POTATO: The Portable Text Annotation Tool

Jiaxin Pei [1] **Aparna Ananthasubramaniam**[1] **Xingyao Wang**[2] **Naitian Zhou** [3]
Apostolos Dedeloudis[4] **Jackson Sargent**[1] **David Jurgens**[1]

[1]School of Information, University of Michigan, Ann Arbor, MI, USA
[2]Department of Computer Science, University of Illinois, Urbana Champaign, IL, USA
[3]School of Information, University of California, Berkeley, CA, USA
[4]The American College of Greece, Athens, Greece
[1]{pedropei, akananth, jackson, jurgens}@umich.edu
[2]xingyao6@illinois.edu [3]naitian@berkeley.edu [4]apostolosdedgmail.com

Abstract

We present POTATO, the **Por**table **t**ext **a**nnotation **to**ol, a free, fully open-sourced annotation system that 1) supports labeling many types of text and multimodal data; 2) offers easy-to-configure features to maximize the productivity of both deployers and annotators (convenient templates for common ML/NLP tasks, active learning, keypress shortcuts, keyword highlights, tooltips); and 3) supports a high degree of customization (editable UI, inserting pre-screening questions, attention and qualification tests). Experiments over two annotation tasks suggest that POTATO improves labeling speed through its specially-designed productivity features, especially for long documents and complex tasks. POTATO is available at https://github.com/davidjurgens/potato and will continue to be updated.

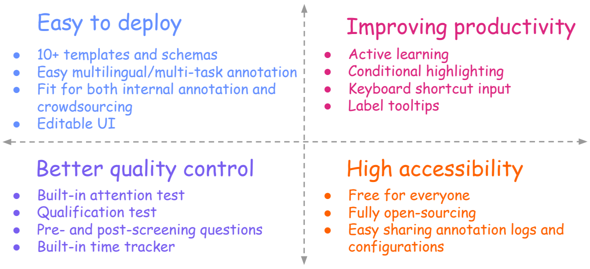


Figure 1: The four core design goals of POTATO: easy to deploy, greater productivity, better quality control, and high accessibility. Each design goal comes with a series of features that can make data annotation easier and more reliable.

1 Introduction

Much of NLP requires annotated data. As NLP has tried to tackle increasingly more complex linguistic phenomena or diverse labeling and classification tasks, the annotation process has increased in complexity—yet the need for and benefit of large labeled datasets remain (Halevy et al., 2009; Sun et al., 2017). Modern annotation tools like Label Studio (Tkachenko et al., 2021), LightTag (Perry, 2021), Doccano (Nakayama et al., 2018), and Prodigy (Explosion, 2017) have partially filled this gap, providing a variety of solutions to different types of annotations. However, these tools each bring their own challenges: requiring external access, limiting visual configurability for complex tasks, or even costing hundreds of dollars—prohibitive for small groups. We introduce POTATO, The **Por**table **t**ext **a**nnotation **to**ol, which allows practitioners to quickly design and deploy complex annotations tasks.

POTATO has been designed, developed, and tested over a two-year period with the following design goals in mind (Figure 1): 1) **High accessibility**. POTATO is open-sourced under the MIT license and free to anyone. POTATO is built with minimal dependencies to allow researchers and developers to easily build and integrate their own features. 2) **Easy to deploy**. POTATO comes with templates covering a wide range of annotation tasks like best-worst-scaling, text classification, and multi-modal conversation. Anyone can start a new annotation project with simple configurations. POTATO is also rapidly and easily deployable in local and web-based configurations and has seamless integration with common crowdsourcing platforms like Amazon Mechanical Turk and Prolific. POTATO flexibly supports diverse annotation needs. With our specially designed schema rendering and custom rendering mechanism, POTATO allows nearly all kinds of text annotation tasks and is visually customizable to support complex task designs and layout. 3) **Better quality control**. Attaining reliable annotations is one of the core goals of data annotation tasks. POTATO is designed with a series of features that can help to improve annotation quality, including built-in attention tests, prestudy qualification tests, and an annotation time tracker. POTATO also allows deployers to easily set up pre- and post-

Proceedings of the The 2022 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, pages 327 - 337
December 7-11, 2022 ©2022 Association for Computational Linguistics

screening questions (e.g. demographics or psychological surveys), which can help researchers to better understand potential biases in data labeling. 4) **Productivity enhancing**. POTATO comes with a series of productivity features for both deployers and annotators like active learning, conditional highlighting, and keyboard shortcuts. While existing systems like Doccano (Nakayama et al., 2018) and Lighttag (Perry, 2021) offer different subsets of these features, POTATO aims to support a holistic annotation experience by meeting all of these needs. Experiments on two annotation tasks demonstrate that POTATO leads to more efficient data labeling for complex tasks.

2 Architecture and Design

POTATO is written in Python and focuses on portability and simplicity in annotation and deployment. The user interface is created through an extensible HTML template and configuration file, which allows practitioners to quickly develop and deploy common setups like Likert-scale annotation while also supporting extensive display customization. The POTATO server populates the interface with data provided by the operator and supports displaying any HTML-supported modality, e.g., text or images. An overview of the architecture is shown in Figure 2.

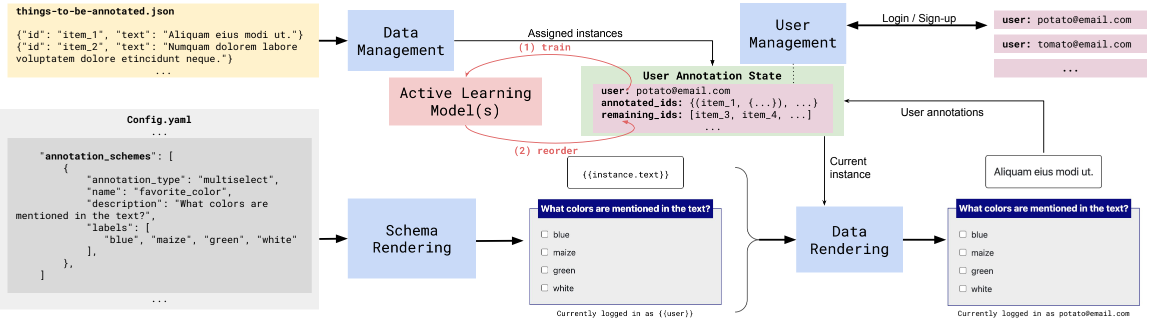
Data Management POTATO loads data in common file formats, such as delimited files or newline-delimited JSON. This allows it to ingest data in the JSON format supplied by the Twitter or Reddit APIs, as well as other types of data used by the deployer. All formats are converted into internal data structures that link the deployer-selected instance ID to annotations. At a minimum, deployers must specify which field represents the unique instance ID and, for most tasks, the text to be annotated. The data may contain other columns which will be included in the final output and can optionally be used in customized visualizations.

Annotation Schema Rendering POTATO allows deployers to select one or more forms of annotation for their data using predefined schema types, such as multiple choice or best-worst scaling. Deployers fill out which options should be shown and then each scheme is rendered into HTML upon the completion of loading the data. Annotation instructions can be provided as an external URL that annotators may view or using HTML text shown in POTATO that annotators may collapse vertically to free up screen space. POTATO provides default HTML templates that automatically lay out each scheme's annotation questions. However, deployers may additionally customize the HTML templates and select their own layout using JINJA expressions (e.g., `{{text}}`; Jinja) to specify where parts of the annotation task and data are populated within the user-defined template.

User Management Annotators create accounts and then log in to view their tasks using a secure user management system. When used with crowdsourcing platforms, POTATO also allows workers to directly jump to the annotation task using their crowdsourcing user ID. For each new annotator, POTATO automatically assigns instances as configured by the deployer and all the annotations are recorded on the backend. When logging out and back in, annotators resume at the most recent unannotated item. POTATO also allows deployers to, with minimum configurations, set up pre- and post-screening questions (e.g., having annotators provide demographics or complete psychological questionnaires), pre-study tests, and attention tests to identify unreliable annotators.

Active Learning In some settings such as data with imbalanced classes, active learning helps re-order items to surface those that may provide more information to downstream classifiers (Settles, 2009; Monarch, 2021). Prior annotation interfaces have included active learning to help maximize data utility (Stenetorp et al., 2012; Wiechmann et al., 2021; Li et al., 2021). POTATO includes a configurable active learning setup to prioritize important samples and potentially improves data quality with limited labeling budgets. In its default setting, POTATO periodically trains a logistic regression classifier using unigram and bigram features on the currently annotated data; unlabeled instances are sorted by classifier confidence and items with low confidence are prioritized, while still including a deployer-specified percentage of a random sample. Deployers may change or reconfigure this model easily.

Design highlights POTATO is designed to flexibly support diverse annotation tasks and improve the productivity of annotators. Here we briefly highlight several features of POTATO. First, with simple configurations, deployers can quickly add *keyboard shortcuts* to specific options or *tooltips* to help annotators. Second, in settings where an annotator is reading a dense or long passage, or where there



```
things-to-be-annotated.json

{"id": "item_1", "text": "Aliquam eius modi ut."}
{"id": "item_2", "text": "Numquam dolorem labore
voluptatem dolore etincidunt neque."}
                              ...
```

```
                              Config.yaml
                                  ...

    "annotation_schemes": [
        {
            "annotation_type": "multiselect",
            "name": "favorite_color",
            "description": "What colors are
mentioned in the text?",
            "labels": [
               "blue", "maize", "green", "white"
            ],
        },
    ]
                                  ...
```

```
User Annotation State
 user: potato@email.com
 annotated_ids: {(item_1, {...}), ...}
 remaining_ids: [item_3, item_4, ...]
                ...
```

```
user: potato@email.com
user: tomato@email.com
...
```

Figure 2: The overall architecture of POTATO features a modular design that decouples the task specification from the rendering, allowing rapid deployment of new task designs and separate customization of the visualization.

are many potentially subtle cues, annotators are likely to struggle due to having to slowly and carefully read each passage or accidentally omitting relevant annotations due to the complexity of the task. POTATO introduces a new feature, *conditional highlighting*, to help in these settings, where the deployer specifies certain keywords to trigger highlights in the text, drawing the annotator's focus to those words or phrases. For example, if annotating for Twitter-based stance towards politicians, a deployer might use keywords and phrases for common politicians or political parties to ensure these are not missed. If conditional highlighting is enabled, POTATO will also randomly label some words with highlights, based on a deployer-specified rate, to ensure annotators do not rely too heavily on highlights.

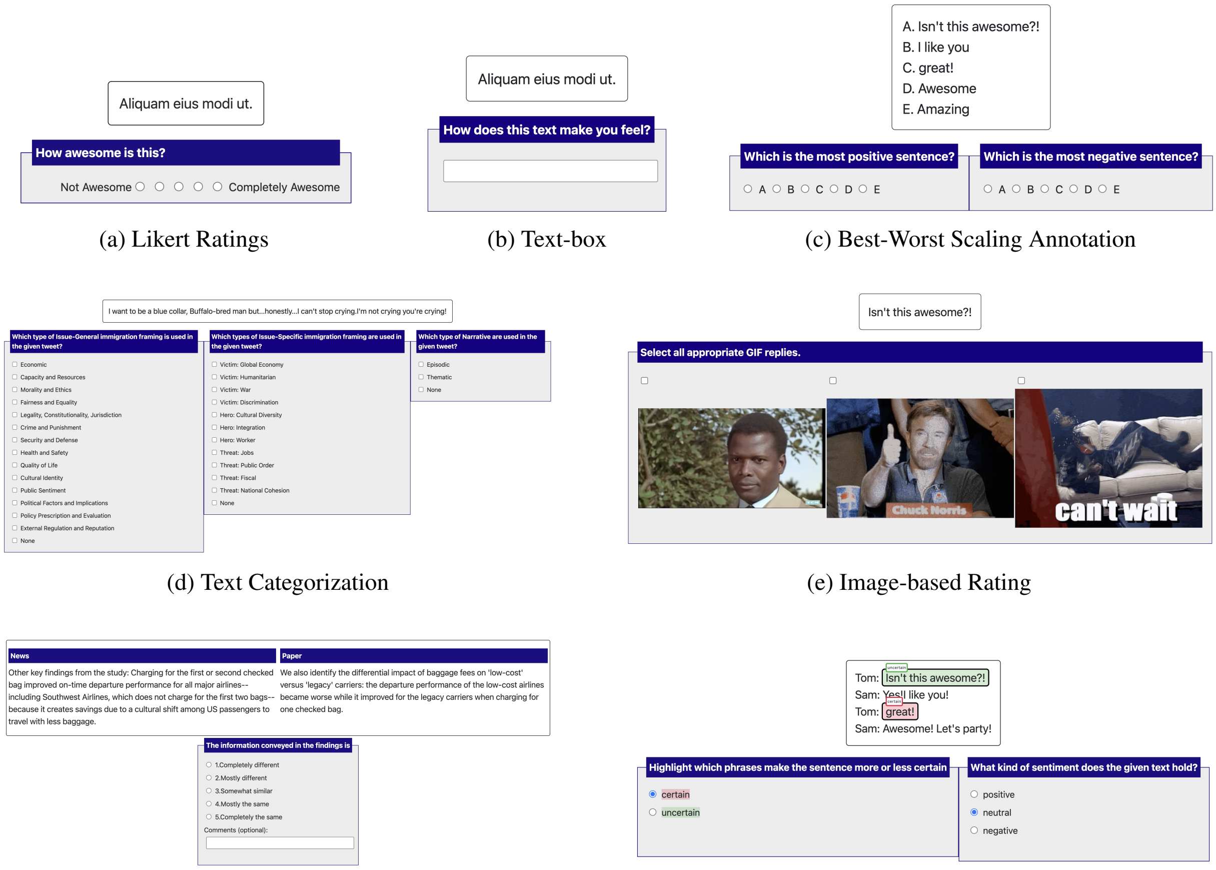
3 Deployment and Tasks

POTATO is designed with a quickly-deployable Python-based server architecture that can be run locally or hosted on any device. To launch a POTATO instance, the deployer first defines a YAML file that specifies the annotation schemes, data sources, server configuration, and any custom visualizations. If POTATO is launched without a YAML, the program will provide the deployer the option of following a series of prompts about their task to automatically generate a YAML file for them. A YAML file is then passed to the server on the command line to launch the server for annotation.

Currently, POTATO supports eight annotation scheme types: multiple-selection (checkboxes), single-selection (radio buttons), best-worst scaling, Likert scale, free-form text, span-based labels, numbers, and dropdown list. Deployers can easily set up one or more of these schemes in the YAML file—e.g., asking annotators to rate a news article on different dimensions using multiple Likert scales and then summarizing the article in a free text response. For each annotation instance, POTATO can take a single document, multiple documents as a list (e.g. dialogue and best-worst-scaling), as well as a dictionary of documents (e.g. a pair of documents for pairwise comparison). POTATO will automatically display the instance to annotators based on the input types and the YAML configurations. The POTATO documentation contains example YAML templates for several common annotation tasks such as sentiment analysis, question-answering, and image-based labeling.

POTATO is self-hosted and can be served locally or exposed publicly. Each running instance of POTATO serves one task, but multiple annotation tasks can be stored in a single *installation*, to be served using different configuration files at different times. POTATO allows flexible ways for annotators to login. For internal usage, POTATO allows annotators to sign up and log in with email addresses. POTATO also allows annotators to directly log in with a URL argument (e.g. username in the crowdsourcing platform), which can be used in crowdsourcing settings where a dedicated link is created for an annotation task. POTATO has been tested with popular crowdsourcing platforms including Prolific and Amazon Mechanical Turk.

POTATO has been deployed in a variety of annotation settings over a two-year period, including a 27-class annotation scheme for classifying immigration framing (Mendelsohn et al., 2021); rating condolences and empathy for Reddit comments with hundreds of words (Zhou and Jurgens, 2020); best-worst scaling for rating intimacy in questions (Pei and Jurgens, 2020); rating Reddit threads for their prosociality (Bao et al., 2021); rating, on a Likert-scale, sentences for scientific uncertainty



Aliquam eius modi ut.

How awesome is this?

Not Awesome ○ ○ ○ ○ ○ Completely Awesome

(a) Likert Ratings

Aliquam eius modi ut.

How does this text make you feel?

(b) Text-box

A. Isn't this awesome?!
B. I like you
C. great!
D. Awesome
E. Amazing

Which is the most positive sentence?

○ A ○ B ○ C ○ D ○ E

Which is the most negative sentence?

○ A ○ B ○ C ○ D ○ E

(c) Best-Worst Scaling Annotation

I want to be a blue collar, Buffalo-bred man but...honestly...I can't stop crying.I'm not crying you're crying!

Which type of Issue-General immigration framing is used in the given tweet?

- □ Economic
- □ Capacity and Resources
- □ Morality and Ethics
- □ Fairness and Equality
- □ Legality, Constitutionality, Jurisdiction
- □ Crime and Punishment
- □ Security and Defense
- □ Health and Safety
- □ Quality of Life
- □ Cultural Identity
- □ Public Sentiment
- □ Political Factors and Implications
- □ Policy Prescription and Evaluation
- □ External Regulation and Reputation
- □ None

Which types of Issue-Specific immigration framing are used in the given tweet?

- □ Victim: Global Economy
- □ Victim: Humanitarian
- □ Victim: War
- □ Victim: Discrimination
- □ Hero: Cultural Diversity
- □ Hero: Integration
- □ Hero: Worker
- □ Threat: Jobs
- □ Threat: Public Order
- □ Threat: Fiscal
- □ Threat: National Cohesion
- □ None

Which type of Narrative are used in the given tweet?

- □ Episodic
- □ Thematic
- □ None

(d) Text Categorization

Isn't this awesome?!

Select all appropriate GIF replies.

□ □ □

(e) Image-based Rating

News	Paper
Other key findings from the study: Charging for the first or second checked bag improved on-time departure performance for all major airlines--including Southwest Airlines, which does not charge for the first two bags--because it creates savings due to a cultural shift among US passengers to travel with less baggage.	We also identify the differential impact of baggage fees on 'low-cost' versus 'legacy' carriers: the departure performance of the low-cost airlines became worse while it improved for the legacy carriers when charging for one checked bag.

The information conveyed in the findings is

- ○ 1.Completely different
- ○ 2.Mostly different
- ○ 3.Somewhat similar
- ○ 4.Mostly the same
- ○ 5.Completely the same

Comments (optional):

(f) Pairwise comparison

Tom: Isn't this awesome?!
Sam: Yes!I like you!
Tom: great!
Sam: Awesome! Let's party!

Highlight which phrases make the sentence more or less certain

- ● certain
- ○ uncertain

What kind of sentiment does the given text hold?

- ○ positive
- ● neutral
- ○ negative

(g) Span-based dialogue analysis

Figure 3: Screenshots of example tasks supported by POTATO, which are included as templates. Examples 3a-3c show single-task annotations, while Example 3d shows a multitask setup with three multi-select labels. Example 3e shows how POTATO supports multimedia as annotation options. 3f shows a pairwise Likert annotation and 3g shows a span-based annotation for dialogue analysis. Examples omit the common interface header that shows annotators how many instances remain and links to the annotation codebook.

(Pei and Jurgens, 2021), intimacy in multilingual tweet (Pei et al., 2022) and similarity in scientific findings (Wright et al., 2022); and rating the appropriateness of GIF replies to messages, which showed an animated GIF in the interface (Wang and Jurgens, 2021).

Figure 3 shows some of the interfaces from our documentation's example templates. These templates cover a wide range of NLP tasks and can be easily adapted to support a quick start of common annotation tasks. The configuration-based setup of POTATO allows researchers to easily share their annotation settings and replicate annotation settings used by existing works. POTATO also comes with a dedicated project hub where researchers can easily open-source their annotation project and already includes projects in our previous studies. Such a feature could help to improve the replicability of NLP/ML annotations and we welcome submissions from the entire research community.

4 Comparison with Existing Systems

POTATO has been developed to fill a key niche left by existing systems for providing visual customization, easy annotator-support features, and rapid development. The ultimate goal is to provide simple and comprehensive solutions to common annotation tasks as well as allow personalized design for complex tasks. Table 1 shows the comparisons between POTATO and other common text annotation tools over a series of important dimensions including flexibility, productivity, quality, and accessibility. We highlight major differentiators next. Please note that we only compare annotation systems that are currently available for anyone to use, free of cost.

Flexibility POTATO is designed to maximize flexibility for a variety of annotation settings. For common annotation tasks like text classification, POTATO comes with a wide range of templates

		Label Studio	Doccano	FLAT	LightTag	Prodigy	Tagtog	FITAnnotator	BRAT	WebAnno/INCEpTION	POTATO
Flexibility	Multiple Schema	✓			✓		✓		✓	✓	✓
	Multimodal	✓	✓		✓	✓					✓
	Span-Based Annotation	✓	✓		✓	✓	✓		✓	✓	✓
	Editable UI	✓		✓		✓		✓			✓
Productivity	Active Learning	✓*				✓		✓		✓	✓
	Conditional Highlighting										✓
	Keyboard Shortcuts	✓	✓		✓	✓			✓		✓
Quality control	Prestudy Qualification Test										✓
	Attention Test										✓
	Behavioral Tracking										✓
	Pre- and Post-screeing Questions										✓
Accessibility	Open-Source	✓	✓	✓					✓	✓	✓
	Easy Sharing and Replicating										✓
	Price	Free	Free	Free	Free for academia	$390	$59/person/month	Not available	Free	Free	Free

Table 1: Comparisons between POTATO and other text annotation systems over four themes. * means the feature is not available for the free plan.

and allows a quick start for deployers. However, unlike many existing annotation tools, which provide fixed user interfaces with selected types of annotation tasks (e.g., Doccano offers neither templates nor an editable UI (Nakayama et al., 2018)), POTATO allows deployers to customize their own annotation interface to support diverse needs. For example, Wang and Jurgens (2021) used animated GIFs as the labels in the annotation and Mendelsohn et al. (2021) used a 27-class scheme under three categories, both of which required visual customization to make the task feasible. POTATO also allows deployers to easily set up unlimited numbers of similar annotation tasks, which can be especially helpful for multilingual annotations. For example, for all the other data annotation systems, the deployers need to set up separate tasks and guidelines for each language. With POTATO, deployers only need to create a sheet containing translated guidelines and POTATO's built-in script can help to generate annotation sites for each language.

Productivity POTATO is designed to maximize the productivity of both annotators and deployers. While most of the existing annotation tools generally focus on labeling data, POTATO supports a series of features that can help with the entire data annotation pipeline. POTATO allows easily-customizable keyboard shortcuts to allow efficient annotation. For visually or cognitively challenging settings, POTATO allows conditional highlights, which helps to reduce task complexity and focus annotators' attention. Finally, active learning can reduce the annotation time needed to curate an informative dataset. Often, annotation tools that offer a highly customizable annotation interface do not also implement productivity features: the open-sourced version of LabelStudio (Tkachenko et al., 2021) only supports keypress shortcuts, while Flat (Gompel et al., 2017) supports none of these features. For deployers, POTATO allows seamless integration with common crowdsourcing platforms like Amazon Mechanical Turk and Prolific.

Quality control Collective high-quality and reliable annotations is the ultimate goal of data labeling tasks and is usually the key to the success of the final ML/NLP systems. POTATO comes with a series of quality control feature which helps deployers to reliably collect annotations. While some other annotation systems like Label Studio and WebAnno also support agreement calculation, none of the existing systems come with features that help deployers to improve the annotation quality and analyze factors affecting it. POTATO allows deployers to easily set up prestudy qualification tests (annotators have to pass a small test to participate in the full annotation) and attention tests (attention test questions are randomly inserted in the annotation queue as configured by the deployer) to identify unreliable annotators before, within, and after annotation. POTATO also allows deployers to freely insert survey questions before and after the annotation phase. Deployers can easily define different pages of pre- and post-screening questions with minimum effort and POTATO also provides a series of templates for common survey questions like user consent and demographic information. Recent studies suggest that the background of annotators has substantial effects on the quality and bias of data labeling and further affects the fairness of ML/NLP models (e.g., Sap et al., 2022). With POTATO, researchers can easily collect background information of annotators and analyze the effect of annotator backgrounds on data labeling.

Accessibility POTATO is free to use and actively maintained. While commercial annotation tools like Prodigy (Explosion, 2017) can come with more functionality, these tools are expensive; for example, Prodigy costs $390 USD for individual users, and Tagtog (Cejuela et al., 2014) costs at least $59 USD per person per month. These costs are potentially prohibitive for students and researchers without access. POTATO is fully open-sourced and is deployed with minimum dependencies. Moreover, in addition to giving the flexibility to freely define UIs and annotation settings, POTATO allows researchers to easily share their annotation settings with a simple YAML file, aiding in replication and future extension of prior work.

5 Experimental Analysis

POTATO was designed to minimize set-up time and per-instance annotation time while maintaining annotation accuracy. Therefore, we conduct a user study to compare the time for setup and annotation time per instance compared to its free competitors. We compare POTATO's performance on two long and complex tasks, each involving identifying themes and causes in narrative summaries of reports of completed suicides:

- **Task 1** contains long documents, with two annotation schemes with a total of 22 labels. The task requires labeling whether each narrative contains each of 13 work-related transitions (e.g., retirement, layoffs) and 9 housing-related transitions. The average document contains 13.4 sentences and 1,180 characters.

- **Task 2** is comprised of shorter documents, with the same tasks and labels. This alternative version shows only a single sentence of the narrative in Task 1 and asks annotators to judge whether each contains the same categories. The average sentence (annotation item) contains 88 characters.

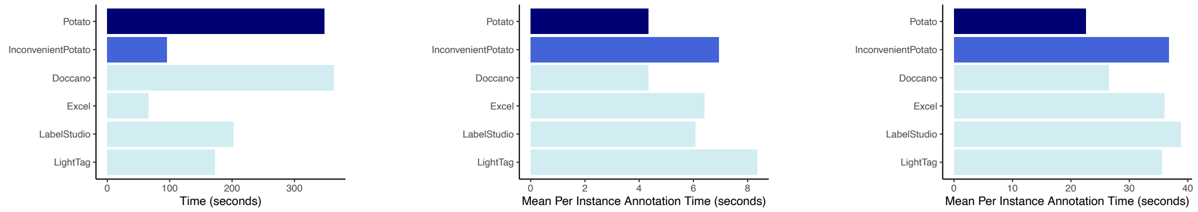
Annotation Setup One annotator completed 50 annotations for Task 1 and 100 annotations for Task 2 on POTATO and a number of freely available and feature-rich annotation tools: Microsoft Excel, Doccano (Nakayama et al., 2018), Label Studio (Tkachenko et al., 2021), and LightTag (Perry, 2021). The annotator was highly familiar with the task and classification scheme, having annotated 1000+ instances of each task prior to this user study, so familiarity with the codebook was not a factor. Given the complexity of the task, POTATO was initialized with 118 keywords to associate with conditional highlights (e.g., retir*, layoff, work*), key bindings, and classes included tooltips summarizing each label. To test the effect of these productivity-enhancing features, we include a version of POTATO that does not include these features, called Inconvenient POTATO. To ensure the same level of familiarity with each document, the documents annotated with each tool are randomly sampled from a larger set of 203,531 documents.

For each annotation tool we measure the time to set up an annotation task without counting time taken to (1) install and familiarize ourselves with the tool (e.g., trial and error in set-up), (2) generate the annotation data files, and (3) write the properties of each label and keywords. Each tool was configured as comparably as possible (e.g., keypress shortcuts were always enabled and active learning disabled). To reduce the influence of initial unfamiliarity with each tool on per-instance timing, the annotator completed 10 untimed instances. Then, for each tool, we record the time spent annotating per instance.

Results Across both annotation tasks, POTATO is approximately 30-50% faster than competitors like Excel, Label Studio, and LightTag (Figure 4b-c). Annotating short documents is comparable in time to Doccano (4b), while long documents are slightly faster in POTATO (4c). Without convenience features like conditional highlighting, key mappings, and tooltips, POTATO's per-instance annotation time increases to be more comparable with other tools. We conclude that the difference in per-instance annotation time is likely attributable to these design features.

Including convenience features increases task setup time (Figure 4a), taking just over 4 extra minutes to configure at set-up. Our results suggest that, compared to using POTATO with convenience features, the base setup without convenience features has lower overall task time (including setup) until an annotator has seen ∼20 long documents or ∼100 short documents. We note that POTATO without convenience features takes less time to set up than Label Studio and LightTag; with these three features, POTATO takes roughly the same amount of time to set up as Doccano, even though Doccano only supports one feature (keypress shortcut).

The two tasks we chose share features with many common NLP annotation tasks that make them well-suited to a system like POTATO. We highlight two comparative observations across the interface from the user study. First, Doccano and POTATO have the most annotator-friendly interfaces, which



(a) Task Setup Time

(b) Short Document, 2 tasks, 22 labels

(c) Long Document, 2 tasks, 22 labels

Figure 4: Times from our user study (§5) to (a) set up a task; (b) annotate one short document or (c) annotate one long document show the time savings of POTATO.

allow for fast coding. For instance, custom keypress shortcuts allow us to create 22 different bindings that make logical sense to the annotator. Unlike other tools, these two tools did not require any use of the mouse (e.g., others required pressing submit with the mouse), which reduced the annotation time; in short document tasks, where the time to read the document is low, these time savings become especially important. The default page layout in POTATO also better supports content interpretation; for example, the text and labels frequently fit on one page with no scrolling, and by placing the text on top of the labels, the annotator did not need to scroll down in order to read the text—and since the annotator had the keypress bindings memorized, the task could often be accomplished with no scrolling. Finally, since other tools often required uploading data to external servers, there was often a load time of 1-2 seconds per document; again, saving this time with locally-deployed tools was especially salient with short documents.

Second, both tasks involve assigning a large number of independent labels. The keyword highlights allow the annotator to quickly identify which subset of these labels are likely to be relevant to the document, while the key mappings allow them to quick apply the correct labels. Keyword highlights are particularly useful for longer documents (e.g., Task 1), because they help identify the text that is likely relevant to a given label, which is often buried in a large amount of irrelevant text (e.g., most labels apply to one of ∼13 sentences). Additionally, if none of the labels applied to a document, the annotator needed to ensure that she did not overlook a relevant phrase in the long text; in POTATO the lack of relevant keywords allowed us to quickly confirm that none of the labels applied, while without keyword highlights, the annotator read these documents twice. The time savings associated with keyword highlights likely explains the slight per-instance advantage POTATO has over Doccano for long documents but not short documents.

6 Conclusion and Future Plans

POTATO distinguishes itself with a comprehensive suite of productivity-enhancing features that allow annotators to efficiently and accurately label data and researchers to quickly configure complex tasks on a wide range of data types. POTATO was created both for the computational scholar and the overburdened student or crowdworker, looking to annotate more data in their limited time.

POTATO has been in continuous development for over two years and will continue to be developed to support new task designs, easier management, and faster annotation.

For management, we aim to have (1) a unified GUI for deployers to create new tasks and manage existing tasks, (2) a GUI that supports real-time monitoring of annotation process, mirroring tools like Webanno (Yimam et al., 2013), and (3) integration with common social media platforms to display content with original interfaces (e.g. displaying tweets with their native UI).

For annotators, we aim to support simple linguistic search to help annotators find and prioritize instances to annotate, and to support personalization in aspects such as annotators' visualization and keybindings. We also plan to conduct experiments and explore different design choices to reduce annotator burn-out.

Acknowledgments

The authors thank members of the Blalablab and all the annotators who have used POTATO throughout the years for their feedback on how to make POTATO better. This material is based in part upon work supported by the National Science Foundation under Grant No IIS-1850221.

7 Ethics and Broader Impacts

As a highly configurable annotation tool, POTATO's biggest ethical and societal implications will likely come from the questions the tool is used to answer and the ways in which researchers choose to deploy the tool. POTATO was built with accessibility, social responsibility, and usefulness at the forefront, and the tool's default settings afford a range of values-driven practices, which we will discuss below. However, a major risk is that POTATO requires researchers to self-regulate when encouraging researchers to opt into ethical values often proves unsuccessful (Hagendorff, 2020). For instance, the tool does not build in any safeguards against unethical questions or harmful applications (Buolamwini and Gebru, 2018; Mitchell et al., 2019; Benjamin, 2019) and does not actively prevent the exploitation of crowdworkers (Irani and Silberman, 2013; Shmueli et al., 2021). Moreover, since POTATO is a tool designed to improve the efficiency of typical prediction task workflows, it cannot address existential critiques of machine learning (e.g., harms of classification as a practice).

POTATO was created using principles of universal design, prioritizing broadly experienced ease of use, low effort, intuitiveness, flexibility, tolerance for error, and perceptibility of key information (Persson et al., 2015). Since POTATO is uniquely annotator-focused, rather than deployer-focused, tasks are readily designed in a way that maximizes worker wellbeing and productivity. The application's design is largely accessible and inclusive and the tool contains many of the types of features that crowdworkers find useful (e.g., low effort to configure, reduces cognitive burden of complex tasks, easy to correct errors by going back, login flow that supports screen readers and doesn't use captcha, annotation guidelines readily visible in tooltip and hyperlink) (Zyskowski et al., 2015; Swaminathan et al., 2017). That said, certain features of the interface may be inaccessible for workers. Moreover, tools like POTATO can worsen annotators' mental health by promoting fragmented work, multitasking, and poor work-life balance (Williams et al., 2019) and by displaying triggering text without masks or warnings (Shmueli et al., 2021). Many of these potential accessibility and psychological harms can be addressed through improvements in the interface. Because of the ease of secondary development — especially adding new HTML front-end templates — POTATO allows the research community to explore more design opportunities for inclusive annotation and responsible crowdsourcing. Ideally, a future version of this tool would use community-led design to develop more universally accessible, inclusive templates for users (Spiel et al., 2020).

In designing POTATO, we prioritized developing mechanisms for just, equitable compensation. By allowing annotators to track time spent on the task, POTATO facilitates paying crowdworkers a fair hourly wage rather than the per-task payment schemes that frequently lead to low hourly wages (Fort et al., 2011; Gray and Suri, 2019). A key accessibility feature, the timer promotes flexibility (e.g., allows people to take longer or build in microbreaks) instead of imposing needlessly restrictive per-task time requirements that can create barriers for disabled workers (Zyskowski et al., 2015). Our goal in creating POTATO was to empower and support the annotator. For instance, although we piloted a timer to alert annotators when the expected task time had elapsed, we ultimately removed this feature in order to eliminate additional stress.

Another important problem in computational social research is inaccurately labeled and biased datasets, which are a cause of inequitably felt downstream harms (Olteanu et al., 2019; Blodgett et al., 2020; Mehrabi et al., 2021). POTATO may have the potential to reduce many common sources of bias by promoting high-quality annotations: convenience features lower cognitive load and reduce reliance on personal heuristics that may increase bias; researchers can use tooltips to provide specific, easily accessed instructions to minimize anticipated sources of bias; since the base annotation time is lower, and there are no per-instance annotation time limits, annotators may feel less pressure to label faster at the expense of poor annotation quality. However, POTATO may amplify the researchers' own biases in the data: annotators may rely too heavily on keyword highlights and tooltips, which can bias the data if keywords common in minority communities are over- or underrepresented, or the tooltip text does not include instructions relevant to certain communities in the data. Future experiments can study the effect of POTATO's productivity-enhancing features on mitigating or amplifying different types of bias.

Finally, an important goal in developing POTATO was to facilitate studying complex social questions without being limited by existing labeled data:

since the tool makes it easier and faster to design complex tasks and collect data, researchers can think critically about what problems would be most beneficial and impactful, and design tasks that actually answer those questions (Wiens et al., 2019; Abebe et al., 2020). Since POTATO facilitates the deployment of multilingual tasks, researchers can more easily test the the generalizability of their results across linguistic and cultural contexts (Joshi et al., 2020). A major challenge in applied machine learning is the lack of diversity among researchers (Orife et al., 2020); since POTATO is free, open-sourced, and easy to use, we hope the tool will facilitate participation by scholars who are not associated with well-funded R1 universities and also by community members outside academia.

References

Rediet Abebe, Solon Barocas, Jon Kleinberg, Karen Levy, Manish Raghavan, and David G Robinson. 2020. Roles for computing in social change. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, pages 252–260.

Jiajun Bao, Junjie Wu, Yiming Zhang, Eshwar Chandrasekharan, and David Jurgens. 2021. Conversations gone alright: Quantifying and predicting prosocial outcomes in online conversations. In *Proceedings of the Web Conference 2021*, pages 1134–1145.

Ruha Benjamin. 2019. Race after technology: Abolitionist tools for the new jim code. *Social Forces*.

Su Lin Blodgett, Solon Barocas, Hal Daumé III, and Hanna Wallach. 2020. Language (technology) is power: A critical survey of "bias" in nlp. *arXiv preprint arXiv:2005.14050*.

Joy Buolamwini and Timnit Gebru. 2018. Gender shades: Intersectional accuracy disparities in commercial gender classification. In *Conference on fairness, accountability and transparency*, pages 77–91. PMLR.

Juan Miguel Cejuela, Peter McQuilton, Laura Ponting, Steven J Marygold, Raymund Stefancsik, Gillian H Millburn, Burkhard Rost, FlyBase Consortium, et al. 2014. Tagtog: interactive and text-mining-assisted annotation of gene mentions in plos full-text articles. *Database*, 2014.

Explosion. 2017. Prodigy. `https://prodi.gy`.

Karën Fort, Gilles Adda, and K Bretonnel Cohen. 2011. Amazon mechanical turk: Gold mine or coal mine? *Computational Linguistics*, 37(2):413–420.

M van Gompel, K Sloot, Martin Reynaert, and APJ van den Bosch. 2017. Folia in practice. the infrastructure of a linguistic annotation format.

Mary L Gray and Siddharth Suri. 2019. *Ghost work: How to stop Silicon Valley from building a new global underclass*. Eamon Dolan Books.

Thilo Hagendorff. 2020. The ethics of ai ethics: An evaluation of guidelines. *Minds and Machines*, 30(1):99–120.

Alon Halevy, Peter Norvig, and Fernando Pereira. 2009. The unreasonable effectiveness of data. *IEEE Intelligent Systems*, 24(2):8–12.

Lilly C Irani and M Six Silberman. 2013. Turkopticon: Interrupting worker invisibility in amazon mechanical turk. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 611–620.

Jinja. JINJA template designer documentation. `https://jinja.palletsprojects.com/en/3.0.x/templates/`.

Pratik Joshi, Sebastin Santy, Amar Budhiraja, Kalika Bali, and Monojit Choudhury. 2020. The state and fate of linguistic diversity and inclusion in the nlp world. *arXiv preprint arXiv:2004.09095*.

Yanzeng Li, Bowen Yu, Li Quangang, and Tingwen Liu. 2021. FITAnnotator: A Flexible and Intelligent Text Annotation System. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Demonstrations*, pages 35–41.

Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. 2021. A survey on bias and fairness in machine learning. *ACM Computing Surveys (CSUR)*, 54(6):1–35.

Julia Mendelsohn, Ceren Budak, and David Jurgens. 2021. Modeling framing in immigration discourse on social media. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2219–2263.

Margaret Mitchell, Simone Wu, Andrew Zaldivar, Parker Barnes, Lucy Vasserman, Ben Hutchinson, Elena Spitzer, Inioluwa Deborah Raji, and Timnit Gebru. 2019. Model cards for model reporting. In *Proceedings of the conference on fairness, accountability, and transparency*, pages 220–229.

Robert Munro Monarch. 2021. *Human-in-the-Loop Machine Learning: Active learning and annotation for human-centered AI*. Simon and Schuster.

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.

Alexandra Olteanu, Carlos Castillo, Fernando Diaz, and Emre Kıcıman. 2019. Social data: Biases, methodological pitfalls, and ethical boundaries. *Frontiers in Big Data*, 2:13.

Iroro Orife, Julia Kreutzer, Blessing Sibanda, Daniel Whitenack, Kathleen Siminyu, Laura Martinus, Jamiil Toure Ali, Jade Abbott, Vukosi Marivate, Salomon Kabongo, et al. 2020. Masakhane–machine translation for africa. *arXiv preprint arXiv:2003.11529*.

Jiaxin Pei and David Jurgens. 2020. Quantifying intimacy in language. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5307–5326.

Jiaxin Pei and David Jurgens. 2021. Measuring sentence-level and aspect-level (un) certainty in science communications. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9959–10011.

Jiaxin Pei, Vítor Silva, Maarten Bos, Yozon Liu, Leonardo Neves, David Jurgens, and Francesco Barbieri. 2022. Semeval 2023 task 9: Multilingual tweet intimacy analysis. *arXiv preprint arXiv:2210.01108*.

Tal Perry. 2021. Lighttag: Text annotation platform. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 20–27.

Hans Persson, Henrik Åhman, Alexander Arvei Yngling, and Jan Gulliksen. 2015. Universal design, inclusive design, accessible design, design for all: different concepts—one goal? on the concept of accessibility—historical, methodological and philosophical aspects. *Universal Access in the Information Society*, 14(4):505–526.

Maarten Sap, Swabha Swayamdipta, Laura Vianna, Xuhui Zhou, Yejin Choi, and Noah A. Smith. 2022. Annotators with attitudes: How annotator beliefs and identities bias toxic language detection.

Burr Settles. 2009. Active learning literature survey.

Boaz Shmueli, Jan Fell, Soumya Ray, and Lun-Wei Ku. 2021. Beyond fair pay: Ethical implications of nlp crowdsourcing. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3758–3769.

Katta Spiel, Kathrin Gerling, Cynthia L Bennett, Emeline Brulé, Rua M Williams, Jennifer Rode, and Jennifer Mankoff. 2020. Nothing about us without us: Investigating the role of critical disability studies in hci. In *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–8.

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.

Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. 2017. Revisiting unreasonable effectiveness of data in deep learning era. In *Proceedings of the IEEE international conference on computer vision*, pages 843–852.

Saiganesh Swaminathan, Kotaro Hara, and Jeffrey P Bigham. 2017. The crowd work accessibility problem. In *Proceedings of the 14th International Web for All Conference*, pages 1–4.

Maxim Tkachenko, Mikhail Malyuk, Nikita Shevchenko, Andrey Holmanyuk, and Nikolai Liubimov. 2021. Label studio: Data labeling software, 2020-2021. *Open source software available from https://github. com/heartexlabs/label-studio*.

Xingyao Wang and David Jurgens. 2021. An animated picture says at least a thousand words: Selecting gif-based replies in multimodal dialog. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3228–3257.

Max Wiechmann, Seid Muhie Yimam, and Chris Biemann. 2021. Activeanno: General-purpose document-level annotation tool with active learning integration. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Demonstrations*, pages 99–105.

Jenna Wiens, Suchi Saria, Mark Sendak, Marzyeh Ghassemi, Vincent X Liu, Finale Doshi-Velez, Kenneth Jung, Katherine Heller, David Kale, Mohammed Saeed, et al. 2019. Do no harm: a roadmap for responsible machine learning for health care. *Nature medicine*, 25(9):1337–1340.

Alex C Williams, Gloria Mark, Kristy Milland, Edward Lank, and Edith Law. 2019. The perpetual work life of crowdworkers: How tooling practices increase fragmentation in crowdwork. *Proceedings of the ACM on Human-Computer Interaction*, 3(CSCW):1–28.

Dustin Wright, Jiaxin Pei, David Jurgens, and Isabelle Augenstein. 2022. Modeling Information Change in Science Communication with Semantically Matched Paraphrases.

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.

Naitian Zhou and David Jurgens. 2020. Condolences and empathy in online communities. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 609–626.

Kathryn Zyskowski, Meredith Ringel Morris, Jeffrey P Bigham, Mary L Gray, and Shaun K Kane. 2015. Accessible crowdwork? understanding the value in and challenge of microtask employment for people with disabilities. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing*, pages 1682–1693.