RF Classifying Legal and Illegal Anime Streaming

Vinson Lin, Joshua Zhou

Abstract

The allure of illegal pirating websites encourages an unprecedented exploration into the economy of the anime industry, particularly based on the rise of online streaming. The study tackles the significant challenge of distinguishing between legal and illegal anime content while also delving into the prediction of core network features related to streaming preferences. This dilemma holds profound implications for the global anime community, grappling with overworked animation teams, voice actors, and Japanese anime studios facing tight budgets and demanding production quotas. The paper demonstrates inherent variability in content availability across platforms as a driving force for illegal streaming alternatives. By shedding light on the prevalence of unauthorized streaming through data from pirating sites, Reddit discussions, and platform shutdowns, the research aims to contribute insights for a more sustainable future for the anime industry.

1 Introduction

The anime industry faces a unique dilemma exacerbated by the growing prevalence of online streaming. Our research endeavors to address the challenge of distinguishing between legal and illegal anime content, as well as predicting core network features in streaming preferences. This issue holds profound significance, as the global anime community grapples with overworked animation teams, voice actors, and Japanese anime studios contending with tight budgets and demanding production quotas. While a solution could simply be increased subscriptions to major streaming platforms like Crunchyroll, Hulu, and Funimation, which could potentially alleviate the burdens on industry stakeholders, the inherent variability in content availability across platforms poses an ironic drawback: many resort to illegal streaming alternatives due to platform-exclusivity of certain anime. This research aims to shed light on the prevalence of unauthorized streaming, evidenced by data from popular pirating sites, discussions within Reddit communities, and the occasional shutdown of such illicit platforms. By understanding the dynamics of legal and illegal anime consumption, we hope to contribute insights that pave the way for a more sustainable and supportive future for the anime industry.

In our approach to this networking classifier problem, we have decided to analyze the efficacy in labeling a distribution of packet captures collected from two sources: crunchyroll.com and animepahe.ru. It is worth mentioning that there are elaborate uses and benefits for both legal and illegal parties of anime streaming. For instance, the average legal consumer may decide they can no longer afford monthly subscriptions to a platform that fails to stream their favorite and upcoming/releasing anime. They can inform themselves on our research findings and decide whether to cancel the subscription in favor of a potentially similar- perhaps, better streaming experience. On the other hand, big Japanese-affiliated companies may be interested in analyzing the performances of sites hosting illegal content so that they can re-adjust or adapt their marketing and domain strategies to eliminate any drawbacks in important network features and consumer tendencies.

Our paper shows that certain features are able to classify network conditions as either legal or illegal content. A focal point of our research lies a critical exploration of the pivotal role of data collection in understanding consumer behavior within the realm of anime streaming, with a particular emphasis on watch time and the appeal of ad-free content. Significantly, our investigation unveils the stark contrast in experiences between legal and illegal anime streaming, as a substantial number of illicit viewers heavily rely on ad-blockers, shaping their interactions with the content. Notably, by delving into the granularity of watch time and strategically positioning advertisements at key intervals, such as the commencement of a 15-second ad followed by subsequent ad clusters around the 2-minute mark, we aim to capture the intricacies of user engagement. This deliberate approach to data extraction ensures that our packet and network feature data encapsulate the genuine conditions encountered by viewers, enhancing the practical relevance of our findings. By pioneering this nuanced exploration, our paper contributes a comprehensive understanding of streaming dynamics, shedding light on the pivotal role of targeted data collection in uncovering core network features. This nuanced perspective not only enriches scholarly discourse but also offers actionable insights for industry stakeholders seeking to navigate the evolving landscape of legal and illegal anime consumption, separated by an increasingly ineffective paywall.

The novelty of our work stems from the distinctive approach we undertake in addressing the complexities of legal and illegal anime streaming. To our knowledge, no prior publications have explored the intricacies of watch time, ad-free content, and user engagement in the context of distinguishing between legal and illegal streaming. Building upon this novel foundation, the remainder of this paper unfolds as follows. In Section 2, we delve into the background and motivation, introducing innovative tools such as NetUnicorn and Trustee, both representing black-box machine learning methodologies. Section 3 navigates through our design, approach, and methodology, expounding on the ad-less content paradigm and our strategic mimicry of paywall-hidden content. It further details our selection of features, including 'Total Fwd Packet,' 'Flow Duration,' and others, crucial to capturing the nuances of user behavior. Section 4 outlines the specific implementation using Python (Jupyter) notebooks, tailoring black-box functions to selectively capture flows and circumvent ad windows during watch time. Notably, this section explores our efforts in adapting these tools to our unique requirements. In Section 5, we present a comprehensive summary of the questions we aim to answer, offering a preview of key results. The evaluation sub-section elucidates our setup, incorporating dataset descriptions, testbed details, and insights into the efficacy of our adapted random forest model, even when faced with challenges such as limited data and time constraints.

2 Background and Motivation

We primarily focused on the machine learning classification problem where we train a model to distinguish different categories based on the given information. Our model was trained to distinguish between two different anime streaming sites based of packet capture data.

The data was collected using black box functions from the NetUnicorn package. However, there were some design limitations to this. It was difficult collecting a sufficient amount of data within the time frame. Furthermore, NetUnicorn watch video functions do not come with ad blocker.

Our model was trained using the random forest classifier class from scikit.

We followed the traditional approaches to the networking classification problem. Which is to collect relevant features, train the model, and then analyze whether or not the model effectively learned from the given features. Then we retrained the model if necessary. However, we have expanded on this procedure by applying it to data collected from "illegal" sites.

3 Design/Approach/Methodology

In crafting our system design, we carefully navigated through critical decisions to ensure a comprehensive and replicable approach. Notably, we maintained consistency in the "skip ad" 15-second packet manipulation and anime episode selection for data collection from Crunchyroll-free episodes and Animepahe. This strategic choice was driven by the need to assess the predictability of legal streaming websites in displaying ads within the initial 15 seconds of content. While it remains plausible to enhance Selenium or Chromium tasks with an adblocker, we confronted constraints in time and technical complexity.

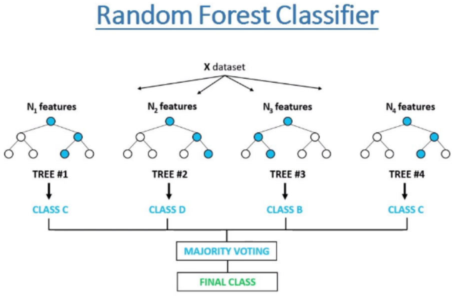
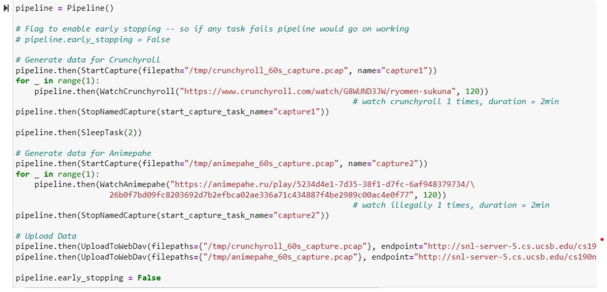


Figure 1: *Random Forest*

Consequently, our pipeline is limited to single iterations, acknowledging the necessity to streamline each and every csv data pre-processing phase. This involved discarding the first 15 seconds' worth of timestamps from each CSV, a process that, with more resources, could be refined through multiple iterations. To alleviate this limitation, we propose the integration of a robust sleep command within the watchCrunchyroll/Animepahe tasks, allowing for a delay in data capture during the initial ad window. This deliberate design choice not only streamlines our methodology but also opens avenues for future refinements and adaptations to accommodate the dynamic nature of online streaming platforms.

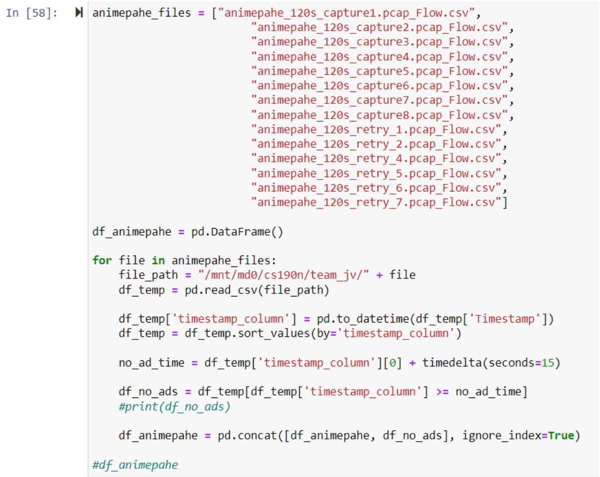
4 Implementation

Data Collection:
Specific implementation of the pipeline process can be see here:



```
pipeline = Pipeline()

# Flag to enable early stopping -- so if any task fails pipeline would go on working
# pipeline.early_stopping = False

# Generate data for Crunchyroll
pipeline.then(StartCapture(filepath="/tmp/crunchyroll_60s_capture.pcap", name="capture1"))
for _ in range(1):
    pipeline.then(WatchCrunchyroll("https://www.crunchyroll.com/watch/G8WUND3JW/ryomen-sukuna", 120))
                                                                    # watch crunchyroll 1 times, duration = 2min
pipeline.then(StopNamedCapture(start_capture_task_name="capture1"))

pipeline.then(SleepTask(2))

# Generate data for Animepahe
pipeline.then(StartCapture(filepath="/tmp/animepahe_60s_capture.pcap", name="capture2"))
for _ in range(1):
    pipeline.then(WatchAnimepahe("https://animepahe.ru/play/5234d4e1-7d35-38f1-d7fc-6af948379734/\
                        26b0f7bd09fc8203692d7b2efbca02ae336a71c434887f4be2989c00ac4e0f77", 120))
                                                                    # watch illegally 1 times, duration = 2min
pipeline.then(StopNamedCapture(start_capture_task_name="capture2"))

# Upload Data
pipeline.then(UploadToWebDav(filepaths={"/tmp/crunchyroll_60s_capture.pcap"}, endpoint="http://snl-server-5.cs.ucsb.edu/cs19
pipeline.then(UploadToWebDav(filepaths={"/tmp/animepahe_60s_capture.pcap"}, endpoint="http://snl-server-5.cs.ucsb.edu/cs190n

pipeline.early_stopping = False
```

This implementation ensures that the proper data will be collected with the understanding that some random noise will be included (which can easily be filtered later based on flows with less than 30 forward or backward packets). After fine-tuning this process and reaching our specified implementation for ad-less flows, we were able



```
In [58]:   animepahe_files = ["animepahe_120s_capture1.pcap_Flow.csv",
                              "animepahe_120s_capture2.pcap_Flow.csv",
                              "animepahe_120s_capture3.pcap_Flow.csv",
                              "animepahe_120s_capture4.pcap_Flow.csv",
                              "animepahe_120s_capture5.pcap_Flow.csv",
                              "animepahe_120s_capture6.pcap_Flow.csv",
                              "animepahe_120s_capture7.pcap_Flow.csv",
                              "animepahe_120s_capture8.pcap_Flow.csv",
                              "animepahe_120s_retry_1.pcap_Flow.csv",
                              "animepahe_120s_retry_2.pcap_Flow.csv",
                              "animepahe_120s_retry_4.pcap_Flow.csv",
                              "animepahe_120s_retry_5.pcap_Flow.csv",
                              "animepahe_120s_retry_6.pcap_Flow.csv",
                              "animepahe_120s_retry_7.pcap_Flow.csv"]

           df_animepahe = pd.DataFrame()

           for file in animepahe_files:
               file_path = "/mnt/md0/cs190n/team_jv/" + file
               df_temp = pd.read_csv(file_path)

               df_temp['timestamp_column'] = pd.to_datetime(df_temp['Timestamp'])
               df_temp = df_temp.sort_values(by='timestamp_column')

               no_ad_time = df_temp['timestamp_column'][0] + timedelta(seconds=15)

               df_no_ads = df_temp[df_temp['timestamp_column'] >= no_ad_time]
               #print(df_no_ads)

               df_animepahe = pd.concat([df_animepahe, df_no_ads], ignore_index=True)

           #df_animepahe
```

Figure 2: *Animepahe dataframe*

to upscale data collection up to 9 raspberry-pi nodes. Watching 1x120seconds averages around 50 flows for animepahe, and 80 flows for crunchyroll.
Following this procedure, we used given blackbox docker image containers to mediate the process of converting pcap (packet capture) data into csv's with specified filenames. Our implementation however, is quite lacking in terms of data collection- resulted in only 149 rows of data after cleaning. Instead of row manipulation for each of the csv's, we could have implemented the watch tasks to include an argument for sleeping a certain amount of seconds in the beginning in order to filter specified timestamps of streaming flows all in the pipeline process.
Feature Extraction:

In order for the model to learn effectively it was important to only train the model on feature which were important for the learning problem.

- First we converged all pcap csvs into a two dataframes, one for crunchyroll and another for animepahe.
- Took the initial timestamp of all video watches and removed all flows captured withing the period then until the next fifteen seconds in order to remove the initial ad from our packet capture data. We felt like this was the most effective way in solving this issue, given the time constraint.
- Removed flows that did not have a total packet count higher than thirty. This significantly reduced the noise within our data. It also allows for higher quality information for which the model can use.
- Added a column to each dataframe called 'Label' which signifies the site the flow came from. Labels are used for the classifier and were named 'crunchyroll' and 'animepahe'.
- Combined the two dataframes into a single table.
- Dropped irrelevant features such as source IP, destination port, timestamp, etc. because they either caused shortcuts within our model, or they were unnecessary.

Model Training:

The model was trained using the RandomForestClassifier() from scikit.

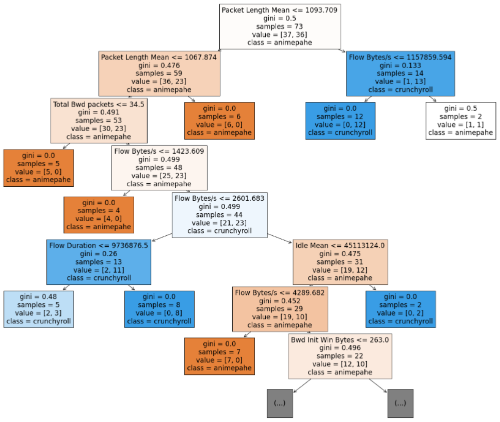
- The training features was set to all columns in the final dataset excluding 'Label'. The target feature was 'Label'

Model Analysis:

We used trustee in order to analyze our model.Trustee is a framework that allows users to view the actions of black-box machine learning models.

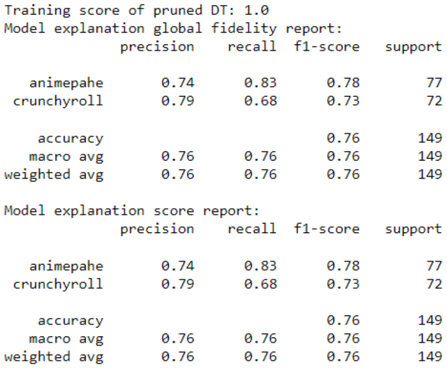
5 Evaluation

We aim to answer the questions of which network features were instrumental in classifying between legal and illegal network connections based on the depth of the decision trees that are produced. Furthermore we want to find out the accuracy of our model and how effective it was at learning the classifying based on the given data. In doing so, we can evaluate our model's performance scores- taken with a grain of salt due to our low data volume. A low Gini index is desirable as it indicates a more effective split in the decision tree, leading to subsets that are more uniform with respect to the target variable. According to our trustee diagram, the blue features effectively split and lead to leaf nodes- thus, have a low gini score. We can conclude that **Flow Bytes** and **Flow Duration** are significant network features that effectively classify crunchyroll and animepahe streaming content based on our relatively low data size.



A low Gini index is desirable as it indicates a more effective split in the decision tree, leading to subsets that are more uniform with respect to the target variable. According to our trustee diagram, the blue features effectively split and lead to leaf nodes- thus, have a low gini score. We can conclude that **Flow Bytes** and **Flow Duration** are significant network features that effectively classify

crunchyroll and animepahe streaming content based on our relatively low data size.



```
Training score of pruned DT: 1.0
Model explanation global fidelity report:
              precision    recall  f1-score   support

   animepahe       0.74      0.83      0.78        77
 crunchyroll       0.79      0.68      0.73        72

    accuracy                           0.76       149
   macro avg       0.76      0.76      0.76       149
weighted avg       0.76      0.76      0.76       149

Model explanation score report:
              precision    recall  f1-score   support

   animepahe       0.74      0.83      0.78        77
 crunchyroll       0.79      0.68      0.73        72

    accuracy                           0.76       149
   macro avg       0.76      0.76      0.76       149
weighted avg       0.76      0.76      0.76       149
```

This graph shows many useful metrics such as Precision, Recall, F1-Score. However if we look at accuracy in particular it can give us a good idea how the model is performing. With an average accuracy of 0.76 the model performs well in most instances and shows that the features seems to provide sufficient discriminatory information for the classification task. However, significant improvements can be made to increase the results in this table. We only used 150 data points so increasing this number by orders of magnitude would greatly benefit the model.

6 Conclusion

In conclusion, our research has provided valuable insights into the intricate dynamics of legal and illegal anime streaming, addressing the challenge of distinguishing between the two and predicting core network features in streaming preferences. By focusing on watch time and the allure of ad-free content, we have highlighted the stark contrast in experiences between legal and illegal anime consumption. Our approach, utilizing innovative tools such as NetUnicorn and Trustee, allows a nuanced exploration that enriches scholarly discourse and offers actionable insights for industry stakeholders. The efficacy of our networking classifier is demonstrated by the identification of significant features such as Flow Bytes and Flow Duration, showcasing the model's ability to classify streaming content effectively. However, the limitations of our approach, including the constrained data volume and the need for further refinement, are acknowledged. Moving forward, future work could involve expanding the dataset size, implementing ad-blocking mechanisms, and refining the feature selection process to enhance the model's accuracy and robustness. Additionally, exploring a Crunchyroll premium dataset would be highly recommended as we could assume their are difference is quality of experience resistricted behind paywall and authentication versus the free episodes we selected. In essence, our research contributes a comprehensive understanding of streaming dynamics and sets the stage for continued advancements in addressing the challenges faced by the anime industry in the realm of online streaming.

References

1. [How to Use the Tree-Based Algorithm for Machine Learning](https://www.freecodecamp.org/news/how-to-use-the-tree-based-algorithm-for-machine-learning/)