification definition to merging and review, which can help the annotation team gradually increase the quality of annotated corpus.

3 DoTAT Features Description

DoTAT is a web-based multilingual text annotation tool. There are three types of user roles: administrator, annotator, and reviewer. The fundamental annotation types include entity annotation, relation annotation, event annotation, and text classification. As shown in Figure 2, a typical annotation process using DoTAT may include the following five steps:

- **Define annotation specifications**: The administrator selects the annotation type and visually defines event types, entity types, relation types or text categories in annotation specifications.

- **Create and assign tasks**: Administrator creates and assigns tasks. Each task contains an annotation specification and several raw texts. It is recommended that two annotators and one reviewer are assigned to each task.

- **Annotate**: Before the annotators interactively annotate events or entities, they can use automatic batch annotation to accelerate the speed.

- **Merge and Review**: The reviewer starts consistency checking and automatic merging of the annotated content by multiple annotators. The reviewer can visually analyze the errors according to the merged events list. When there are too many similar errors, the reviewer can give feedback for administrator to redefine the annotation specification. With iterative annotation function, all existing annotations can be reused.

- **Export results**:After the review process, the annotated content can be exported by administrator to a result file and saved in JSON format.

3.1 Event Annotation

The annotation interface of DoTAT contains annotation panel and event list panel, as shown in Figure 3. Users can interactively annotate in the former panel, and the results are summarized in the later one. Users can also select an event in the event list panel and view this event in another panel.

When a user begins annotation, he can use dictionary matching or regular expression matching to automatically annotate entities to reduce manual efforts. For example, in the scenario of Figure 3, we use a medical organs dictionary to automatically annotate "lung" in the text. Then the user begins annotate events. He firstly selects the event type, then uses the mouse to pick a text span in the annotation panel, and then all arguments of this event type will appear immediately, the user can select an argument to annotate. As shown in Figure 3, the annotator selects the argument “身体结构(body structure)” to annotate. The user repeatedly selects each span and corresponding argument to finish the event annotation. For the nested events, when the key of one event becomes an argument of another event, DoTAT considers the former as the internal event of the later one. As shown in Figure 3, the key argument “lung” of the body structure event (7531) is nested in the event (7532) as an argument. For the nested entity annotation, theoretically the internal entity overlaps the outer entity. In order to make both entities displayed well, we make the shadow of the internal entity a little smaller and put it in the top layer, the effects is shown on bottom left of Figure 3.

Algorithm 1 Automatically merge event annotations by using the Kuhn-Munkres Algorithm.

```
Input: $A_n$: the n events of annotator-A; $B_m$: the m events of annotator-B
Output: $C$: the set of merged events; $K$: the consistency checking score
Calculating the similarity matrix $S_{n,m}$ of $A_n$ and $B_m$. Let $s_{i,j}$ denote the element in the i-th row and j-th column of the matrix $S_{n,m}$, and its value represents the similarity between the event $a_i$ of $A_n$ and the event $b_j$ of $B_m$.
Using Kuhn-Munkres Algorithm to find the optimal event merging strategy $W_n$ in matrix $S_{n,m}$
for each event $a_i$ in $A_n$ do
    if $a_i \in W_n$ then
        merge $W_i$
        if the original entities in the two events are the same then the state is ‘Consistent’
        else  the state is ‘Inconsistent’
        end if
        add the merged event to the set $C$
    else
        add $a_i$ to the set $C$ with state ‘Only A’
    end if
end for
for each event $b_j$ in $B_m$ do
    if $b_j \notin W_n$ then
        add $b_j$ to the set $C$ with state ‘Only B’
    end if
end for
$K = \sum s_{i,j}/n$, where $a_i \in W_n$ and $b_j \in W_n$
return $C$,$K$;
```

3.2 Review of Event Annotation

The review procedure supports consistency checking, automatic merging, and manual revision. Before the review, the system will check the consistency of the annotated content of the two annotators. The problem is to find matched events between two annotated text, the detail is shown in Algorithm 1. **1)** We calculate the similarity between each event. The event similarity is calculated as the number of matched entities divided by the number of all entities. The result is recorded as matrix $S_{n,m}$. **2)** Then the problem is defined as the maximum weight matching of weighted bipartite graphs. We apply the Kuhn-Munkres Algorithm to find optimized matching pairs. The consistency checking score is the sum of similarity values of matched pairs divided by the maximum number of events. When consistency checking score reaches the threshold, the system can start the merging process. **3)** The merge criteria depends on the state, and there are four states for each event, “Consistent”, “Only A”, “Only B” and “Inconsistent”. The system automatically merges all the arguments for events in “Inconsistent” state. For the other three states, the system will only keep the larger event.

In the review procedure, the reviewer can view the merged annotations. If the reviewer doubts on the merged event, he can trace the source to view the original annotated event by clicking role switching bar to change current view. The reviewer can also perform manual modification. Typically they should modify the the events in “Inconsistent” state. The whole annotation process finishes after the reviewer submit the refined result.

4 Implementation

DoTAT is a web-based text annotation tool with the software license Apache-2.0. We uses the Vue.js and Element UI to build the user interface. The core of Vue.js is a responsive data binding framework, which makes it pretty easy to synchronize data with the DOM (Document Object Model). Therefore, Vue.js is particularly suitable for real-time visualization of text annotations. The server side utilizes the Python-based open-source Django framework to build RESTful web services. MySQL database is adopted to organize, store and manage data. The code is available at the GitHub repository https://github.com/FXLP/MarkTool, which also contains a live demo website.

Group	Tool	Annotation Time (seconds)					
		20%	40%	60%	80%	100%	$Time_{avg}$
Group-1	WebAnno	1703	3493	5123	6704	8359	418
	Brat	1870	3113	4303	5456	6374	319
	DoTAT	**1340**	**2497**	**3937**	**5007**	**5887**	**295**
Group-2	WebAnno	1518	3138	4589	6055	7516	386
	Brat	1767	3239	4755	6077	7513	375
	DoTAT	**1210**	**2385**	**3845**	**4956**	**5645**	**282**
Group-3	WebAnno	1321	2771	4119	5314	6704	335
	Brat	1503	3055	4218	5293	7174	358
	DoTAT	**1156**	**2167**	**3446**	**4592**	**5387**	**269**

Table 1: Annotation time comparison of annotation tools in ACE2005 Dataset. The average annotation time of annotation tool is arithmetic mean value of $Time_{avg}$ in three group. The average annotation time of Webanno is 380s. The average annotation time of Brat is 351s. The average annotation time of DoTAT is 282s.

5 Experiments

5.1 Annotation time

We compare DoTAT with the other two text annotation tools (Brat and WebAnno) on the event annotation task. We randomly select 20 news texts from the ACE2005 dataset (Consortium, 2005), and each text contains at least four sentences. Six students randomly divided into three groups are invited to annotate those texts. For each user, if a tool is used first, more time may be spent since the user is not familiar with the text. To eliminate the influences, each student is given extra time to view the text before the annotation, and each is assigned a different tool using sequences. We separately record the time (in seconds) spent by each group using the three tools when completing 20%, 40%, 60%, 80%, and 100% of the texts. As we can calculate from Table 1, the average annotation time of DoTAT is reduced by 19.7% compared with Brat and 25.8% compared with WebAnno. DoTAT spends less time, since it is time consuming for Brat and Webanno to connect arcs between the trigger and multiple arguments. The mouse movements in the process may be forward and backward. However, DoTAT only needs to select the arguments from a pop up menu on a text span, and the mouse move is typically from left to right.

5.2 Accuracy

We also evaluate the accuracy by comparing with the golden results from ACE20005 data set. The accuracy is computed as:

$$acc = \frac{\sum_{i=1}^{n}(Trig_i^{correct} + \sum_{j=1}^{m_i} Arg_{i,j}^{correct})}{\sum_{i=1}^{n}(1 + m_i)} \tag{1}$$

where n is the total number of golden events, and m_i is total number of arguments in event i. In event i, $Trig_i^{correct} = 1$ when trigger is correct, and if argument j is correct then $Arg_{i,j}^{correct} = 1$. Since annotation quality is too low in real projects with new annotation specifications or new annotators, we often add a particular training process in real application scenarios. Therefore in this paper we design two rounds of experiments, first round is for training and the second round is formal annotation. After round-1, we have a meeting to discuss with annotators about the error-prone events and entities. In Round-2, we select five other most error-prone texts from ACE 2005. As we can see from Table 2, the average accuracy of unreviewed annotations is less than 60% in experiment Round-1. The main reason is that annotators often missed a whole event or missed particular arguments. For example, when using Brat, the proportion of missing events is 33.67% and The proportion of missing arguments is 14.38%. The accuracy of DoTAT is better since it is less possible for DoTAT to miss arguments. When a text span is picked, DoTAT will show all arguments, the pop menu reminds the annotator about the arguments. DoTAT also performs better than Brat and Webanno in Round-2. Besides, the overall accuracy increase in Round-2, which shows that the training process has effects.

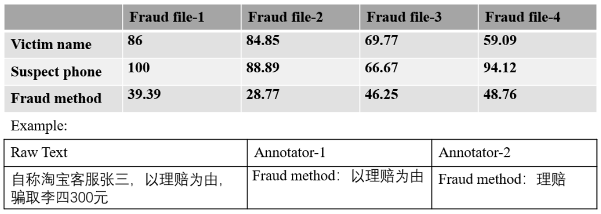
In experiment Round-1, the average accuracy of DoTAT's reviewed annotations reaches 76.2%, which is an increase of 20.9% compared to the aver-

Round	Tool	Accuracy			
		Group-1	Group-2	Group-3	Average
Round-1	WebAnno	44.5%	49.0%	51.7%	48.4%
	Brat	34.5%	44.9%	47.8%	42.4%
	DoTAT-U	45.4%	55.7%	64.8%	55.3%
	DoTAT-R	**67.7%**	**72.6%**	**88.3%**	**76.2%**
Round-2	WebAnno	75.48%	82.58%	86.45%	81.5%
	Brat	79.19%	83.87%	85.16%	82.74%
	DoTAT-U	78.71%	86.45%	87.1%	84.09%
	DoTAT-R	**93.54%**	**92.9%**	**94.84%**	**93.76%**

Table 2: Accuracy comparison of annotation tools in ACE2005 Dataset. DoTAT-U denotes the unreviewed annotation content of DoTAT. DoTAT-R denotes the reviewed annotation content of DoTAT.

Domain	Task	Annotated
Public security	10 types 10,000 texts	6 types 6,000 texts 20,000 events 80,000 entities
Medical	4 types 300 long texts	6,000 events 18,000 entities

Table 3: Application of DoTAT.



	Fraud file-1	Fraud file-2	Fraud file-3	Fraud file-4
Victim name	86	84.85	69.77	59.09
Suspect phone	100	88.89	66.67	94.12
Fraud method	39.39	28.77	46.25	48.76

Example:

Raw Text	Annotator-1	Annotator-2
自称淘宝客服张三，以理赔为由，骗取李四300元	Fraud method：以理赔为由	Fraud method：理赔

Figure 4: The fraud case annotation example.

age accuracy of DoTAT's unreviewed annotations. In experiment Round-2, the average accuracy of DoTAT's reviewed annotations has also increased by 9.67%. It indicates that the review procedure can effectively improve the accuracy. The review procedure not only can complement the missing events and entities, but also reduce the erroneous annotations often caused by the ambiguity of the annotators' understanding of annotation specifications.

6 Case Study

DoTAT has been used in the annotation projects of three different domains. The details in the public security and medical domains are shown in Table 3. For the criminal case type "fraud" which contains 5 event types and altogether 23 arguments in public security domain, the training process before formal annotation involves four original files and eight annotators. Each file contains 20 texts, which are assigned to two annotators. Consistency checking is performed to inspect the specification understanding of each annotator, and part of the results are shown in Figure 4. We found that the argument "fraud method" scored less than 50% in the four files, because the text span of this argument is not fixed. In the example of Figure 4, some annotator annotated "claim settlement(理赔)" and some annotated "on the ground of claim settlement(以理赔为由)". Besides, we also found that some simple arguments (such as "name" and "telephone number") did not reach consistency score of 100%. There are two reasons for this, one is binding an argument to wrong event, e.g. take the "name" of victims as suspects, the other is missing annotation, e.g. "name" of victims appears more than once, but only one place is annotated. Therefore further training is required to solve the disagreement between annotators.

7 Conclusions

The demands for annotation corpus in different domains are rapidly increasing with the development of deep learning. We propose a web-based text annotation tool, DoTAT, which is suitable for domain-oriented complex event annotation. We demonstrate the powerfulness of our tool with experiments and real-world scenarios. We also find training and reviewing are valuable steps to improve the quality of corpus. In the future, we plan to integrate the active learning algorithm into DoTAT to reduce the manual annotation work.

References

Kalina Bontcheva, Hamish Cunningham, Ian Roberts, Angus Roberts, Valentin Tablan, Niraj Aswani, and Genevieve Gorrell. 2013. Gate teamware: a web-based, collaborative text annotation framework. *Language Resources and Evaluation*, 47(4):1007‘Ă‘Ş1029.

Wei-Te Chen and Will Styler. 2013. Anafora: A web-based general purpose annotation tool. In *Proceedings of the 2013 NAACL HLT Demonstration Session*, pages 14–19, Atlanta, Georgia. Association for Computational Linguistics.

L. D. Consortium. 2005. Ace (automatic content extraction) english annotation guidelines for entities.

Stephan Druskat, Lennart Bierkandt, Volker Gast, Christoph Rzymski, and Florian Zipser. 2014. Atomic: an open-source software platform for multi-level corpus annotation.

Thomas Morton and Jeremy LaCivita. 2003. WordFreak: An open tool for linguistic annotation. In *Companion Volume of the Proceedings of HLT-NAACL 2003 - Demonstrations*, pages 17–18.

Philip V. Ogren. 2006. Knowtator: A protégé plug-in for annotated corpus construction. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Demonstrations*, pages 273–275, New York City, USA. Association for Computational Linguistics.

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 *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 102–107, Avignon, France. Association for Computational Linguistics.

Jie Yang, Yue Zhang, Linwei Li, and Xingxuan Li. 2018. YEDDA: A lightweight collaborative text span annotation tool. In *Proceedings of ACL 2018, System Demonstrations*, pages 31–36, Melbourne, Australia. Association for Computational Linguistics.

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 *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 1–6, Sofia, Bulgaria. Association for Computational Linguistics.