Artifact Evaluation README

[494] AdHere: Automated Detection and Repair of Intrusive Ads

1 About AdHere

ADHERE is a technique that can automatically detect violating ads and suggest repair proposals. The artifact contains 1) AdHere tool written in Python, 2) scripts developed for the preliminary study, and 3) the dataset used in the paper, and is publicly available at osf.io at https://osf.io/s8mhw/?view_only=42a1f52903964e68836faa76f84a180f with the file name 'AdHere_artifact.zip'.

2 Directory Structure

```
AdHere_artifact.zip
├── README.pdf /* README: How to run AdHere */
├── REQUIREMENTS.txt /* Dependencies */
├── STATUS.txt /* Badges applying for */
├── LICENSE.txt /* MIT LICENSE */
├── INSTALL.txt /* Additional installation instructions */
├── icse2023-paper494.pdf /* Accepted paper */
├── src /* Source Files */
│   ├── AdHere.py /* AdHere for Windows */
│   ├── AdHere_Linux.py /* AdHere for Linux */
│   ├── google.py /* Script for preliminary study */
│   └── *.csv, *.crx /* Dependency */
├── data /* Data used in the paper */
│   ├── adhere_partial.csv /* Violating ad samples found by AdHere */
│   ├── fix_list.csv /* Raw data of violating ad fix time */
│   ├── preliminary
│   │   └── *.csv /* Raw Google Ad Experience Report Results */
│   ├── top_1m_complete.csv /* Alexa Top 1 Million Websites list */
│   ├── manual_Google.xlsx[1] /* Google: Manual inspection result on 1k websites
│   │   */
│   └── manual_AdHere.xlsx[2] /* AdHere: Manual inspection result on 1k websites
│       */
└── fix_example /* A fix example on the mobile website 'getsongbpm.com' */
    ├── fig.png /* Figure showing the fix */
    ├── getsongbpm.com_popup_ad.html /* Ad source code before fix */
    └── getsongbpm.com_normal_ad.html /* Ad source code after fix */
```

[1]: This file contains the Google detection result on 1,000 websites (mobile FAILING 250, mobile PASSING 250, desktop FAILING 250, desktop PASSING 250). The 'Manual Positive' column lists the ground truth number of websites with violating ads in four types. Based on this we compute the accuracy/recall scores of the Google Ad Experience Report.

[2]: This file contains the AdHere detection result on 1,000 websites (mobile FAILING 250, mobile PASSING 250, desktop FAILING 250, desktop PASSING 250). And it presents detailed confusion matrix data for each violating category. In the end, it shows the accuracy/recall scores of the AdHere tool.

3 Run AdHere

3.1 AdHere

1. Install Python 3 and all dependencies (see REQUIREMENTS.txt) and your current dir should be 'src'.

2. Based on the OS and Chrome version, download the corresponding version of Chromedriver at https://chromedriver.chromium.org/. Unzip the downloaded file and put 'chromedriver.exe' in the same folder as 'AdHere.py'. Linux users: put the downloaded 'chromedriver' in the same folder as 'AdHere_Linux.py'. When the following instruction mentions 'AdHere.py', replace it with 'AdHere_Linux.py'.

3. Fill Google Chrome's user profile directory after 'USRPROFILE = ' in 'AdHere.py'.
   Possible Linux directory: ~/.config/google-chrome.
   Possible Windows directory: C:/Users/YOUR_USER_NAME/AppData/Local/Google/Chrome/User Data.

4. Run 'python AdHere.py domain_url' in the command line to run AdHere on the given URL. Leaving 'domain_url' blank will perform a self-inspection on 'google.com'. It will scan the website with the headless (no GUI) Google Chrome. After finishing the scan, AdHere will generate 'violations.txt' in the same folder as 'AdHere.py'. The text file contains violations (i.e. the id, violation type, and XPath) and their fix suggestions. Please note that 'google.com' usually won't contain ads so no violations will be found.

3.2 Preliminary Study Toolset

5. Install Python 3 and all dependencies (see REQUIREMENTS.txt) and your current dir should be 'src'.

6. Create at least one project using Google Ad Experience Report API in Google Developer Console at https://console.cloud.google.com/apis/dashboard. If you want to accelerate the execution, create multiple projects and apply for the API key for each project.

7. Fill API key(s) in 'API_KEY_LIST' in 'google.py'. The program won't run without an API key. The API key should replace the empty string placeholder.
   Adjust 'THREAD_COUNT' based on the comments in 'google.py'. If you only get one API key, set the COUNT to 1. If you want to perform a small-scale test, please change the value of 'TARGET_LIST' from 'top-1m' to 'top-10' to quickly scan the top 10 websites only.

8. Run 'python google.py' in the terminal to get Google Ad Experience Report's result on the Alexa Top 1 Million Websites. With 1 API key, the process on 1 million websites can last for more than one day. Please try to keep the network connection stable.

9. After the program exits normally, the script will generate files in its folder. '[R]Alexa_done [XX_XX].csv' is the raw file that records the compliance status of the 1 million websites.