Erase and Rewind: Manual Correction of NLP Output through a Web Interface

Valentino Frasnelli, Lorenzo Bocchi
Dept. of Psychology and Cognitive Science
University of Trento
Rovereto (Trento), Italy
[name.surname]@studenti.unitn.it

Alessio Palmero Aprosio
Digital Humanities Unit
Fondazione Bruno Kessler
Trento, Italy
aprosio@fbk.eu

Abstract

In this paper, we present Tintful, an NLP annotation software that can be used both to manually annotate texts and to fix mistakes in NLP pipelines, such as Stanford CoreNLP. Using a paradigm similar to wiki-like systems, a user who notices some wrong annotation can easily fix it and submit the resulting (and right) entry back to the tool developers. Moreover, Tintful can be used to easily annotate data from scratch. The input documents do not need to be in a particular format: starting from the plain text, the sentences are first annotated with CoreNLP, then the user can edit the annotations and submit everything back through a user-friendly interface.

A video showing Tintful and its feature is available on YouTube.[1]

1 Introduction

In the last years, NLP tools are being more and more used in tasks such as textual inference, machine translation, hate speech detection (Socher et al., 2012). Most of these tasks rely on machine learning systems trained on large amounts of data, which have been manually labeled by annotators, often domain experts. In particular, recent deep learning algorithms are more accurate, but they need more data for training, making the data collection a major challenge for the NLP community.

When the annotation task does not require a specialised competence, one can use a platform such as Amazon Mechanical Turk (AMT),[2] that enables the distribution of low-skill but difficult-to-automate tasks to a network of humans, who could work in parallel, when and where they prefer.

However, not all NLP assignments can be solved by non-experts because they may require background knowledge or linguistic expertise. For these tasks, expert annotators should be hired and receive a specific training, which is time-consuming and can become costly.

A similar problem arises when a known task has to be ported to another domain, and existing tools turn out to have a poor accuracy, because the original training data does not include annotated instances from that domain (Ben-David et al., 2007). For example, the performance of standard NLP tools (part-of-speech taggers, dependency parsers, and so on) is severely degraded on tweets for this very reason (Ritter et al., 2011). One of the solutions to this set of problems would be to make NLP tools more similar to wiki-like systems, where a user who notices some wrong annotation can easily fix it and submit the resulting (and right) entry back to the tool developers, so that they can add the instance to the training examples and re-generate the model.

This is basically the paradigm already used for active learning (Settles, 2011) which uses "humans in the loop" to increase the accuracy of a system, by including in the workflow a targeted correction of instances that are misclassified (Fan et al., 2017). As someone said way back in 1969: "Computers are incredibly fast, accurate and stupid. On the other hand, a well trained operator as compared with a computer is incredibly slow, inaccurate and brilliant". (Various Authors, 1969)

To obtain a seamless integration between automatic classification and human correction, we need to develop NLP tools that are accessible through a user-friendly interface (Holzinger, 2013), easing the interaction between non-technical persons and the underlying technology.

In this paper, we present Tintful, a working example of the described paradigm, an interface that combines the output of Stanford CoreNLP (Manning et al., 2014) with a newly created annotation tool that allows the user to edit and fix the output

[1] https://youtu.be/iFDCbtfWdTg
[2] http://www.mturk.com/

data, in a wiki-like style. The user (registered or not) can edit the tokens, the lemmas, the parts of speech, the dependency trees and the named entities (persons, locations, organizations). Thanks to an API released with the web interface, the resulting annotation can be stored for later use (for example, the software retraining), so that users can contribute to improving the system performance on specific tasks or domains of interest.

Compared to other similar tools (see Section 2), Tintful has some main strengths:

- There is no need to pre-process the data one wants to annotate. The user can easily enter the raw text into the system and directly edit the output annotation.
- There is no need for an expert to setup the environment. Just install the tool and start to annotate.
- "Casual" users can contribute to the annotation by submitting their anonymous annotations to the system.
- The annotated data is immediately available just by querying the database, and can be used in an incremental learning framework (Schlimmer and Fisher, 1986).[3]

We believe that this paradigm may foster the adoption of NLP tools in domains and settings that so far have not taken full advantage of text processing. For example, social scientists or humanities scholars would have the possibility to correct the output of a parser or NER trained on news, which may perform poorly on other types of texts, directly through the tool interface, making it easier to adapt the model to new domains and genres.

Finally, the whole tool is released open source and available on Github (see Section 7).

2 Related work

Some of the available programs for the manual annotation of texts are generic and can be configured and used for a great variety of tasks. These are usually powerful but need some work for configuration. Some other, on the contrary, have been developed for a particular purpose, and usually are easier to launch and configure.

WebAnno[4] (Eckart de Castilho et al., 2016) is a general purpose web-based annotation tool for a wide range of linguistic annotations and belongs to the former category. It is multi-user and supports different roles to guarantee a quality check of the annotated data.

INCEpTION[5] (Klie et al., 2018) is an open-source and multi-user text annotation platform developed at the Technische Universität Darmstadt. It is general-purpose and can be configured to perform a number of annotation tasks.

Similarly, Doccano[6] (Nakayama et al., 2018) provides annotation features for text classification, sequence labeling and sequence to sequence tasks.

Regarding specifically the annotation of dependency graphs, there several tools that help researchers to manage complex output formats such as CoNLL-U.[7] For instance, ConlluEditor[8] (Heinecke, 2019) is an actively maintained tool which facilitates the editing of syntactic relations and morphological features of files in CoNLL-U format. Similarly, UD-Annotatrix[9] (Tyers et al., 2018) is a language-independent tool for editing dependency trees according to the guidelines established by the Universal Dependencies project. TrUDucer[10] (Hennig and Köhn, 2017) is a software for transforming dependency treebanks from one schema to another, especially to CoNLL-U.

Finally, Arborator[11] is an annotation tool for dependency trees that allows users to perform collaborative work. The tool has a specific focus on HCI aspects, since most of the actions can be done using mouse drag and drop.

In this paper, we propose a tool that is different from all the ones described above. With Tintful the user does not need the data to be in a particular format: starting from the plain text, the sentences are first annotated with Stanford CoreNLP, then the user can edit the annotations and submit everything back to the server.

[3]This part is not included in Tintful out-of-the-box, yet. For now, it can be done by external tool by a machine learning expert. We plan to add a feature to make it easy also for non-expert users in the future, see Section 8.

[4]`https://webanno.github.io/`
[5]`https://inception-project.github.io/`
[6]`https://github.com/doccano/doccano`
[7]`https://universaldependencies.org/format.html`
[8]`https://github.com/Orange-OpenSource/conllueditor`
[9]`https://github.com/jonorthwash/ud-annotatrix`
[10]`http://nats.gitlab.io/truducer/`
[11]`https://arborator.ilpga.fr/`

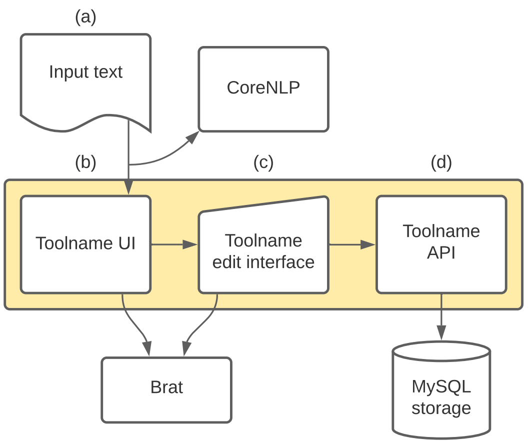


Figure 1: A graph representation of the Tintful architecture.

3 Existing components

Tintful is mainly based on two existing pieces of software: Stanford CoreNLP and Brat.

3.1 Stanford CoreNLP

Stanford CoreNLP (Manning et al., 2014) is an open-source framework written in Java that provides most of the common Natural Language Processing tasks out-of-the-box for several languages. The framework provides also an easy interface to extend the annotation to new tasks and/or languages.

Tintful is agnostic w.r.t. the language. One can use the plain Stanford CoreNLP software with Tintful if they need to annotate texts in English, German, Spanish, and so on. We processed Italian texts and therefore used TINT (Palmero Aprosio and Moretti, 2018), a language-specific extension whose output is compatible with CoreNLP.

3.2 BRAT

The BRAT Rapid Annotation Tool[12] (Stenetorp et al., 2012) is a web-based tool for text annotation and is developed at University of Manchester. Although its last version is quite old (November 2012), a large number of research organization still uses it, since it is very intuitive and can be used also for visualization-only (the official demo pages of Stanford Stanza[13] (Qi et al., 2020) and CoreNLP both use it). In Tintful, Brat is used for the graphical annotation of the syntactic dependencies. The main issue of this library is that it is not responsive, meaning that it is not optimized for mobile phones and tablets. Therefore, we slightly modified it to make Tintful usable on most devices.

[12]http://brat.nlplab.org/
[13]http://stanza.run/

4 Tintful architecture

The architecture of Tintful is represented in Figure 1.

1. First, the user inserts a text (a) in the Tintful interface (Figure 6).

2. The NLP pipeline (Stanford CoreNLP or compatible variants) is then launched, using the text as input.

3. The resulting JSON is parsed by Tintful and shown in the UI (b). In this screen, the user can browse through all the annotation layers (tokens, lemma, POS, dependency parsing, NER, and so on). See Figure 7. Additional modules not included in CoreNLP are shown, if the corresponding annotation is present in the JSON file. Among these modules, the Tint readability module (Palmero Aprosio and Moretti, 2018), that estimates the difficulty level of the document and calculates some indexes, such as lexical density, semantic richness, Flesch score (Gulpease for Italian), and so on.

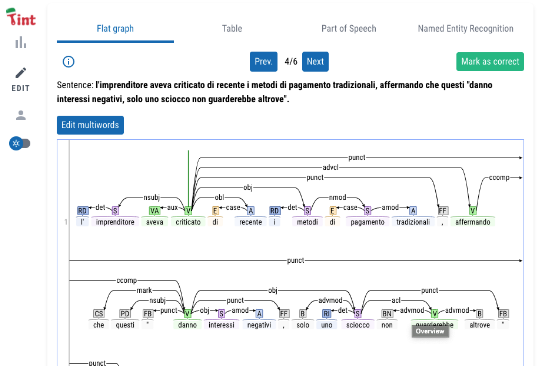
4. If the annotation contains one or more mistakes, the user can enter the edit mode (c) to fix them; otherwise, the annotation can be saved as is (go to step 6). See, for example, Figure 2.

5. The edit screen shows the parsing results to the user sentence by sentence. One can edit every aspect of the annotation: tokens, lemma, POS, dependency labels, dependency tree hierarchy, NER).

6. Once the editing is finished, the user can save the resulting annotation, that is stored in the server (d).

7. If the user is logged in when submitting the edited data, they can recover the annotation and edit/delete it (Figure 8).

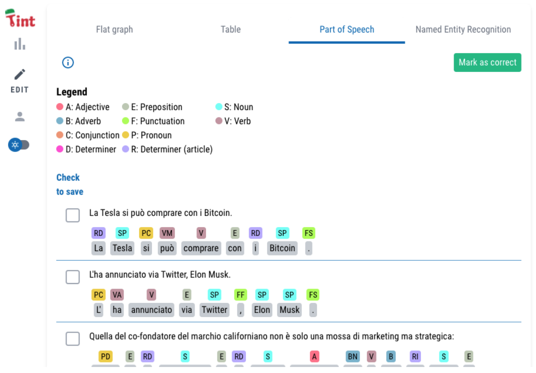
8. Finally, the annotated data can be exported and downloaded in CoNLL-U format.


Tint
EDIT
Flat graph
Table
Part of Speech
Named Entity Recognition
Prev.
4/6
Next
Mark as correct
Edit multiwords
Overview

Sentence: **l'imprenditore aveva criticato di recente i metodi di pagamento tradizionali, affermando che questi "danno interessi negativi, solo uno sciocco non guarderebbe altrove".**

RD det S nsubj VA aux V obl E case A obj RD det S nmod E case S amod A FF V punct advcl punct ccomp
l' imprenditore aveva criticato di recente i metodi di pagamento tradizionali , affermando
CS PD FB punct V obj S amod A FF B RI det S BN advmod V acl advmod B FB ccomp mark nsubj punct obj advmod punct
che questi " danno interessi negativi , solo uno sciocco non guarderebbe altrove "

Figure 2: The syntactic dependency editing interface using Brat.



Legend
- ● A: Adjective ● E: Preposition ● S: Noun
- ● B: Adverb ● F: Punctuation ● V: Verb
- ● C: Conjunction ● P: Pronoun
- ● D: Determiner ● R: Determiner (article)

☐ La Tesla si può comprare con i Bitcoin.

RD	SP	PC	VM	V	E	RD	SP	FS
La	Tesla	si	può	comprare	con	i	Bitcoin	.

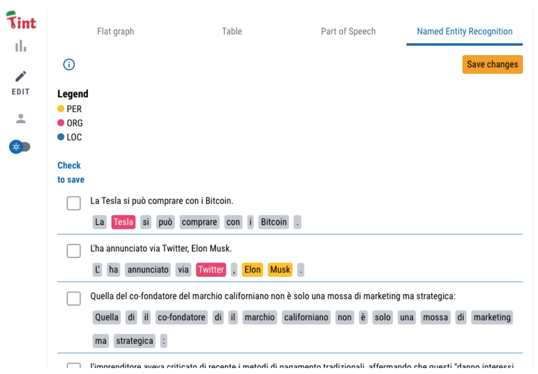
☐ L'ha annunciato via Twitter, Elon Musk.

PC	VA	V	E	SP	FF	SP	SP	FS
L'	ha	annunciato	via	Twitter	,	Elon	Musk	.

☐ Quella del co-fondatore del marchio californiano non è solo una mossa di marketing ma strategica:

PD E RD S E RD S A BN V B RI S E

Figure 3: The editing interface for POS.



Legend
- ● PER
- ● ORG
- ● LOC

☐ La Tesla si può comprare con i Bitcoin.

La Tesla si può comprare con i Bitcoin .

☐ L'ha annunciato via Twitter, Elon Musk.

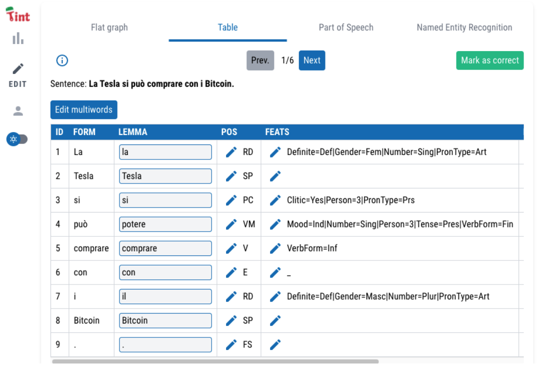
L' ha annunciato via Twitter , Elon Musk .

☐ Quella del co-fondatore del marchio californiano non è solo una mossa di marketing ma strategica:

Quella di il co-fondatore di il marchio californiano non è solo una mossa di marketing ma strategica :

☐ l'imprenditore aveva criticato di recente i metodi di pagamento tradizionali, affermando che questi "danno interessi

Figure 4: Interface for editing named entities.



Sentence: **La Tesla si può comprare con i Bitcoin.**

ID	FORM	LEMMA	POS	FEATS
1	La	la	RD	Definite=Def|Gender=Fem|Number=Sing|PronType=Art
2	Tesla	Tesla	SP	
3	si	si	PC	Clitic=Yes|Person=3|PronType=Prs
4	può	potere	VM	Mood=Ind|Number=Sing|Person=3|Tense=Pres|VerbForm=Fin
5	comprare	comprare	V	VerbForm=Inf
6	con	con	E	_
7	i	il	RD	Definite=Def|Gender=Masc|Number=Plur|PronType=Art
8	Bitcoin	Bitcoin	SP	
9	.	.	FS	

Figure 5: Editing the text information using a tabular view.

Figures 2, 3, 4, and 5 show some screenshots of the annotation interface. Manual annotations can be performed both by occasional and registered users, so that the administrator knows which data belongs to whom.

5 Additional features

5.1 Modular structure

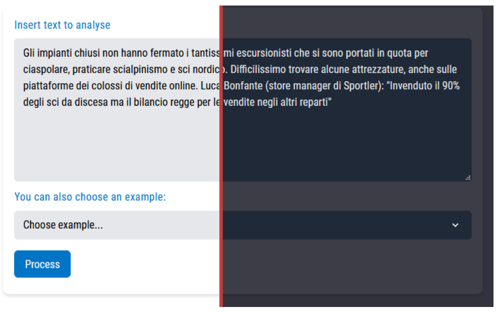
When running Tintful, one can edit everything that normally is included in the CoNLL-U format (see Section 2): token, lemma, POS, morphological features, syntactic dependency hierarchy and labels. There is also room for the miscellaneous data (last field of the CoNLL-U file) and the named entities, that are not included in the format but can be added with our interface. It can happen that the casual user only edits some parts of the text (for example, the POS or the NER, without even touching the syntactic tree). When saving the data, Tintful select only the parts where the user did some edits. If a sentence is already correct and therefore is not edited by the user, no information is sent to the server. The user can manually force the sending, by clicking the ”already correct” button (see, for instance, Figures 2, 3, and 5.

5.2 User management

The administrator can create users (with login and password), so that the data sent by that user can be identified. In addition, the registered user can retrieve the annotated sentences and edit them again. When multiple submissions occur, the server will merge the different parts in a smart way. For instance, if the user first edits the syntactic tree of sentence 5 and then it loads it again editing only the POS, the system will merge the edited POS into the syntactic structure. If the same user only edit POS for sentence 2, only the POS information is saved, ignoring the syntactic tree.

5.3 Contextual help

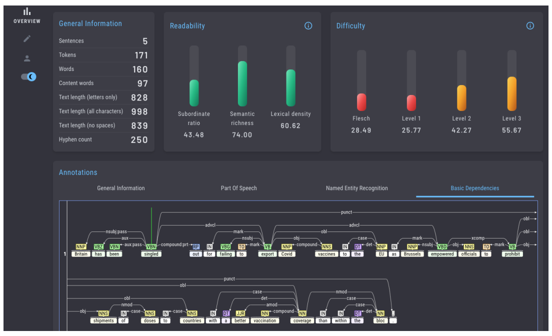
Since Tintful is meant to be used by non-expert users, every screen of the tool provides information buttons, where a quick documentation on how to use that screen is provided (see Figure 10).


Insert text to analyse

Gli impianti chiusi non hanno fermato i tantissimi escursionisti che si sono portati in quota per ciaspolare, praticare scialpinismo e sci nordico. Difficilissimo trovare alcune attrezzature, anche sulle piattaforme dei colossi di vendite online. Luca Bonfante (store manager di Sportler): "Invenduto il 90% degli sci da discesa ma il bilancio regge per le vendite negli altri reparti"

You can also choose an example:

Figure 6: The input text screen of Tintful (light and dark).



General Information

Sentences	5
Tokens	171
Words	160
Content words	97
Text length (letters only)	828
Text length (all characters)	998
Text length (no spaces)	839
Hyphen count	250

Readability

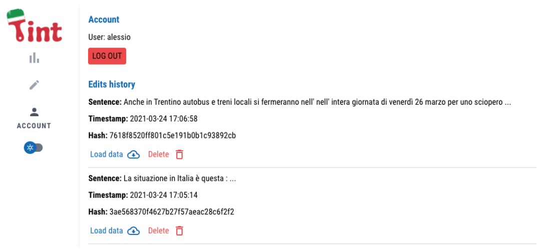
Subordinate ratio	Semantic richness	Lexical density
43.48	74.00	60.62

Difficulty

Flesch	Level 1	Level 2	Level 3
28.49	25.77	42.27	55.67

Annotations

Figure 7: Visualization interface for an English text.


Account

User: alessio

Edits history

Sentence: Anche in Trentino autobus e treni locali si fermeranno nell' nell' intera giornata di venerdì 26 marzo per uno sciopero ...

Timestamp: 2021-03-24 17:06:58

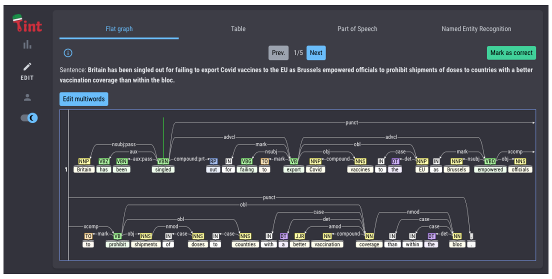
Hash: 7618f8520ff801c5e191b0b1c93892cb

Sentence: La situazione in Italia è questa : ...

Timestamp: 2021-03-24 17:05:14

Hash: 3ae568370f4627b27f57aeac28c6f2f2

Figure 8: The edit history of Tintful.



Sentence: **Britain has been singled out for failing to export Covid vaccines to the EU as Brussels empowered officials to prohibit shipments of doses to countries with a better vaccination coverage than within the bloc.**

Figure 9: An example of the editing interface for an English text.

5.4 Language independence

Tintful is agnostic with respect to the language of the texts. The whole architecture is based (for now, see Section 8) on the Stanford CoreNLP json output. A developer can easily adapt the interface to work with any pipeline that work on top of CoreNLP or that can give the same format.

6 User experience

In developing Tintful, a particular attention has been paid to the interface, given that annotators performing linguistic tasks need to be focused and typically spend a lot of time interacting with the tool.

6.1 Human-Computer interaction

Some tools for linguistic annotations are very generic and can be configured for a potentially infinite set of guidelines. As a side effect of that flexibility, they sometime suffer from slowness in the practical use.

Tintful, instead, is optimized for a small list of possible annotations (syntactic tree, NER, part-of-speech), and therefore the interaction with the user is optimized, to spend as little time as possible for the annotation.

As an example, the NER annotation can be performed just by clicking on the word and looping between the different labels. Since there are only four possibility (PER, LOC, ORG, O), this action can be done very quickly.

On the contrary, the list of tags for part-of-speech is very long, therefore an intermediate modal screen with a dropdown menu is more practical.

6.2 Design and accessibility

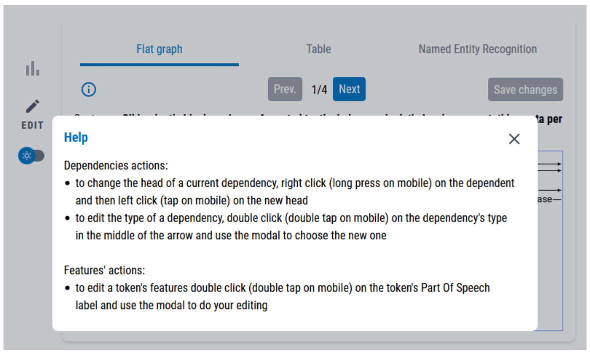
The design is inspired by Material,[14] a set of guidelines, components, and tools developed by Google that support the best practices of user interface design.

The interface also satisfies the most common accessibility guidelines and it is responsive, therefore all the annotation steps can be performed on a tablet or a smartphone.

6.3 Dark mode vs. light mode

The high density of the data in the interface may lead to eyestrain, therefore we add a button that switches the interface between dark and light. The dark mode makes it more comfortable for users to use their devices outside the light hours or in environments with bad lighting conditions (Eisfeld and Kristallovich, 2020; Kim et al., 2019). In addition, reading white text from a black screen or tablet

[14] https://material.io/



Help

Dependencies actions:

- to change the head of a current dependency, right click (long press on mobile) on the dependent and then left click (tap on mobile) on the new head
- to edit the type of a dependency, double click (double tap on mobile) on the dependency's type in the middle of the arrow and use the modal to choose the new one

Features' actions:

- to edit a token's features double click (double tap on mobile) on the token's Part Of Speech label and use the modal to do your editing

Figure 10: An example of contextual help.

may be a way to inhibit myopia (Aleman et al., 2018). When there is enough light, instead, one can have better reading performances on a white background (Piepenbrock et al., 2013). For all these reasons, users can switch independently between the two modes using the button in the Tintful interface. Figure 6 shows an example of the input screen of Tintful in light and dark mode.

6.4 Flat interface

Although in general there is criticism around using flat design for everything (Burmistrov et al., 2015), past studies showed that flat design allows expert users to execute their task faster (Spiliotopoulos et al., 2018). On the contrary, skeuomorphism[15] visually distracts users from intended targets. We therefore decide to use a flat design interface for Tintful: on one side, we want our interface to be as simple as possible; on the other side, NLP is a specialized discipline and we expect our users to have confidence with such tools.

7 Tintful release

The web interface of Tintful is written using VueJS.[16] and the structure of the website is built with Tailwind CSS.[17] The API is written in php and needs a machine with at least version 7 of the interpreter and MySQL server installed. It must be configured to work in a web server (such as Apache or Nginx).

The whole Tintful package is available on GitHub[18] and released under the Apache license.

[15]In graphical user interface design, skeuomorphism is the term describing interface objects that mimic their real-world counterparts in how they appear and how the user can interact with them.

[16]https://vuejs.org/

[17]https://tailwindcss.com/

[18]https://github.com/dhfbk/tintful

8 Conclusions and Future Work

In this paper, we present Tintful, an NLP annotation software that can be used both to annotate texts from scratch and to fix mistakes in NLP pipelines. Differently from other similar tools, data do not need to be in a particular format: starting from plain text, the sentences are first annotated with Stanford CoreNLP, then the user can edit the annotations and submit everything back to the server.

In the future, we will extend the tool to accept more input formats, so that Tintful can work with software different from CoreNLP, such as SpaCy (Honnibal et al., 2020) and UDPipe (Straka, 2018).

In addition, we want to improve the user management part, by creating an admin interface to simplify the user creation. We also want to add the login through external services, such as Google, Github, Facebook, and so on. For registered users, this means that they do not need to remember the password. For casual users, this allows them to review already submitted annotations.

Finally, we are integrating Tintful with the CoreNLP scripts that perform the training of the models (in particular for part-of-speech, dependency parsing and named-entities recognition), to obtain a fullly automatic incremental learning pipeline.

References

Andrea C. Aleman, Min Wang, and Frank Schaeffel. 2018. Reading and myopia: Contrast polarity matters. *Scientific Reports*, 8(1):10840.

Shai Ben-David, John Blitzer, Koby Crammer, Fernando Pereira, et al. 2007. Analysis of representations for domain adaptation. *Advances in neural information processing systems*, 19:137.

Ivan Burmistrov, Tatiana Zlokazova, Anna Izmalkova, and Anna Leonova. 2015. Flat design vs traditional design: Comparative experimental study. In *Human-Computer Interaction – INTERACT 2015*, pages 106–114, Cham. Springer International Publishing.

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.

Henriette Eisfeld and Felix Kristallovich. 2020. The rise of dark mode: A qualitative study of an emerging user interface design trend.

Yang Fan, Fei Tian, Tao Qin, Jiang Bian, and Tie-Yan Liu. 2017. Learning what data to learn. *arXiv preprint arXiv:1702.08635*.

Johannes Heinecke. 2019. ConlluEditor: a fully graphical editor for Universal dependencies treebank files. In *Universal Dependencies Workshop 2019*, Paris.

Felix Hennig and Arne Köhn. 2017. Dependency schema transformation with tree transducers. In *Proceedings of the first Workshop on Universal Dependencies (22 May, Göteborg)*. Universität Hamburg.

Andreas Holzinger. 2013. Human-computer interaction and knowledge discovery (hci-kdd): What is the benefit of bringing those two fields to work together? In *Availability, Reliability, and Security in Information Systems and HCI*, pages 319–328, Berlin, Heidelberg. Springer Berlin Heidelberg.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-strength Natural Language Processing in Python.

Kangsoo Kim, Austin Erickson, Alexis Lambert, Gerd Bruder, and Greg Welch. 2019. Effects of dark mode on visual fatigue and acuity in optical see-through head-mounted displays. In *Symposium on Spatial User Interaction*, SUI '19, New York, NY, USA. Association for Computing Machinery.

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. Association for Computational Linguistics.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60.

Hiroki Nakayama, Takahiro Kubo, Junya Kamura, Yasufumi Taniguchi, and Xu Liang. 2018. doccano: Text annotation tool for human. Software available from https://github.com/doccano/doccano.

Alessio Palmero Aprosio and Giovanni Moretti. 2018. Tint 2.0: an all-inclusive suite for nlp in italian. In *Proceedings of the Fifth Italian Conference on Computational Linguistics CLiC-it*, volume 10, page 12.

C. Piepenbrock, S. Mayr, I. Mund, and A. Buchner. 2013. Positive display polarity is advantageous for both younger and older adults. *Ergonomics*, 56(7):1116–1124.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A Python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*.

Alan Ritter, Sam Clark, Oren Etzioni, et al. 2011. Named entity recognition in tweets: an experimental study. In *Proceedings of the 2011 conference on empirical methods in natural language processing*, pages 1524–1534.

Jeffrey Schlimmer and Doug Fisher. 1986. A case study of incremental concept induction. pages 496–501.

Burr Settles. 2011. From theories to queries: Active learning in practice. In *Active Learning and Experimental Design workshop In conjunction with AISTATS 2010*, volume 16 of *Proceedings of Machine Learning Research*, pages 1–18, Sardinia, Italy. JMLR Workshop and Conference Proceedings.

Richard Socher, Yoshua Bengio, and Christopher D. Manning. 2012. Deep learning for nlp (without magic). In *Tutorial Abstracts of ACL 2012*, ACL '12, page 5, USA. Association for Computational Linguistics.

Konstantinos Spiliotopoulos, Maria Rigou, and Spiros Sirmakessis. 2018. A comparative study of skeuomorphic and flat design from a ux perspective. *Multimodal Technologies and Interaction*, 2(2):31.

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.

Milan Straka. 2018. UDPipe 2.0 prototype at CoNLL 2018 UD shared task. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 197–207, Brussels, Belgium. Association for Computational Linguistics.

Francis M. Tyers, Mariya Sheyanova, and Jonathan North Washington. 2018. Ud annotatrix: An annotation tool for universal dependencies. In *Proceedings of the 16th International Workshop on Treebanks and Linguistic Theories (TLT16)*, pages 10–17.

Various Authors. 1969. *Advances in Instrumentation, Vol. 24: Proceedings of the 24th Annual ISA Conference, Houston, Oktober 27-30, 1969*. pt. 4. Instrument Soc. of America.