2024 Project Timeline: Sprint Reviews, Retrospectives, and Planning

Week 3 (January 15-19, 2024):
- Event: Sprint 2 Review, Retrospective, and Sprint 3 Planning.

Week 5 (January 29 - February 2, 2024):
- Event: Sprint 3 Review, Retrospective, and Sprint 4 Planning.

Week 7 (February 12-16, 2024):
- Event: Sprint 4 Review, Retrospective, and Sprint 5 Planning.

Week 9 (February 26 - March 1, 2024):
- Event: Sprint 5 Review, Retrospective, and Sprint 6 Planning.

Week 10 (March 4-7, 2024):
- Event: Second Project Review.

Week 11 (March 11-15, 2024):
- Event: Sprint 6 Review, Retrospective, and Sprint 7 Planning.

Week 13 (March 25-29, 2024):
- Event: Sprint 7 Review, Retrospective, and Sprint 8 Planning.

Week 15 (April 8-12, 2024):
- Event: Sprint 7 Review, Retrospective, and Sprint 8 Planning (Final Sprint).

Week 17 (April 22-25, 2024):
- Event: Final Project Review.

Team 10 - Public materials – Google Drive

https://drive.google.com/drive/folders/1B4ScPnNRhTT8z66bg-mTBlwe22kNZpX2

Topic Results

https://mycourses.aalto.fi/course/view.php?id=39356&section=6

Task 1: Analysis of the Austrian Paper

Objective: Study specific aspects of the Austrian paper on fake webshop detection.

Sub-Tasks:

Feature Extraction Analysis:
- Understand how they conduct feature extraction.
- Identify what features they use.

Approach to Problem Solving:
- Examine their method for detecting fake webshops.

Experimental Setup Summary:
- Summarize their experimental setup, including the number of malicious vs. safe sites.

Result Analysis:
- Analyze and understand their claimed results.

Resources:

- Austrian Paper GitHub Repository: mal2-project/fake-shop-detection_models.

Note: Focus on reading and understanding the code and paper; running the training script is not required.

Task 2: Presentation on the Austrian Paper

Objective: Create a detailed presentation based on the Austrian paper.

Sub-Tasks:

Extracting Code Segments:
- Use parts of their code to explain feature extraction from HTML files.
- Discuss the process of transforming `.html` files into features.

Machine Learning Models:

- Explain why the authors used Random Forest and XGBoost models.
- Discuss the importance of features in these models.

Technical Aspects:

- Elaborate on the use of n-grams and TF-IDF vectorizer in their approach.

Target Audience: Khalid, Nina, Sean, and Binh.

Goal: Ensure the presentation is straightforward enough for Binh and Nina to understand. If not, consider simplifying the content.

Task 3: Implementation and Presentation Preparation

Objective: Prepare a presentation on data sourcing and processing, and develop a script for feature extraction from HTML.

Sub-Tasks:

Data Sourcing and Processing Presentation:

- Present where and how the authors of the paper sourced their data.
- Explain their data processing methods.

Script Development for Feature Extraction:

- Develop a Python script to extract tokens from HTML.
- Focus on raw HTML extraction, using Regex for text removal.
- Explore lemmatization and punctuation removal, comparing it with the Austrian paper's approach.

**Technical Tools and Concepts
:**

- Utilize tools such as Beautiful Soup and Scikit-learn.
- Create a TF-IDF vectorizer and implement n-gram (bigram) analysis.
- Ensure modularity, analysability, and modifiability in your script for maintainability.

Resources:

- Austrian Paper and corresponding GitHub repository.
- Sample websites for testing the feature extraction script.

Additional Notes:

- Aim to replicate and validate key components from the paper.
- Reuse the Austrian code as much as possible.
- Initially test on a Jupyter notebook, with later conversion to a `.py` script.
- Ensure the content and script are accessible to team members, especially Binh and Nina.

Meetings and Reviews:

- Schedule votes for available days for planning.
- Prepare for the Sprint Review, with presentations by Binh and Nina to the team.
- Align with the sprint timeline (Sprint 2 Review, Retro + Sprint 3 Planning, etc.).

quick summary of meeting 16/01/2024: I showed code just that it saves files and then I opened one html file. Suggestions:
1 to be ready to tell the team what exactly we want from them, what format, what data
2 figure out what happens after raw html is sent to tf-idf, make some statistic how many times some tag was there, what you do with html
3 cannot copy Austrian code, have to rewrite

Here are the points Khalid mentioned during the meeting on 01/02/2024:

- Be consistent with scraping, as some would provide more stuff than others. I'd suggest sticking with playwright for scraping data for analysis/ml
- Before our meetings & the review, always think about how you're going to present your work to us. Make the flow smooth and understandable, and focus on the important work that has been done. If time allows, we can dig deeper into the details
- When doing feature extraction, point out 1) what features were studied in the literature, 2) which features are implemented/provided in the extractor, 3) did you use use them as is or did you extend them, 4) if you implemented a totally new function for feature extraction, mention that and 4) suggest if the feature extraction function could be useful for all types of classifications (i.e., a generic one that might be useful for detection of phishing, malicious, adult, ... sites) or just a specific one (e.g., fake shop detection in this context)
- When using AI for code generation, indicate that in the docstring of the function or as a comment in the code. Make sure that we can easily distinguish between code that is written by you and code that's generated. It would also be useful to document the process used for generating them (e.g., context provided, prompt and such); however, this step isn't very very important.
- Like in the thesis, you can also add a third type of websites that's for legitimate generic websites (e.g., a small subset of the top 1m domains that aren't shops)

Sean said, Fantastic points from Khalid

We think this is now going a lot better and currently the area for most improvement is presenting your work which will make our meetings much more useful.

When presenting: give an overall summary of what your aim was, where you got to and then more detailed discussion on what you think are any important specific points.

Let's say the aim is working towards matching the statistics/graphs in the thesis: The overall summary would be saying how many features there are in the thesis, how many we can currently obtain, whether you were able to match the thesis statistics/graphs or not etc. A table to summarize this seems like it would be most appropriate here.

Then for more detailed discussion you could take one or two examples of each case matching or not to show the output and you can say either "yes, our statistics/graphs matched the thesis and the explanation for why you expect this feature to look like this for fake/legit shops is _____". For the ones that don't match: "we think there is something ambiguous in how this feature has been defined"/"we think there is something wrong in the thesis"/"we are not sure and this is something to follow up with".

https://scikit-learn.org/stable/auto_examples/ensemble/plot_forest_importances.html

https://scikit-learn.org/stable/auto_examples/ensemble/plot_forest_importances.html

1. I would incorporate Nina's features into text tf-idf.

2. Imputing missing values, or put it onto other side

3. Plot the feature importance from the Random Forest Classifier

Exam week: 19-25th of February

Random Forest Classifier

First, we have a non-text features dataframe (from Nina)
Second

train-test split of 70-15-15

display keywork in tf-idf in feature importance instead of index number

Displaying the label as the confidence of 0/1 (like gptzero)

========
Notes 22.02.2024

Hi, you can use either of pickle or joblib, I just know joblib is more optimized but I encountered some compatibility issues. Regardless of which one you'll use, make sure you use the latest version and document the version number.

For the process, I expect the classification code and the extraction of fake-shop features to be implemented in here:

https://github.com/F-Secure/Aalto2023project/tree/main/library/web_classifier/classifications

(not in a notebook). You can create a model directory at the root folder:
https://github.com/F-Secure/Aalto2023project/tree/main

where you host the model (given that it is small in size :smile:)

The format can follow the one here
https://github.com/F-Secure/Aalto2023project/blob/main/library/web_classifier/classifications/classifier.py

The score would be the confidence score.
Implement the abstract class from the classifier, integration of the model, not the quality yet, using the 90s dataset. Focus on one classifier, from non-text features of Nina.

Choose the 12 important features Nina presented, make a classifier for it and document the command to run the ML model.

There are no option flags. Follow the template from Khalid's.

Try to install the code from Tuomo to actually run the scraper.

Nina
I am working on integration of ML and API, and when I am done with it, I will need your help to continue on it. I will make colab about this - save model, then load model, take new url, extract all features from it into X_new_url, run forest.predict(X_new_url), print forest.predict_proba(X_new_url) and save into json. You DON'T need to do this yet, I plan to work on it tomorrow.
You need to save feature extraction and model training into github. I won't be doing that.

Nina
also a reminder - project review is 5.03 14.10-15.00

actually no, I won't be working on ML-API tomorrow. I need to figure out differences between Kaarlo's features and my own. He has added some that I have too (semantic similarity of body, edit distance of title, numerical strings, currencies and unique currencies), but I haven't looked through all of them. I'd say for purposes of initial integration you can go with any features you want, then it can be edited. For example number of mail and tel links - use Kaarlo's (I am also using them in that colab I sent in extraction-ml channel). 'nb_imgs', 'nb_meta_desc', 'nb_meta_keyw' - doesn't matter whose you take, they are same.

The scrapers and extractor mostly work, there are still some edge cases that might break. Take a look at https://github.com/F-Secure/Aalto2023project/tree/main/library#readme for instructions on running the cli commands and all the options. I'll be at work quite late today, but I am reachable by text and tomorrow after mid day I'll be on my pc so can help more if needed.

Xuan Binh
I understood. So after 4pm today, when will be the latest hour you will be offline? I always work until 10 PM

Kaarlo Kauriinvaha
I'll probably be reachable all evening by text on slack, but I'll be at work so I can't fix / test anything. But if there is anything unclear that needs some more explanation, I'll be able answer as best as I can

Yes naturally, I don't ask you to fix or code, I only ask for your help about running the code if I'm stuck somewhere :cry: Thank you so much. Then see you soon. Sorry to make you and Nina worried...

Kaarlo Kauriinvaha
There is some work I did in https://github.com/F-Secure/Aalto2023project/blob/feature/FWD-101-extraction/library/web_classifier/classifications/webshop.py on setting up the class for the classifier. There are some of the extractions, but some that Nina made might need to be adjusted / added. Also there is no CLI interface for the classifier yet, I have been testing that with the api and with unit tests :eyes:

Kaarlo Kauriinvaha
Something I'll note is that all the work here https://github.com/F-Secure/Aalto2023project/blob/feature/FWD-101-extraction/library/web_classifier/classifications/webshop.py is not set in stone, so please modify anything if you feel the need:+1:

git fetch
git merge origin main
$ scrape --scraper curl -i tests/fixtures/test_urls.txt -o test_scrape.jsonl

$ extract -i test_scrape.jsonl -o test_extraction.jsonl

Note 29/02/2024

Load the model in the

```
  def __init__(self, *args, **kwargs) -> None:
      super().__init__(*args, **kwargs)
```

Also I just pushed a commit to feature/FWD-101-extraction with a CLI for interacting with the classifier.
The usage is as follows:
PYTHONPATH=. python webclassifier/cli/classify.py -i test_extraction.jsonl -o test_classification.jsonl
3:13
Where test_extraction.jsonl is of course the inputfile with extraction results and test_classification.jsonl contains the results that WebshopClassifier.predict() returns

Focus on the predict() function of the Webshop Classifier
Looking at PhishingClassifier predict() function

At the moment, just pick 2 - 3 features from Nina's model.

Freeze the features from Nina's notebooks

hint: You have divergent branches and need to specify how to reconcile them.
hint: You can do so by running one of the following commands sometime before
hint: your next pull:
hint:
hint: git config pull.rebase false # merge (the default strategy)
hint: git config pull.rebase true   # rebase
hint: git config pull.ff only   # fast-forward only
hint:
hint: You can replace "git config" with "git config --global" to set a default
hint: preference for all repositories. You can also pass --rebase, --no-rebase,
hint: or --ff-only on the command line to override the configured default per
hint: invocation.

To solve this error run git config pull.rebase false

run git commit before we run git fetch git merge origin <branch>

Also I just pushed a commit to feature/FWD-101-extraction with a CLI for interacting with the classifier.
The usage is as follows:
PYTHONPATH=. python web_classifier/cli/classify.py -i test_extraction.jsonl -o test_classification.jsonl

runs in the library folder

Where test_extraction.jsonl is ofcourse the inputfile with extraction results and test_classification.jsonl contains the results that WebshopClassifier.predict() returns

The cli commands is in cli/classify.py

Notes 07/03/2024

Kaarlo said he is unavailable until friday night/ saturday as he is travelling for work
10:20
you can ask Tuomo, he was running it on review on Tuesday
10:21
I sent TO DO list in chat with POs but I can send it here too
10:21
Sean said number 6 is most important

TO DO:

1. handle the issue with one hot encoding for Registrar feature. Registrar was one of the most important features but I had to drop it temporarily because test data had only 1 registrar and training data has around 30 Registrars, and number of columns was different. I didn't explore ways to handle this issue so I just dropped the feature until somebody fixes that. Model is ok without it, but might be better with it, or maybe not.

1.1 Also there is an issue that when I dropped one of features that are highly correlated with each other (like N of iframes and iframe tags or something like that that had correlation 0,99 or 1,00, I don't remember now), or an insignificant feature that has 0,03 correlation (N of all urls) with Legitimacy - model's plot of features in permutation plot became very small, with like 2-3 features, usually domain age first, and then some other. I dropped for example number of li tags (and some other ones) which has high correlation with several other features, but that resulted in more misclassifications than normally, so I put them back. So _model_plot_ (not model itself) instability in terms of dropping and adding features is something to look into. version of 28.2 has around 15 features in permutation plot, but 1.03 version committed (without

registrar) has only 2 features in that plot (domain age and lexical diversity), but same accuracy though, only 1 misclassified site, and on this dataset there was always 1 misclassified site. Model 3.03 has domain age and GoogleAnalytics as the only features on the plot, but accuracy is the same, so I came to conclusion that this plot with features based on permutation shouldn't be taken as the truth on which to base decisions whether to drop or take features, only confusion matrix is the only reliable sign. So this is not necessarily a problem, just something to keep in mind and maybe look into.

2. when new data is available, look how model performs, are there any more invalid and incomplete values that need to be handled. Now missing domain age is imputed with zero, maybe not the best value (median for fakes is aroung 124 if I remember correctly, for legit several thousand). Code has to handle all missing data. It does handle quite a lot of these issues but there probably are more that are not accounted for. All kinds of input should be tested. Also might be good to check whether site is shop or not before using it in training, and 28.02, 1.03, 3.03 models didn't exclude non-shops. Here is code that checks shop or not https://colab.research.google.com/drive/10B4OoRNqB6vx9ciuVG5bXMCbBKgg5nOm#scrollTo=sUS55GhFTpWy.

3. when there is not enough data, model interprets zeros or lack of values as ok, while it should state that there is not enough data to classify, rather than predict based on few features. For example some sites have null in html_extraction partly due to scraping issues, so they have only registrar and SSL info, and they are classified just based on that, which might not be particularly meaningful, especially if domain age is also missing. Model has to inform user that there is not enough data to classify. maybe if it's a good idea to have this information in web UI - make a very shortened version of extractor output with html_extractor data and SSL and registrar data or some version of dataframe so that TPT can view what values the features have, so that they can understand on what features the classification is based and if some values are missing, so if html_extraction is null, they will see it straight in web UI. I don't know if it's worthwhile, just an idea.

4. SSL certificate issuer is not used in one-hot encoding because it resulted in misclassification on 90 sites data, but maybe on new data it will be good for accuracy.

5. make plots on new data. Here is a colab with code to do plots, they are in the end: https://colab.research.google.com/drive/1H7h5wd1jF29oDcJWJVGT_CzTsTJNKEVO

6. code review and editing, documentation, code explanation, making sure code is maintainable, modular, analysable, modifiable as per Definition of Done.

7. consider new features, such as href="\javascript:" and onclick etc, comments, but that's probably the least important thing on this list. If TF-IDF introduces _any_ extra misclassification, then it shouldn't be included in the model, in my opinion.

There is enough work on the ML part without TF-IDF. It might be good to push it to github for future purposes if someone ever wants to use that but for this model probably better to not introduce any more features. Maybe TF_IDF might slow down the process of predicting, I don't know. There were features that used imported nltk and spaCy, and they made training run for 65-75 seconds, while without them it was running for 3-7 seconds. I know time of training is not important, but time of predicting might also be affected by this. Response time of whole pipeline should be as fast as it can, and if some feature introduces say 2 second lag, it shouldn't be included, in my opinion.

#####
What Binh can do?
PYTHONPATH=. python web_classifier/cli/classify.py -i test_extraction.jsonl -o test_classification.jsonl
Task 1:
Task 2:
Task 3:
Task 4:
The CLI command
The whole needs to be run locally.

cd library
source .venv/bin/activate

PYTHONPATH=. python web_classifier/cli/classify.py -i test_extraction.jsonl -o test_classification.jsonl

=====

Putting together, summarizing what was done for this ML component in this project
Overview, technology,
What are the features used by the
Classify the website as fake or shop
Obtain the scoring component
Documentation of what has been done? What can be summarized and presented to the team.
Hey want was done here to the ML, we did this. That's the priority of presentation. Accuracy or whatever is not a problem.

Showing the documentation and hey these are the things we have done, like which branches are merged?

==========

Question 1: May I know if Kaarlo or on your side have a larger dataset prepared?

Yes, Kaarlo has a new dataset.
There is a link to the dataset since it is very heavy and cannot be committed to git

Question 2: I would document everything as it is that Nina has done, no more and no less. If you want me to elaborate on something, I hope to hear your advice

Answer is Yes

Actually, Binh needs to ask Nina again what is the most recently updated file?
Question 3: Now I am clear of the workflow. Would you allow me to develop a classification model on my own?: Yes, but unless binh finished the docs first.

Write an overview of what was actually done for the ML component.

What has been done: most important.
Level of details: if there is another student, and we need to present to new audience, how could we even present the topic and the component to that audience?

300 fakes, 600
accuracy, lots of false negatives

Question 4: Finally, I believe I have not installed the project code yet. If you have indeed installed correctly, would you be kind enough to send me the command of running the ML CLI or API command so I can test the code?

For the 3rd Project Meeting: at the end of April (23rd -24th): Make 2-3 slides on just the ML component.

Create labels for safe and fake as the strings safe and fake itself, do not use 0 and 1

In meetings tomorrow and wednesday, show Mr Sean what I have done and planned, DO NOT CODE DURING MEETING.

What Mr.Sean actually wants: describe the datasets, tell how many fake and safe websites, how many URLs, What features have NaN imputation.