

YEDDA: A Lightweight Collaborative Text Span Annotation Tool

Jie Yang, Yue Zhang

Singapore University of Technology and Design
Somapah Road 8, Singapore
jieynlp@gmail.com, yue_zhang@sutd.edu.sg

Abstract

In this paper, we introduce YEDDA, a lightweight but efficient open-source tool for text span annotation. YEDDA provides a systematic solution for text span annotation, ranging from collaborative user annotation to administrator evaluation and analysis. It overcomes the low efficiency of traditional text annotation tools by annotating entities through both command line and shortcut keys, which are configurable with custom labels. YEDDA also gives intelligent recommendations by training a predictive model using the up-to-date annotated text. An administrator client is developed to evaluate annotation quality of multiple annotators and generate detailed comparison report for each annotator pair. YEDDA is developed based on Tkinter and is compatible with all major operating systems.

Keywords: annotation, span, entity, collaboration, recommendation, evaluation

1. Introduction

Natural Language Processing (NLP) systems rely on large-scale training data for robust accuracies. However, annotation can be time-consuming and expensive. Despite detailed annotation standards and rules, inter-annotator disagreement is inevitable because of human mistakes, language phenomena which are not covered by the annotation rules and the ambiguity of language itself.

Existing annotation tools (Morton and LaCivita, 2003; Cunningham et al., 2002) mainly focus on user annotation process but rarely consider the post-annotation quality analysis, which is necessary due to the inter-annotator disagreement. In addition to the annotation quality, efficiency is also critical in large-scale annotation task, while it is less optimized in many annotation tools (Stenetorp et al., 2012; Ogren, 2006).

To address the challenges above, we propose YEDDA, a lightweight and efficient annotation tool for text span annotation. A snapshot is shown in Figure 2. Here a text span boundary is selected and assigned with a label, which can be useful for Named Entity Recognition (NER) (Tjong Kim Sang and De Meulder, 2003), word segmentation (Sproat and Emerson, 2003), chunking (Tjong Kim Sang and Buchholz, 2000) etc. To keep annotation efficient and accurate, YEDDA provides a systematic solution across the whole annotation process, which includes the shortcut annotation, batch annotation with command line, intelligent recommendation, format exporting and administrator evaluation/analysis.

Figure 1 shows the general framework of YEDDA. It offers annotators with a clear and efficient Graphical User Interface (GUI) to annotate raw text. For the administrator, it provides two useful toolkits to evaluate multi-annotated text and generate detailed comparison report for each annotator pair. The main advantages of our tool are listed:

- **Efficient:** it supports both shortcut and command line model to accelerate the annotation process.
- **Intelligent:** it offers user with real-time system suggestion to reduce duplicated annotation.

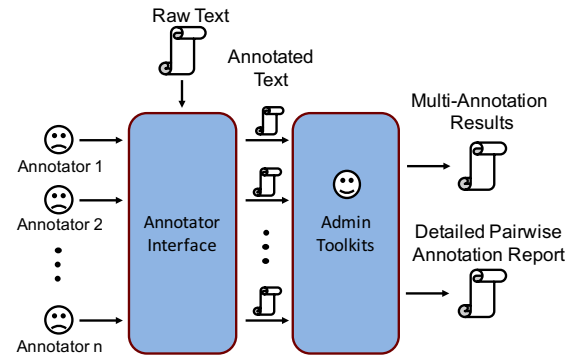


Figure 1: Framework of YEDDA.

- **Comprehensive:** it integrates useful toolkits to give statistical index of multi-user collaborative annotation results and generate detailed content comparison for annotator pairs.

This paper is organized as follows: Section 2 gives an overview of previous text annotation tools and the comparison with ours. Section 3 describes the architecture of YEDDA and its detail functions. Finally, Section 4 concludes this paper and give the future plans.

2. Related Work

There exist many text span annotation tools which focus on different aspects of the annotation process. Stanford manual annotation tool ¹ is a lightweight tool but with limited function. Knowtator (Ogren, 2006) is a general-task annotation tool which links to a biomedical onto ontology to help identify named entities and relations, it supports quality control during the annotation process by integrating inter-annotator evaluation and analysis. Besides, GATE ²

¹<http://nlp.stanford.edu/software/stanford-manual-annotation-tool-2004-05-16.tar.gz>

²GATE is a general NLP tool which includes annotation function.

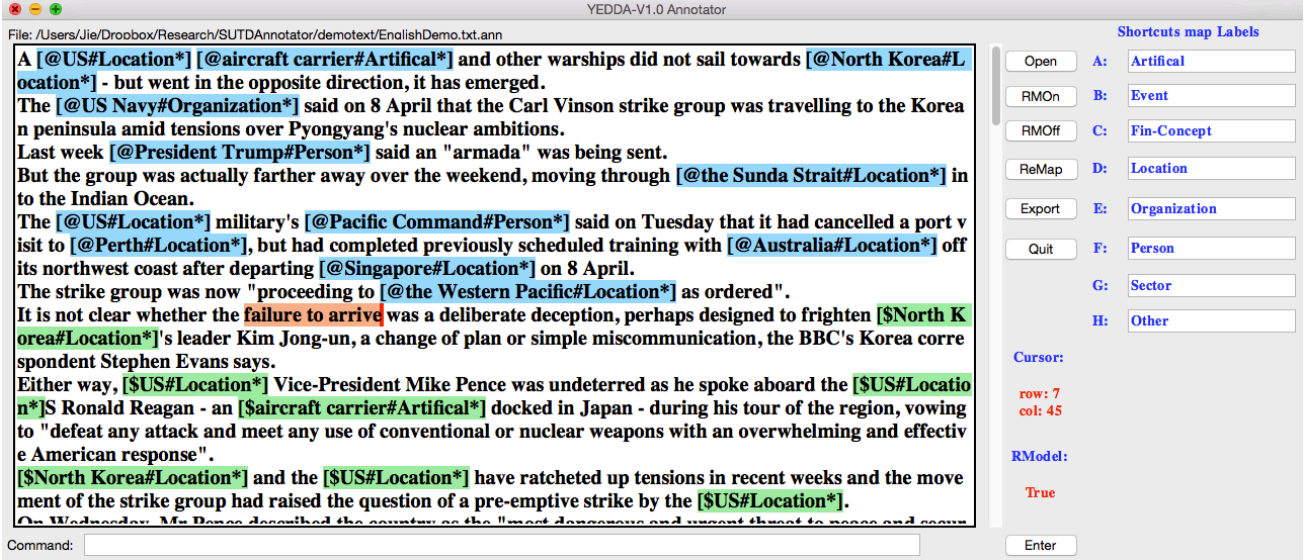


Figure 2: Annotator Interface.

Table 1: Annotation Tool Comparison .

Tool	Operating System			Self Consistency	Command Line Annotation	System Recommendation	Analysis	Size	Language
	MacOS	Linux	Win						
WordFreak	✓	✓	✓	✓	×	✓	×	1.1M	Java
GATE	✓	✓	✓	✓	×	✓	×	544M	Java
Knowtator	✓	✓	✓	×	×	×	✓	1.5M	Java
Stanford	✓	✓	✓	✓	×	×	×	88k	Java
BRAT local	✓	✓	×	×	×	×	✓	31.1M	Python
YEDDA	✓	✓	✓	✓	✓	✓	✓	80k	Python

(Bontcheva et al., 2013) includes a web-based with collaborative annotation framework which allows users to work collaboratively by annotating online with shared text storage. Unlike the tools mentioned above, WordFreak (Morton and LaCivita, 2003) adds a system recommendation function and brings the thought of active learning to rank the unannotated sentences based on the recommend confidence.

BRAT (Stenetorp et al., 2012) is one of the most popular text annotation tools in recent years, it supports both online and local model and provides powerful annotation functions and rich visualization function. However, its online annotation is slow and local annotation requires complex system configuration.

Table 1 concludes the differences of YEDDA with above annotation tools, where “Self Consistency” represents whether the tool works independently or it relies on pre-installed packages. YEDDA provides a lighter but more systematic choice with more flexibility, efficiency and less dependence on system environment for text span annotation. Besides, YEDDA offers administrator useful toolkits for evaluating the annotation quality and analyze the disagreements within annotators.

3. YEDDA

YEDDA provides a simple and clear interface for annotators and useful analysis toolkits for the administrator. It is

developed based on standard Python GUI library Tkinter³, and hence does not need complex environment configuration and is compatible with all Operating System (OS) platforms with Python installation. It offers two user-friendly interfaces for annotators and administrator, respectively, which are introduced in detail in Section 3.1. and Section 3.2., respectively.

3.1. Annotator Client

The client is designed to accelerate the annotation process as much as possible. It supports shortcut annotation to reduce the user operation time. Command line annotation is designed to annotate multi-span in batch. Besides, the client provides system recommendations to lessen the workload of duplicated span annotation.

Figure 2 shows the interface of annotator client on an English entity annotation file. The interface consists of 5 parts. The working area in the up-left which shows the texts with different colors (blue: annotated entities, green: recommended entities and orange: selected text span). The entry at the bottom is the command line which accepts annotation command. There are several control buttons in the middle of the interface, which are used to set annotation model. The status area is below the control buttons, it shows the cursor position and the status of recommending model. The right side shows the shortcut map, which can be configured

³<https://docs.python.org/2/library/tkinter.html>

Algorithm 1: Forward Maximum Matching

Input : $(a, sent)$ **Process:**entity dictionary $d = \{\}$ **for** entity e **in** a **do**| $ADD(e, d)$ **end** $ml = \max_length(d)$ suggested entity $s = []$ $l = LEN(sent)$ $start = 0$ **while** $start < l$ **do**| $end = \min(l, start + ml)$ | **while** $start \leq end$ **do**| | **if** $sent[start:end]$ **in** d **then**| | | $start = end$ | | | $end = -1$ | | | $ADD(sent[start:end], s)$ | | **end**| | **else**| | | $end = end - 1$ | | **end**| **end****end****return** s

easily⁴. Details are introduced as follows.

3.1.1. Shortcut Annotation

YEDDA provides the function of annotating text span by selecting using mouse and press shortcut to map the selection into a specific label. It is a common annotation process in many annotation tools (Stenetorp et al., 2012; Bontcheva et al., 2013). It binds each label with one custom shortcut key, this is shown in the “Shortcuts map Labels” part of Figure 2. The annotator needs only two steps to annotate one text span, i.e. “select and press”. The annotated file updates simultaneously with each key pressing process.

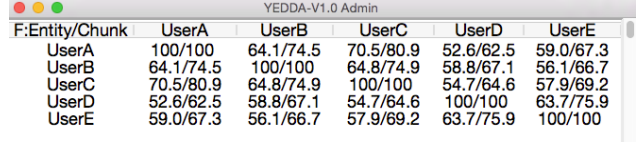
3.1.2. Command Line Annotation

YEDDA also support the command line annotation function (see the command entry in the bottom of Figure 2) which can execute multi-span annotation at once. The system will parse the command automatically and convert the command into multi-span annotation instructions and execute in batch. It is quite efficient for character-based languages, such as Chinese and Japanese. The command follows a simple rule which is “ $n1 + key1 + n2 + key2 + n3 + key3 + \dots$ ”, where ‘ $n1, n2, n3$ ’ are the length of the entities and ‘ $key1, key2, key3$ ’ is the corresponding shortcut key. For example, command “ $2a3d2b$ ” represents annotating following 2 characters as label ‘ a ’ (mapped into a specific label name), the following 3 characters as label ‘ d ’ and 2 characters further as label ‘ b ’.

3.1.3. System Recommendation

It has been proven that using pre-annotated text and manual correction increases the annotation efficiency in many

⁴Type the corresponding labels into the entries following shortcut keys and press “ReMap” button.



F:Entity/Chunk	UserA	UserB	UserC	UserD	UserE
UserA	100/100	64.1/74.5	70.5/80.9	52.6/62.5	59.0/67.3
UserB	64.1/74.5	100/100	64.8/74.9	58.8/67.1	56.1/66.7
UserC	70.5/80.9	64.8/74.9	100/100	54.7/64.6	57.9/69.2
UserD	52.6/62.5	58.8/67.1	54.7/64.6	100/100	63.7/75.9
UserE	59.0/67.3	56.1/66.7	57.9/69.2	63.7/75.9	100/100

Figure 3: Multiple annotators F-value matrix.

annotation tasks (Meurs et al., 2011). YEDDA offers annotators with system recommendation and correction function. Algorithm 1 shows the forward maximum matching algorithm used in our recommending system on an entity annotation example. It parses the annotated text part a and adds the annotated entities into a dictionary d . For the unannotated sentence $sent$, recommending system scans the sentence from the left to right with a span with initial length ml , the entity will be added into an entity list s if the entity can be found in d , otherwise the last character of the span will be stripped and rematch. The final predicted s is returned and colored in the annotation interface, as the green part shown in Figure 2. Annotators can use shortcut to confirm, correct or veto the suggestions. The recommending system keeps online updating during the whole annotation process, which learns the up-to-date annotation information. The recommending system is designed as “pluggable” which ensures that the recommending algorithm can be easily extended into other sequence labeling models, such as Conditional Random Field (CRF)⁵ (Lafferty et al., 2001). The recommendation can be controlled through two buttons “RMON” and “RMOFF”, which enables and disables the recommending function, respectively.

3.1.4. Annotation Modification

It is inevitable that the annotator or the recommending system gives incorrect annotations or suggestions. Based on our annotation experience, we found that the time cost of annotation correction can not be neglected. Therefore, YEDDA provides several efficient modification actions to revise the annotation:

- **Action Withdraw:** annotators can cancel their previous action and let system return to the last status by press the shortcut key $Ctrl+z$.
- **Span Label Modification:** if the selected span was in right boundary but got an incorrect label, annotator only needs to put the cursor inside the span (or select the span) and press the shortcut key of the right label to correct label.
- **Label Deletion:** when the plain text was incorrectly annotated (recommended) with a label by annotator (system), similar to the label modification, the annotator can put the cursor inside the span and press shortcut key q to remove the annotated (recommended) label.

⁵Those sequence labeling models work well on big training data. For limited training data, the maximum matching algorithm gives better performance.

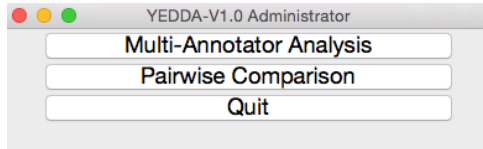


Figure 4: Administrator Interface.

3.1.5. Export Annotated Text

As the annotated file is saved in `.ann` format, YEDDA provides the “Export” function which exports the annotated text as standard format (ended with `.anns`). Each line includes one word/character and its label, sentences are separated by an empty line. The exported label can be chosen in either BIO or BIOES format (Ratinov and Roth, 2009).

3.2. Administrator Toolkits

For the administrator, it is important and necessary to evaluate the quality of annotated files and analyze the detailed disagreements of different annotators. Shown as Figure 4, YEDDA provides a simple interface with several toolkits for administrator monitoring the annotation process.

3.2.1. Multi-Annotator Analysis

To evaluate and monitor the annotation quality of different annotators, our Multi-Annotator Analysis (MAA) toolkit imports all the annotated files and gives the analysis results in a matrix. As shown in Figure 3, the matrix gives the F-values in full level (consider both boundary and label accuracy) and boundary level (ignore the label correctness, only care about the boundary accuracy) of all annotator pairs.

3.2.2. Pairwise Annotators Comparison

If an administrator wants to look into the detailed disagreement of annotators, it is quite convenient by using the Pairwise Annotators Comparison (PAC). PAC loads two annotated files and generates a specific comparison report file⁶ for the two annotators. As shown in Figure 5, the report is mainly in two parts:

- **Overall Statistics:** it shows the specific precision, recall and F-measure value⁷ of two files in all labels. It also gives the three accuracy indexes on overall full level and boundary level in the end.
- **Content Comparison:** this function gives the detailed comparison of two annotated files in whole content. It highlights the annotated parts of two annotators and assigns different color for the agreed and disagreed span.

4. Conclusion and Future Work

We have presented a lightweight but systematic annotation tool, YEDDA, for annotating the entities in text and analyzing the annotation results efficiently. The source

⁶The report is generated in `.tex` format and can be compiled into `.pdf` file.

⁷Notice that we assume “File1” as a gold standard, this only affects the order of precision and recall, while the F-value keeps same if we choose the other file as gold standard.

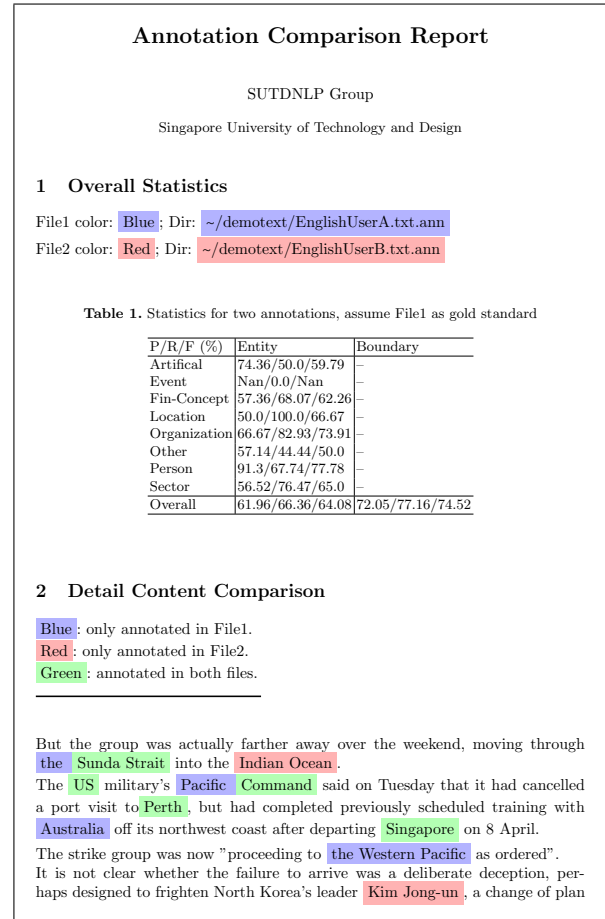


Figure 5: Detailed report for annotator pair

codes of this tool are released at <https://github.com/jiesutd/SUTDAnnotator>. In order to reduce the workload of annotators, we are going to integrate active learning strategy in our system recommendation part in the future. A supervised sequence labeling model (such as CRF) is trained based on the annotated text, then unannotated sentences with less confidence (predicted by this model) are reordered in the front to ensure annotators only annotate the most confusing sentences.

5. Acknowledgements

We thank Yanxia Qin, Hongmin Wang, Shaolei Wang, Jiangming Liu, Yuze Gao, Ye Yuan, Lu Cao, Yumin Zhou and other members of SUTDNLP group for their trials and feedbacks. Yue Zhang is the corresponding author.

6. Bibliographical References

- Bontcheva, K., Cunningham, H., Roberts, I., Roberts, A., Tablan, V., Aswani, N., and Gorrell, G. (2013). Gate teamware: a web-based, collaborative text annotation framework. *Language Resources and Evaluation*, 47(4):1007–1029.
- Cunningham, H., Maynard, D., Bontcheva, K., and Tablan, V. (2002). Gate: an architecture for development of robust hlt applications. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 168–175. Association for Computational Linguistics.

- Lafferty, J., McCallum, A., and Pereira, F. C. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning*, volume 951, pages 282–289.
- Meurs, M.-J., Murphy, C., Naderi, N., Morgenstern, I., Cantu, C., Semarjit, S., Butler, G., Powlowski, J., Tsang, A., and Witte, R. (2011). Towards evaluating the impact of semantic support for curating the fungus scientific literature. *WS2*, page 34.
- Morton, T. and LaCivita, J. (2003). Wordfreak: an open tool for linguistic annotation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology: Demonstrations-Volume 4*, pages 17–18. Association for Computational Linguistics.
- Ogren, P. V. (2006). Knowtator: a protégé plug-in for annotated corpus construction. In *Proceedings of the 2006 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology: companion volume: demonstrations*, pages 273–275. Association for Computational Linguistics.
- Ratinov, L. and Roth, D. (2009). Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, pages 147–155. Association for Computational Linguistics.
- Sproat, R. and Emerson, T. (2003). The first international chinese word segmentation bakeoff. In *Proceedings of the second SIGHAN workshop on Chinese language processing-Volume 17*, pages 133–143. Association for Computational Linguistics.
- Stenetorp, P., Pyysalo, S., Topić, G., Ohta, T., Ananiadou, S., and Tsujii, J. (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. Association for Computational Linguistics.
- Tjong Kim Sang, E. F. and Buchholz, S. (2000). Introduction to the conll-2000 shared task: Chunking. In *Proceedings of the 2nd workshop on Learning language in logic and the 4th conference on Computational natural language learning-Volume 7*, pages 127–132. Association for Computational Linguistics.
- Tjong Kim Sang, E. F. and De Meulder, F. (2003). Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*, pages 142–147. Association for Computational Linguistics.