The Journal of Open Source Software

Social-Annotate: Browser extension to annotate and collect social media data

Ismail Uluturk[1] **and Onur Varol***[2]

1 Department of Electrical Engineering, University of South Florida **2** Faculty of Engineering and Natural Sciences, Sabanci University

DOI: 10.21105/joss.0XXXX

Software

- Review
- Repository
- Archive

Editor: Editor Name

Submitted: 01 January 1900
Published: 01 January 3030

Summary

Human annotated data is essential in building machine learning systems and analyzing online systems. Most research efforts go through cumbersome data collection phases where they either build complicated annotation pipelines for online data or annotate offline data shared on spreadsheets or surveys. These approaches introduce significant overhead to any study requiring data collection and causes researchers to divert significant effort away from analysis and modelling to data collection, especially when they have limited programming experience. Here we present `Social-Annotate`, an extendable and configurable browser extension, to assist annotation and collection of online data. In this project we aim to develop a tool that can easily adapt to different tasks and extend to most online platforms. `Social-Annotate` allows collection of data directly on the online platform as a user sees and interacts with it as they normally would by injecting surveys into webpages on the browser, and annotations can easily be exported as files or sent to API endpoints for collaborative data collection. We aim with this tool to promote and expedite annotation of online datasets and reduce the data collection overhead so that researchers can focus on research.

Introduction

Machine learning applications rely on high-quality annotated data to build supervised models. This is especially true in social media analysis, and models trained on account behavior and content have been used to detect automated activities and to study conversations in the past (Mitra & Gilbert, 2015; Potthast, Kiesel, Reinartz, Bevendorff, & Stein, 2017; Shu, Mahudeswaran, Wang, Lee, & Liu, 2018; Varol, Ferrara, Davis, Menczer, & Flammini, 2017; Yang et al., 2019).

Researchers are often faced with inefficient tasks of inspection and labeling to build such annotated datasets. Most data collection studies, content to be inspected is often shares as a collection of URLs contained in platforms such as online forms and spreadsheets. This content is then inspected manually by following the URL directly on a web-browser, or the content is isolated and extracted from its source on a separate data collection platform. Crowd-sourcing approaches can help scale the annotation task, however the cost of high-quality annotated data is still high with crowd-sourcing since performance non-expert users requires further validation or larger samples (Hsueh, Melville, & Sindhwani, 2009; See et al., 2013). A crowd-sourcing platform can also provide annotated data in a machine-readable format to help researchers get to the analysis more conveniently.

While the data collection approaches described so far appear feasible, they often require a constant back-and-forth between the browser and forms, or display data that have been

*Corresponding author

previously extracted alongside forms on a separate platform, isolated from the original online source of the content. Both of these approaches introduce significant overhead in collection, preparation, and access of content to be annotated. Furthermore, these approaches isolate content annotation task from the natural use of the online platform which may user experience and introduce or remove biases compared to regular users. Recommendation algorithms, the sequence content is loaded, and content layout on the page are some examples of these sources of influence inherent to each platform.

Aspects of this task that can be automated or augmented to help annotators and reduce overhead. However, these automated approaches often require implementing custom data pipelines or tools. This introduces a cost of entry that excludes researchers with limited technical know-how or financial resources and makes studies of smaller scope less feasible in terms of effort or financial cost.

Here we build `Social-Annotate`, an extendable and configurable browser extension for data collection, to achieve following goals:

- Bringing online data sources and data collection platform together to speed-up the annotation process.
- Survey forms that are easy to customize for non-technical users, and extendable for other websites.
- Datasets collected for a study can be exported to JSON or CSV files, or submitted to an API endpoint automatically.
- Browser extension can inject survey question directly into websites so that annotation can be collected while interacting with the source platform naturally, or manipulate the displayed content for controlled studies.

System design

We build our browser extension following the design guidelines and best practices provided by Google Chrome Chrome extension: https://developer.chrome.com/extensions/overview. Extensions architecture mainly consists of following units:

- **manifest.json**: Provide information about the extension, functional elements of it, and necessary permissions. Required for every extension by Google Chrome. This configurations allows us to request permission to edit certain websites by injecting style and scripts.
- **Background scripts**: Handles background operations and listens for events interfacing with the extension.
- **Content scripts**: Main functional elements of the extension. Facilitates the interface between the page content, extension, and browser APIs.
- **Popup page**: A simple interface to control the extension, such as enable/disable, annotation statistics, and switching between different collection tasks.
- **Options page**: A page for configuring and customizing the extension to change the behavior of certain aspects. Configurations on this page can be exported to or imported from a file.

The Journal of Open Source Software

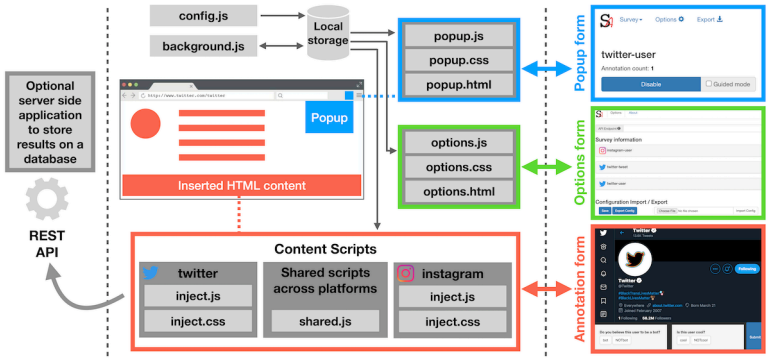

Figure 1: `Social-Annotate` schematic design. Components of the extension presented in the middle panel and their UI designs shown on the right panel.

In Figure 1, we present the high-level representation of the extension. Elements of the extension are presented in the middle panel. We implement a separate content script for each social media platform, and only inject the relevant scripts to the page. Content scripts inject the surveys to the page, where the injection location and the survey itself can be configured. Users can interact with the extension using "popup" and configure it using "options" pages.

Our extension does not require any changes to the code for customizing the survey questions. We aim to be accessible for all users, especially those with limited programming experience. Surveys can be customized in the options page, and entire configurations including customized surveys can be exported as a configuration file that can be shared with study participants for easy onboarding.

Advanced users can also implement their own content scripts for extending to different platforms. Using this additional functionality, users can design their own data collection strategy and manipulate content for controlled studies. We are providing details on how to easily extend `Social-Annotate` for supporting other platforms in section called "Extending to other platforms".

Annotation forms

Surveys can have an arbitrary number of questions of various types, according to the needs of the study. We rely on JSON schemas as templates for easy configuration and sharing custom survey forms. These surveys are then injected into the page when the URL matches with the target domain. It is possible to have different types of surveys for each platform, which will also effect which elements the surveys are injected to. For example, it is possible to annotate users or tweets on Twitter.

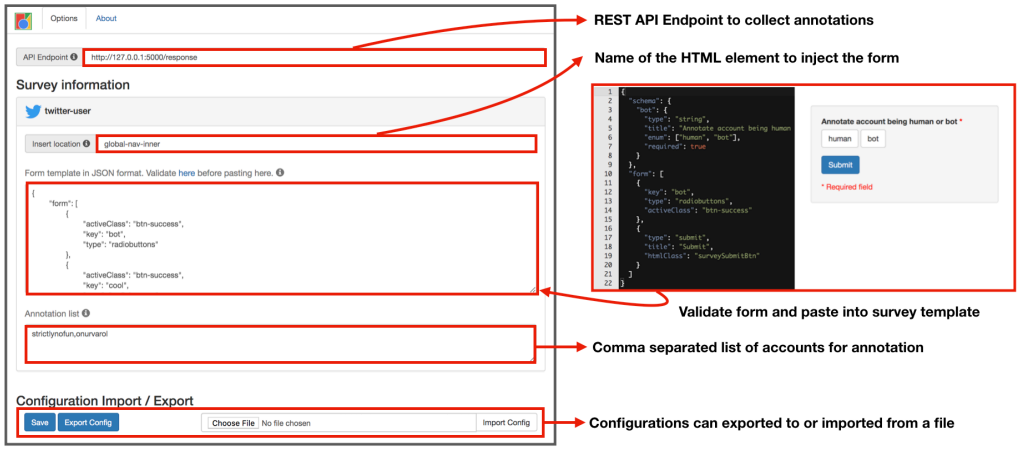

Figure 2: Example form can be input following a JSON schema format. Options page provide various controllers for configuring the annotation study.

An example form is presented in Figure 2 (right) for social bot detection research. Users can easily design their own questions and provide the JSON schema code through the options page, or import from a configuration file prepared and distributed to the participants by researchers conducting the study. Using JSON schema to define survey forms gives access to a wide range of form elements and controls without writing any HTML code, and enables researchers with limited technical know-how to create custom forms easily following provided examples (jsonform package: https://github.com/jsonform/jsonform)

Each survey form collects answers for survey questions, including but not limited to multiple choice and free text entry, as well as additional meta-data such as time it takes to answer each question and time of submission. This data is temporarily stored on the local storage of the browser, and can be exported to a file at any time. It is also possible to post each annotation to a remote server through HTTP requests.

Data collection

Collection of the survey responses are an important step prior to analysis of the outcomes. One of the common formats for analyzing such data is comma-separated files that can be opened using any spreadsheet application. Alternatively, programmatic manipulation of these results are also possible. Here we also consider exporting outcomes of the study in a JSON file where each survey response and their corresponding meta-data information are nested.

Collecting annotation results from multiple users may require gathering all the exported files and merging them. To centralize all the responses into a single location, we also provide the option to send responses to a user defined API endpoint. Having a simple server-side application listening to this endpoint, users can gather results from multiple clients. We provide a sample Flask server-side application to demonstrate this functionality in our code repository. This feature can be used to enable easier crowd-sourcing.

Extending to other platforms

`Social-Annotate` is designed with easy deployment to other platforms such as Instagram, Reddit, Facebook, etc. in mind. We demonstrate this extension process by extending `Social-Annotate` to support annotating users on Instagram. This section can be considered as

a walk-through on extending for any other platforms. We welcome and strongly encourage contributions to our open-source repository.

Social-Annotate has a central configuration file, called config.js, that contains information on extensions custom properties and details about the surveys. First step of extending this tool to other platforms is adding a new survey object into "surveys" field as shown in the highlighted code block below.

```
# config.js
var config = {
  "exportFormat": "csv",
  "activeSurveys": ["twitter-user"],
  "surveys": {
    "instagram-user":{
      "socialMediaPlatform": "instagram",
      "injectElement": {
        "name": "global-nav-inner",
        "type": "class",
        "index": 0
      },
      "surveyFormSchema" : {
        ...
      }
    },

    "twitter-user":{
      ...
    }
  }
};
```

Content scripts for the new platform should reside in its own directory under the content-scripts folder. Name of this folder should match with the key of "socialMediaPlatform" field in the config. In this particular case "instagram" should be used as folder name.

Behavior of the extension on this new platform is defined by inject.js and inject.css files, while shared/shared.js includes the shared logic and definitions. The content-script for each platform implements the Context class defined in shared.js. This requires implementing an injectSurvey method and an auxiliaryCheck method. The injectSurvey method locates the DOM element the survey will be injected into and creates a container element for the survey. The survey is rendered into a shadow tree using Shadow DOM to isolate it from the page DOM. The auxiliaryCheck method does any platform specific checks such as validating the URL for user pages, or it can return True if no checks are necessary. If there are multiple Contexts per platform, such as annotating users and tweets on Twitter, each context should be defined separately within the content-script for that platform. Finally, the platform specific content script defined under platform/inject.js loads the configuration file and renders the survey following the Context instances available and the configurations. platform/inject.css provides the style for the injected survey, isolated from the page DOM through Shadow DOM.

We now need to identify the DOM element that we want to inject the form content into on the Instagram webpage. Developer tools for modern browsers provide inspection tools to identify DOM elements and capabilities to test changes in the codes simultaneously, which can be used to determine an element that will be a suitable container for the survey. In the config.js file, we provide this information in a field called "injectElement". This field captures the matching field of the HTML div element such as "class" or "id".

After creating the content-scripts, we need to set-up storage space for the new contexts defined in the new platform specific content-scripts. The storage initialization can be found in the `background.js` file, as seen below. An empty array for each new context needs to be initialized under `resultsArrays` and `annotatedElements` objects.

```
let initialStorage = {
  "resultsArrays": {
    "twitter-user": [],
    "twitter-tweet": [],
    "instagram-user": []
  },
  "annotatedElements": {
    "twitter-user": [],
    "twitter-tweet": [],
    "instagram-user": []
  },
  "clientID": clientID,
  "config": config,
  "isEnabled": true,
  "isGuided": false,
  // clone the array below, keep the initial list for future reference.
  "activeTargetList": [...config.surveys["twitter-user"].screenNameList]
};
```

Finally, we need to update `manifest.json` declare when the new content-script files and any other files needed for this addition to support a new platform should be injected to the page. The style sheets are required to be made available as `web_accessible_resources` because the surveys are isolated from the original page DOM through Shadow DOM, and we need to access the style sheets from within the shadow tree.

```
# manifest.js
{
  "name": "Content Annotator",
  "version": "1.0",
  "manifest_version": 2,
  ... // Other configurations
  "content_scripts": [
    {
      "matches": [
        "*://twitter.com/*",
        "*://instagram.com/*"
      ],
      "css": [
        "content-scripts/instagram/inject.css",
        "content-scripts/twitter/inject.css",
        ... // Other CSS files
      ],
      "js": [
        "content-scripts/instagram/inject.js",
        "content-scripts/twitter/inject.js",
        "content-scripts/shared/shared.js",
        ... // Other JS files
      ],
      "web_accessible_resources": [
```

```
        "content-scripts/instagram/inject.css",
        "content-scripts/twitter/inject.css",
        "dep/jsonform/deps/opt/bootstrap.css"
      ],
    }
  ]
}
```

Conclusion

Researchers have already been using browser extension to collect data on user behavior (Capra, 2010), crowdsourcing tags (Kaur, JafariAsbagh, Radicchi, & Menczer, 2014), study news consumption behavior of online accounts (Chakraborty, Paranjape, Kakarla, & Ganguly, 2016), or identification of social bots (Khayat, Karimzadeh, Zhao, & Ebert, 2019). Especially in domains require datasets collected from wide range of behaviors, sampling biases for annotation can become a significant issue. Crowdsourcing the annotation task to real users can surface a more diverse set of accounts since users can label the accounts they have are observing in their own social media timelines. Tools for filtering social bot accounts are also becoming more important since platforms struggle catching up with the novel automated behaviors. Tools like BotSentinel are designed to mask problematic accounts, however such a system can be instrumented for data collection and experimentation as well. `Social-Annotate` can also be extended to insert scores for social media users or content using popular services like Botometer and Hoaxy.

We hope that by providing this tool we will reduce the overhead required for data annotation and improve the feasibility of research projects with limited resources. We believe enabling annotation directly on the social media platforms with minimal impact to the regular user experience can also make a wider range of studies possible.

Finally, although we emphasize collecting annotated social media datasets as the main focus of this tool, it is possible to extend it to be used on any content on the web. We welcome and strongly encourage contributions to our open-sourced repository.

Code repository

`Social-Annotate` is available online on our website and through Github repository listed below.

Resource	URL
Website	http://uluturki.github.io/social-annotate
Github	https://github.com/uluturki/social-annotate

References

Capra, R. (2010). HCI browser: A tool for studying web search behavior. *Proceedings of the American Society for Information Science and Technology*, *47*(1), 1–2.

Chakraborty, A., Paranjape, B., Kakarla, S., & Ganguly, N. (2016). Stop clickbait: Detecting and preventing clickbaits in online news media. In *2016 ieee/acm international conference on advances in social networks analysis and mining (asonam)* (pp. 9–16). IEEE.

Hsueh, P.-Y., Melville, P., & Sindhwani, V. (2009). Data quality from crowdsourcing: A study of annotation selection criteria. In *Proceedings of the naacl hlt 2009 workshop on active learning for natural language processing* (pp. 27–35).

Kaur, J., JafariAsbagh, M., Radicchi, F., & Menczer, F. (2014). Scholarometer: A system for crowdsourcing scholarly impact metrics. In *Proceedings of the 2014 acm conference on web science* (pp. 285–286). ACM.

Khayat, M., Karimzadeh, M., Zhao, J., & Ebert, D. S. (2019). VASSL: A visual analytics toolkit for social spambot labeling. *IEEE transactions on visualization and computer graphics*, *26*(1), 874–883.

Mitra, T., & Gilbert, E. (2015). Credbank: A large-scale social media corpus with associated credibility annotations. In *Ninth international aaai conference on web and social media*.

Potthast, M., Kiesel, J., Reinartz, K., Bevendorff, J., & Stein, B. (2017). A stylometric inquiry into hyperpartisan and fake news. *arXiv preprint arXiv:1702.05638*.

See, L., Comber, A., Salk, C., Fritz, S., Van Der Velde, M., Perger, C., Schill, C., et al. (2013). Comparing the quality of crowdsourced data contributed by expert and non-experts. *PloS one*, *8*(7), e69958.

Shu, K., Mahudeswaran, D., Wang, S., Lee, D., & Liu, H. (2018). Fakenewsnet: A data repository with news content, social context and dynamic information for studying fake news on social media. *arXiv preprint arXiv:1809.01286*.

Varol, O., Ferrara, E., Davis, C. A., Menczer, F., & Flammini, A. (2017). Online human-bot interactions: Detection, estimation, and characterization. In *Eleventh international aaai conference on web and social media*.

Yang, K.-C., Varol, O., Davis, C. A., Ferrara, E., Flammini, A., & Menczer, F. (2019). Arming the public with artificial intelligence to counter social bots. *Human Behavior and Emerging Technologies*, *1*(1), 48–61.