# DoTAT: A Domain-oriented Text Annotation Tool

**Yupian Lin[1], Tong Ruan[1,*], Ming Liang[1], Tingting Cai[1], Wen Du[2], Yi Wang[3]**
[1]East China University of Science and Technology, Shanghai 200237, China
[2]DS Information Technology Co., Ltd., Shanghai 200032, China
[3]Fudan University Shanghai Cancer Center,
ruantong@ecust.edu.cn
duwen@dscomm.com.cn, tonywang@shca.org.cn

## 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 events, nested event and nested entity, which are frequently required in domain-related text structuring tasks. Finally, DoTAT provides visual 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 most neural Natural Language Processing (NLP) systems. However, annotation is also one of the most time-consuming and costly components of many NLP research work, and the quality of the annotation results greatly affects the effect of the trained model.

Currently more and more domain-oriented information extraction tasks (Pyysalo et al., 2011, 2012; Miwa and Ananiadou, 2013; Huang et al., 2020) are proposed, therefore annotation tools should be redesigned to meet the new requirements:

**1) Multiple specifications support.** There are many document types in each domain, and the specifications of the target structured data are different. Therefore, different annotation specifications need to be defined for each document type.

**2) Nested event.** (Espinosa et al., 2019; Trieu et al., 2020) An event is called nested event when it has other events in its arguments, while an event is called flat event when there are only entities in its arguments. Domain-oriented information extraction tasks often require event and nested event annotation.

**3) 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. When there are multiple annotation specifications in domain-oriented annotation tasks, more errors may appear 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 automatic merging and human reviewing.

However, the existing annotation tools only support one or two of the above requirements. Only Brat (Stenetorp et al., 2012), Webanno (Yimam et al., 2013; Eckart de Castilho et al., 2016) and INCEpTION (Klie et al., 2018; Boullosa et al., 2018) support event annotation, but they 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. Specifically, it satisfies the above-mentioned new requirements through the following methods which even support iterative annotation and automatic batch annotation:

- **Visual annotation specifications definition** The annotation specifications are defined by



*Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics System Demonstrations*, pages 1 - 8
May 22-27, 2022 ©2022 Association for Computational Linguistics

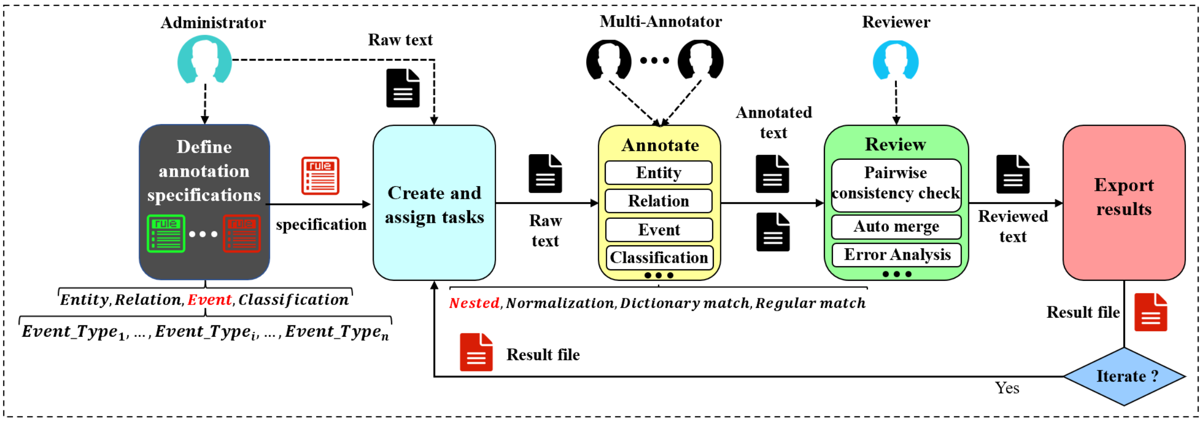


Figure 1: Typical workflow using DoTAT.

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.

- **Nested event and nested entity** The tool not only supports nested event (Figure 2) but also supports nested entity (Figure 4). Nested Entity means that one entity is inside another entity. Besides complex event annotation, DoTAT also supports entity normalization annotation (Figure 5) that is useful when annotating domain-specific corpora, especially in medical domain.

- **Merge and review** It provides pairwise consistency checking and automatic merging of content annotated by pairwise 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.

- **Automatic batch annotation** The tool provides automatic batch annotation by text matching based on regular expressions (Figure 6) and dictionaries (Figure 7).

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), Webanno (Yimam et al., 2013; Eckart de Castilho et al., 2016) and INCEpTION (Klie et al., 2018; Boullosa et al., 2018) support event annotation. However, it is difficult for them to annotate nested event. The method used by them 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

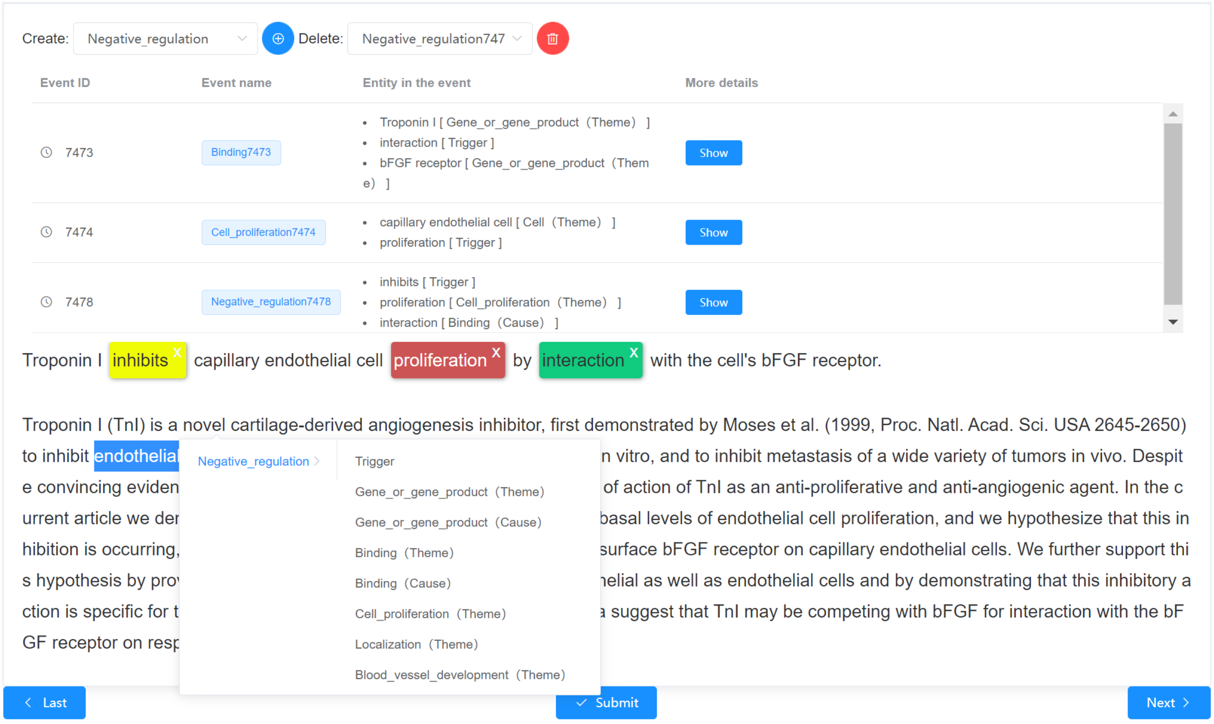


Figure 2: The event annotation of MLEE (Pyysalo et al., 2012). Top: event list panel, bottom: annotation panel.

compare annotations. However, only Webanno and INCEpTION provide the curation with automatic merging function. INCEpTION is partially based on WebAnno (Eckart de Castilho et al., 2016).

Compared to these tools, event annotation in DoTAT is much easier to perform. 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

DoTAT is a web-based multilingual text annotation tool. The raw texts that need to be annotated can be in Chinese or in any other language. 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 1, 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. The detailed annotation process can be seen in section 3.1.

- **Merge and Review**: The reviewer starts consistency checking and automatic merging of the annotated content by multiple annotators (See section 3.2 for details). The reviewer can visually analyze the errors according to the merged events list. When there are 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 (JSON format).

## 3.1 Annotate

The event annotation interface of DoTAT contains annotation panel and event list panel, as shown in Figure 2. Users can interactively annotate in the former panel, and the results are summarized in the latter one. Users can select an event in the event list panel and view this event in another panel.

When beginning annotation, the user first selects the event type. Then he can use dictionary matching or regular expression matching to automatically annotate text span which reduces manual efforts. On this basis, the user manually annotates the trigger or other parameters in the event. Specifically, he uses the mouse to pick a text span in the annotation panel, and then all arguments of this event type will appear immediately, then the user can select an argument to annotate. As shown in Figure 2, the annotator selects the argument "Cell proliferation (Theme)" to annotate the text span "endothelial cell". The user repeatedly selects each span and corresponding argument to finish the event annotation. For the nested events, where the trigger of one event becomes an argument of another event, as shown in Figure 2, the trigger "interaction" of the Binding event (7473) is nested in the negative regulation event (7478) as an argument.

### 3.2 Merge and Review

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 event similarity between pairwise annotators. 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, as shown in Figure 8. If the reviewer doubts on the merged event, he can trace

**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

$S_{n,m} = similarity(A_n, B_m)$, where $S_{i,j} = similarity(a_i, b_j)$, $a_i \in A_n$ and $b_j \in B_m$.
$W_n = Kuhn - Munkres(S_{n,m})$ denote the optimal event merging strategy.
**for** $\forall a_i \in A_n$ **do**
    **if** $a_i \in W_n$ **then**
        $C_i = a_i \cup b_k$, where $b_k = W_i(a_i)$
        **if** $a_i = b_k$ **then**
            $state_{C_i} = Consistent$
        **else** $state_{C_i} = Inconsistent$
        **end if**
    **else** $C_i = a_i$ and $state_{C_i} = OnlyA$
    **end if**
**end for**
**for** $\forall b_j \in B_m$ **do**
    **if** $b_j \notin W_n$ **then**
        $C_{i+j} = b_j$ and $state_{C_{i+j}} = OnlyB$
    **end if**
**end for**
$K = \sum S_{i,j}/n$, where $a_i \in W_n \wedge b_j \in W_n$
**return** $C$,$K$;

the source to view the original annotated event by clicking role switching bar to change current view. The reviewer can also perform manual modification. He should modify the events in "Inconsistent" state. The whole annotation process finishes after the reviewer submits the refined result.

## 4 Implementation

DoTAT is a web-based text annotation tool with the software license Apache-2.0. We used 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

We compared DoTAT with the other two text annotation tools (Brat and WebAnno) for annotation time (see section 5.1) and annotation result (see section 5.2) on the event annotation task.

### 5.1 Annotation time

We randomly selected 20 news texts from the ACE2005 dataset (Consortium, 2005), and each text contained at least four sentences. Six students randomly divided into three groups were invited to annotate those texts. For each user, if a tool was used first, more time might be spent since the user was not familiar with the texts. To eliminate the influences, each student was given extra time to view the text before the annotation, and each was assigned a different tool using sequences. We separately recorded 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 could calculate from Table 1, the average annotation time ($Time_{avg}$) of DoTAT was reduced by 19.7% compared with Brat and 25.8% compared with WebAnno. DoTAT spent less time, since it was time consuming for Brat and Webanno to connect arcs between the trigger and multiple arguments. The mouse movements in the process might be forward and backward. However, DoTAT only needed to select the arguments from a pop up menu on a text span, and the mouse typically moved from left to right.

### 5.2 Annotation result

We also evaluated the accuracy by comparing with the gold standard 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 gold standard 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 $Arg_{i,j}^{correct} = 1$, otherwise the value is 0. Since annotation quality was too low in real projects with new annotation specifications or new annotators, we often added a particular training process in real application scenarios. Therefore, we designed two rounds of experiments, the first round (Round-1) was for training and the second round (Round-2) was a formal annotation. After Round-1, we have a meeting to discuss with annotators about the error-prone events and entities. In Round-2, we selected five other most error-prone texts from ACE 2005. As we could see from Table 2, the average accuracy of unreviewed annotations was less than 60% in experiment Round-1. The main reason was that annotators often missed a whole event or missed particular arguments. The accuracy of DoTAT was better since it was less possible for DoTAT to miss arguments. When a text span was picked, DoTAT would show all arguments, the pop menu reminded the annotator about the arguments. DoTAT also performed better than Brat and Webanno in Round-2. Besides, the overall accuracy increased in Round-2, which showed that the training process had effects.

In Round-1, the average accuracy of DoTAT's reviewed annotations reached 76.2%, which was an increase of 20.9% compared to the average accuracy of DoTAT's unreviewed annotations. In

| 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<br>10,000 texts | 6 types<br>6,000 texts<br>20,000 events<br>80,000 entities |
| Medical | 4 types<br>300 long texts | 4 types<br>300 long texts<br>6,000 events<br>18,000 entities |

Table 3: Application of DoTAT.

Round-2, the average accuracy of DoTAT's reviewed annotations had also increased by 9.67%. It indicated that the review procedure could effectively improve the accuracy.

## 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. Consistency checking is performed to inspect the specification understanding of each annotator, and part of the results are shown in Figure 3. 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. For the example in Figure 3, 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 "phone") did not reach a consistency score of 100%. There are

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

The example annotation of Fraud method:

| 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。 | 自称淘宝客服张三，以理赔为由，骗取李四300元。 | 自称淘宝客服张三，以理赔为由，骗取李四300元。 |

Figure 3: The fraud case annotation example.

two reasons for this: one is binding an argument to the wrong event, e.g. take the "name" of the victim as suspect; the other is missing annotation, e.g. "name" of victim 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 find that the pre-annotation and reviewing are critical 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.


## Acknowledgments

We would like to appreciate the valuable comments and suggestions from the anonymous reviewers. This work was supported by Zhejiang Lab (No.2019ND0AB01). We also would like to thank Hongli Sun, Chuang Chen, and Yuqiu Song for their assistance with annotation experiments.

## 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.

Beto Boullosa, Richard Eckart de Castilho, Naveen Kumar, Jan-Christoph Klie, and Iryna Gurevych. 2018. Integrating knowledge-supported search into the INCEpTION annotation platform. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 127–132, Brussels, Belgium. Association for Computational Linguistics.

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.

Richard Eckart de Castilho, Éva Mújdricza-Maydt, Seid Muhie Yimam, Silvana Hartmann, Iryna Gurevych, Anette Frank, and Chris Biemann. 2016. A web-based tool for the integrated annotation of semantic and syntactic structures. In *Proceedings of the Workshop on Language Technology Resources and Tools for Digital Humanities (LT4DH)*, pages 76–84, Osaka, Japan. The COLING 2016 Organizing Committee.

Kurt Junshean Espinosa, Makoto Miwa, and Sophia Ananiadou. 2019. A search-based neural model for biomedical nested and overlapping event detection. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3679–3686, Hong Kong, China. Association for Computational Linguistics.

Kung-Hsiang Huang, Mu Yang, and Nanyun Peng. 2020. Biomedical event extraction with hierarchical knowledge graphs. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1277–1285, Online. Association for Computational Linguistics.

Jan-Christoph Klie, Michael Bugert, Beto Boullosa, Richard Eckart de Castilho, and Iryna Gurevych. 2018. The INCEpTION platform: Machine-assisted and knowledge-oriented interactive annotation. In *Proceedings of the 27th International Conference on Computational Linguistics: System Demonstrations*, pages 5–9, Santa Fe, New Mexico. Association for Computational Linguistics.

Makoto Miwa and Sophia Ananiadou. 2013. NaCTeM EventMine for BioNLP 2013 CG and PC tasks. In *Proceedings of the BioNLP Shared Task 2013 Workshop*, pages 94–98, Sofia, Bulgaria. Association for Computational Linguistics.

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.

Sampo Pyysalo, Tomoko Ohta, Makoto Miwa, Han-Cheol Cho, Jun'ichi Tsujii, and Sophia Ananiadou. 2012. Event extraction across multiple levels of biological organization. *Bioinformatics*, 28(18):i575–i581.

Sampo Pyysalo, Tomoko Ohta, Rafal Rak, Dan Sullivan, Chunhong Mao, Chunxia Wang, Bruno Sobral, Jun'ichi Tsujii, and Sophia Ananiadou. 2011. Overview of the infectious diseases (ID) task of BioNLP shared task 2011. In *Proceedings of BioNLP Shared Task 2011 Workshop*, pages 26–35, Portland, Oregon, 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.

Hai-Long Trieu, Thy Thy Tran, Khoa N A Duong, Anh Nguyen, Makoto Miwa, and Sophia Ananiadou. 2020. DeepEventMine: end-to-end neural nested event extraction from biomedical texts. *Bioinformatics*, 36(19):4910–4917.

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.

## A Nested entity annotation

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 example is shown in Figure 4.

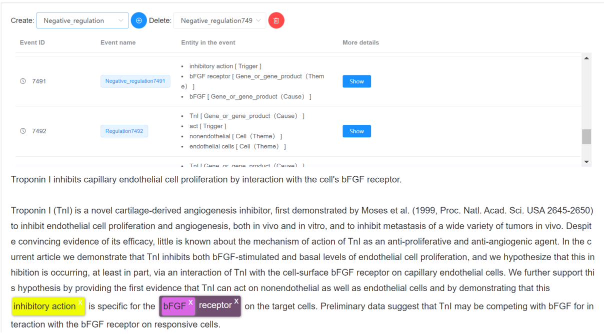

Figure 4: The example of nested entity annotation in DoTAT. The entity "bFGF" is nested in the entity "bFGF receptor".

## B Entity normalization

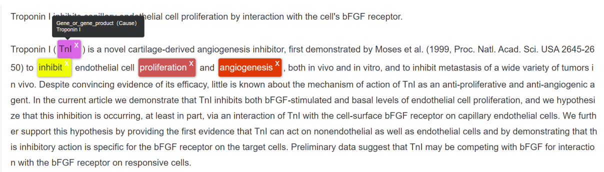

Figure 5: The example of entity normalization in DoTAT. The entity "TnI" has been normalized as "Troponin I".

## C Automatic batch annotation

The example of automatic batch annotation based on regular expressions is shown in Figure 6. Specifically, the user chooses the created regular expression "(angiogenesis|angiogenic){1}" to automatically annotate the trigger of "Blood vessel development" event. And the example of automatic batch annotation based on dictionaries is shown in Figure 7. Specifically, the user chooses the created dictionary to automatically annotate "TnI" as the argument "Gene or gene product(Cause)" of "Negative regulation" event.

## D Review of event annotation

The review interface of event annotation in DoTAT is shown in Figure 8.

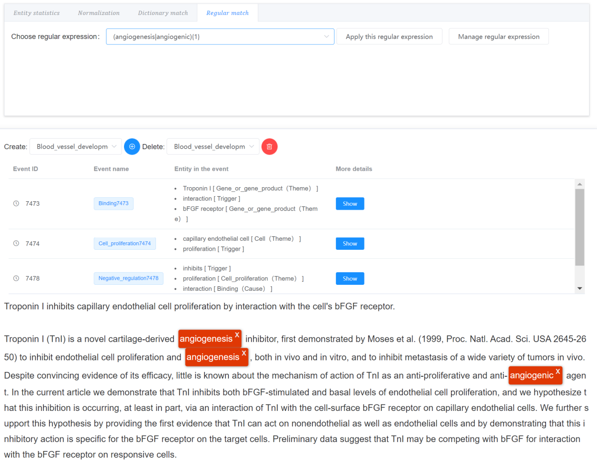

Figure 6: Automatic batch annotation based on regular expressions. Top: regular expression panel, middle: event list panel, bottom: annotation panel.

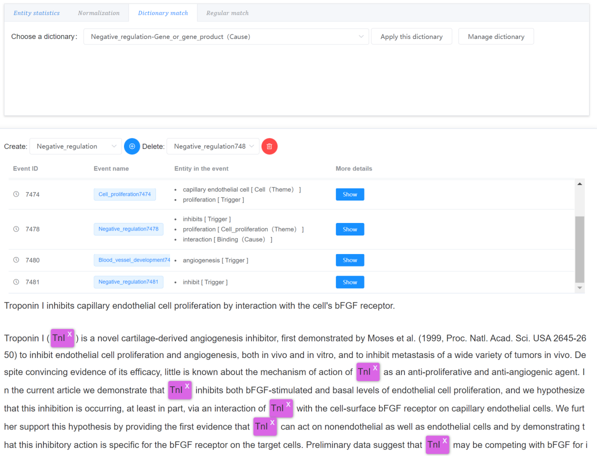

Figure 7: Automatic batch annotation based on dictionaries. Top: dictionary panel, middle: event list panel, bottom: annotation panel.

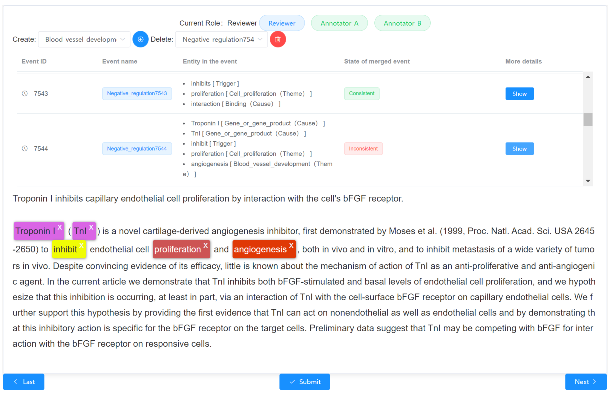

Figure 8: Review of event annotation in DoTAT. Top: role switching bar, middle: event list panel, bottom: annotation panel. Each merged event in the event list has a status and a merged annotation result.