Non-Fungible Token Fraud Detection System

Chan Tsz Hei 3035692060
Choi Yik Ho 3035684415

January 22, 2023
Abstract

While non-fungible tokens (NFT) have been gaining in popularity, fraud has also been on the rise. Artwork theft and artist impersonation have all become more common as people look to purchase and sell digital artwork using this blockchain-based technology. At present, there are no established fraud detection systems on popular NFT marketplaces to mitigate the impacts of various frauds and scams. Our project aims to use machine learning to better classify NFTs and detect which tokens may be suspicious based on their collection metadata and activities taking place on the blockchain. In partnership with Dotted Company Limited (DTTD) a local start-up that is developing a mobile-first NFT social platform, we are creating a system that will be integrated into their platform to provide real-time classifications for NFTs in their mobile app.

We have completed the data collection and data pipeline, and have successfully trained and deployed a preliminary classifier. In the future, we will continue to refine the model and the system pipeline to improve the accuracy of the model.
Acknowledgements

We would like to express our gratitude to our project supervisor, Dr. John Yuen, for his guidance and support throughout the project. We are also grateful to be working with DTTD for offering this invaluable partnership opportunity and for providing us with the necessary technical infrastructure and data which makes this project possible.
Contents

1 Introduction 5
   1.1 Background ....................................................... 5
   1.2 Problem Statement .............................................. 5
   1.3 Objectives ....................................................... 6
   1.4 Related Work ................................................... 6
   1.5 Scope and Deliverables ....................................... 7
   1.6 Report Outline .................................................. 8

2 Methodology 8
   2.1 Infrastructure and Pipeline Design .......................... 8
   2.2 Model Training and Optimization ............................. 9
   2.3 Choice of software ............................................ 10

3 Results and Discussion 11
   3.1 Data Pipeline .................................................. 11
   3.2 Data Collection and Analysis .................................. 11
   3.3 Data exploration ................................................ 12
   3.4 Data preprocessing ............................................. 13
   3.5 Dashboard Development ........................................ 14
   3.6 Preliminary Model Training .................................... 15
   3.7 Difficulties Encountered ...................................... 21
   3.8 Remaining Limitations and Future Work ..................... 21

4 Plans and Timeline 22

5 Conclusions 23
List of Figures

Figure. 1 Example of a scam (plagiarized) NFT collection on Looksrare 6
Figure. 2 Infrastructure and Pipeline Design 8
Figure. 3 Homepage of the Prototype Dashboard 14
Figure. 4 Model output page 15
Figure. 5 ROC curves for LGBM classifier 17
Figure. 6 Classification report for LGBM classifier 18
Figure. 7 Confusion matrix for LGBM classifier 18
Figure. 8 Feature importance plot 19
Figure. 9 SHAP feature importance of class safe 20
Figure. 10 SHAP feature importance of class suspicious 20

List of Tables

Table. 1 Overview of the datatypes in the dataset 12
Table. 2 Standard of DTTD’s classification 12
Table. 3 Model In-sample Performances (in descending order of accuracy) 16
Table. 4 Model In-sample Performances after tuning (in descending order of accuracy) 16
Table. 5 LBGM Out-of-sample Performance 17
Table. 6 Project Timeline 22
1 Introduction

The purpose of this chapter is to introduce the interim progress report, beginning with a brief background on Non-Fungible Tokens (NFTs). This will be followed by a concise problem statement describing the current issues, as well as outlining the purpose and final deliverable of the project. Finally, an overview of the structure and content covered in this report will be provided.

1.1 Background

NFTs are unique digital assets that can be purchased, traded and collected (ethereum.org, 2022). Since their conception in 2017 with the launch of CryptoPunks on Ethereum by programmer Axiom Zen (Takahashi, 2018), NFTs have been creatively used in a variety of ways by artists, game developers and blockchain enthusiasts. Most NFTs are sold as standalone digital assets attached to a unique and permanent record of ownership on the blockchain (ethereum.org, 2022). Therefore, they are considered to be "tamper-proof" to buyers and collectors. As such, they have become increasingly popular with artists and musicians as a means of selling their work in digital form. However, with a lack of established fraud detection systems on popular NFT marketplaces, fraudulent activities such as artwork theft and artist impersonation has surged in recent years. This disrupts the functioning of many marketplaces, increases costs for buyers and sellers, and hinders NFT adoption in general. Therefore, there is a need to develop a real-time fraud detection algorithm to classify NFTs.

1.2 Problem Statement

Currently, popular NFT marketplaces such as OpenSea and Rarible rely on the community to report fraud cases (Pearson, 2022). However, this is not an efficient way to detect fraud because users are anonymous. Fraudsters can easily create multiple accounts to avoid being banned.
As NFTs exist on public blockchains, their transaction history and their ownership can be easily traced (Ayers, 2022). By analyzing the transaction history and metadata of NFTs, the patterns of suspicious NFT collections could be detected and classified. The vast amount of metadata and transaction history stored on the blockchain provides a rich source of information for training classifiers to detect fraudulent behaviour before the damage of such behaviour is done.

1.3 Objectives

To address the challenges of scams, this project aims to create an NFT fraud classification system by developing a machine learning classifier to pinpoint and flag suspicious NFT collections using a wide range of data sources. This will help establish the legitimacy of these contracts and the contributing factors to fraud. This information will be useful for NFT users who want to avoid problematic contracts in the future.

From our industry partner’s DTTD perspective, the system would be able to improve the user experience by sifting out scam tokens and warning users of suspicious NFT collections while they trade inside the app. This would not only reduce the costs associated with fraudulent activity but also increase trust and confidence in the industry.

1.4 Related Work

Despite the common problem of fraud in the NFT industry, there is currently no established fraud detection system on popular NFT marketplaces or Web3 services. After performing
market research, we have discovered that only the Optic Engine and Alchemy provide NFT scam detection services.

1.4.1  Optic Engine

The Optic Engine is a copymint (Optic, 2022) detection engine processing millions of NFTs minted each day, adding up to a total of 2TB metadata. They are compared to existing NFT collections in respect of visual similarities, including flips and color changes. Optic then reveals a percentage showing how likely the artwork is a counterfeit based on the results. However, copyminting is allowed by some projects to increase their popularity. For example, the Creative Commons Zero (CC0) license, a tool moving artwork to the public domain by allowing the owner to give up their copyrights, was used by Goblintown to encourage the emergence of derivatives.

1.4.2  Alchemy

Alchemy offers a public API (Alchemy, 2022) that helps to filter spam NFTs. They provide a list containing the addresses of smart contracts on Ethereum having been classified as spam. The API also checks whether a particular NFT originates from any of the suspicious contracts in the list. Currently, the API has identified more than 5000 spam contracts. However, the classifications are still in beta testing and contain many false positives and lack accuracy upon DTTD’s investigation.

As existing solutions are not holistic enough and only focus on narrow aspects of NFTs, there is a need for a comprehensive classifier that considers more factors.

1.5  Scope and Deliverables

The deliverables of the project include a deployed machine learning model that can classify and detect suspicious NFT collections on the Ethereum chain, the blockchain most commonly used for NFTs. To facilitate integration of the classifier into the DTTD infrastructure, a dashboard will be developed for DTTD to monitor the models results, create new labelled data and retrain a model if needed.
1.6 Report Outline

The following chapters of the report are arranged as follows. Chapter 2 provides an overview of the two-part methodology adopted in this study. It includes explanations for various software choices, data collection methods and decisions made throughout the research process. Chapter 3 includes the initial findings and limitations of the project as well as how to mediate them. Future work is detailed in Chapter 4 while Chapter 5 contains a summary of the progress that has been made to date.

2 Methodology

The following chapter discusses the project methodology, which is divided into two parts. The first part focuses on laying the infrastructural foundation (such as data collection, and dashboard integration) to pave the way for the second part, which is dedicated to model training and optimization. This chapter will conclude with a detailed explanation of the data requirements and reasoning for various technical decisions that were made.

2.1 Infrastructure and Pipeline Design

As the proposed system is to be integrated into the DTTD technical stack, the development of the infrastructure will be closely linked to the data pipeline and dashboard development.

Figure 2: Infrastructure and Pipeline Design

Figure 2 illustrates the proposed infrastructure and pipeline design. The system is being
built on top of the existing DTTD cloud stack, which is built on Amazon Web Service.

Data is first fetched from external sources (such as OpenSea and Etherscan) and are stored in databases (highlighted in blue) after being processed and cleaned. The data is then fed into the machine learning pipeline (highlighted in green). The pipeline consists of a data pre-processing step, a model training step, and a model evaluation step.

2.1.1 Dashboard Development

An internal dashboard serves as the front-end interface of the system. It will be used by staff in DTTD to monitor the model’s performance, to manually override classifications, and to create label new data if needed.

As the model matures, the interface also allows the overriding of classifications from the model output and model retraining. The interactive dashboard would provide DTTD with the necessary flexibility and oversight throughout the development process. As a result, incremental learning of the classifier can be achieved with an iterative approach of training and optimization.

2.2 Model Training and Optimization

The second part of the project focuses on the model training and optimization. The model is trained on a dataset of NFTs that are classified by DTTD staff.

2.2.1 Machine Learning Approach

The flagging of problematic contracts requires manual labeling, and most contracts have unknown safety parameters, so a semi-supervised approach is justified. This approach combines a small amount of labeled data with a large amount of unlabeled data during training. Semi-supervised learning falls between unsupervised learning (with no labeled training data) and supervised learning (with only labeled training data) (Fabio & Ira, 2006).

As there are new NFT contracts every day, an online/continual learning approach is suitable to iteratively update the model with new information. (Huyen, 2022)
Since data are only available in a sequential manner, the best predictor cannot be obtained with the entire training data set at once by batched learning techniques, which necessitates the use of online learning approaches. Online learning can also combat data distribution shift, which is common in fraud detection tasks. (Huyen, 2022)

2.2.2 Model Architectures

Algorithms such as the support vector machine, random decision forests, and gradient boosting trees have been considered for the NFT model classifier. These techniques are chosen as they are robust to the presence of outliers and noise in the data. (Burkov, 2019) However, more experimentation is required to determine which algorithm best suits our application requirements. Therefore, we will continue to explore different algorithms and validate the model through experiments.

2.3 Choice of software

2.3.1 Programming Language

The programming language used for the development of the system is Python. Compared to the common alternative R, python is used more widely for general data science purposes as it provides a more comprehensive set of libraries for data manipulation and machine learning. (Cloud, 2021)

2.3.2 Machine Learning Framework

The machine learning framework used is pycaret (Pycaret, 2022). It is a low-code machine learning library in Python that allows users to perform end-to-end machine learning experiments with minimal code. It is also a good fit for the project as it provides a wide range of machine learning algorithms and tools for model training and optimization.

2.3.3 Dashboard Framework

The dashboard is written in streamlit (Streamlit, 2022). It is a popular light-weight dashboard python framework. Compared to other similar frameworks like bokeh and dash, streamlit is simpler and enables faster prototyping.
3 Results and Discussion

In this chapter, the findings of the project to date since the previous progress report is presented. A discussion of the challenges encountered and possible approaches to address them is also included.

3.1 Data Pipeline

A robust data pipeline is vital to any strong data analysis, which expedites both model development and data updating. Because some project steps - such as extensive data preprocessing or multiple training models - can take a while, it is inefficient to have to start them from scratch every time a new experiment is run.

We use Makefiles to manage the dependencies of steps by automatically generating a dependency graph. This process ensures that only necessary tasks are executed. Additionally, multiple independent tasks can be executed in parallel, which increases pipeline efficiency and provides shorter feedback loops. Currently, our pipeline is used for data cleaning and preprocessing; however, as the design matures, model training will also be integrated into the final design.

3.2 Data Collection and Analysis

As mentioned in previous chapters, the main data sources for the project are collection metadata and on-chain activity. Collection metadata of around 160000 NFT collections, along with DTTD’s rule-based classifications, were exported from their databases into text files for analysis. For on-chain activity, 20 Million NFT transactions on the Ethereum network have been scrapped from Dune Analytics.

As the on-chain transaction data is larger than 20GB in size, the project team were not successful in trying to load the entire dataset into the memory using the data library Pandas. Alternative libraries that support the loading of larger-than-memory datasets such as Polars and Dask will be considered.
3.3 Data exploration

The dataset mainly consists of three types of data, namely social data, numerical data and binary data. Table 1 shows an overview of the datatypes.

<table>
<thead>
<tr>
<th>Datatype</th>
<th>Definition</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Social</td>
<td>Data collected from social media networks</td>
<td>external.instagram referring to the Instagram URL of the collection</td>
</tr>
<tr>
<td>Numerical</td>
<td>Data in the form of numbers</td>
<td>display.stats.holders referring to the number of holders of the collection</td>
</tr>
<tr>
<td>Categorical</td>
<td>Data that can be grouped into categories</td>
<td>display.state referring to the DTTD’s label on the token</td>
</tr>
</tbody>
</table>

Table 1: Overview of the datatypes in the dataset

Among the categorical features, display.state refers to DTTD’s label on the NFTs. We will consider it as the target variable in the model training. DTTD classified the NFTs into six classes, details are shown in table Table 2.

<table>
<thead>
<tr>
<th>Class</th>
<th>Standard</th>
</tr>
</thead>
<tbody>
<tr>
<td>Safe</td>
<td>Verified on Opensea</td>
</tr>
<tr>
<td>Normal</td>
<td>1. Approved on Opensea, or</td>
</tr>
<tr>
<td></td>
<td>2. Smart contract is an known ERC1967 / ERC1167 proxy contract, or</td>
</tr>
<tr>
<td></td>
<td>3. Total volume traded is larger than 100 ETHs</td>
</tr>
<tr>
<td>Hidden</td>
<td>1. Hidden on Opensea, or</td>
</tr>
<tr>
<td></td>
<td>2. Less than 10 functions in the smart contract, or</td>
</tr>
<tr>
<td></td>
<td>3. Smart contract has less than 1kb bytecode size</td>
</tr>
<tr>
<td>Deleted</td>
<td>The contract self-destructed</td>
</tr>
<tr>
<td>Caution</td>
<td>Not falling into any other categories</td>
</tr>
<tr>
<td>Suspicious</td>
<td>The number of times setApprovalForAll is called is too few</td>
</tr>
</tbody>
</table>

Table 2: Standard of DTTD’s classification

setApprovalForAll is a smart contract function granting the operators the right to transfer all tokens of the sender (OpenZeppelin, 2022). Thus, if the function is seldom called, it implies that the collection is inactive in the market, giving that the number of transfer of its NFTs is too few.
3.4 Data preprocessing

The dataset provided by DTTD requires preprocessing before it can be used for model training. The data preprocessing stage is divided into 3 steps.

3.4.1 Removing noisy data

The first step is to remove data with labels “deleted” as they will no longer be available in the market. After that, we have removed data with fewer than 5 holders. These inactive NFTs do not contain enough information to be useful for model training, and were regarded as noisy data. This prevents data corruption, which may lead to a false sense of accuracy and conclusions.

3.4.2 Data encoding

The second step is to encode the data. Machine learning algorithms like decision trees can handle categorical features directly without data transformation. Yet, many other algorithms can only deal with numerical features before training a model (Andrew, 2022). Therefore, we have encoded categorical data such as display_state into numerical data.

3.4.3 Deriving data

The last step is to derive additional information from existing data. Some of the features do not contribute much to the model training without further processing. For example, display_name referring to the collection name of the NFTs cannot be compared nor encoded directly. Thus, we focus on their existence (or lack thereof) rather than their value. We have converted those features to boolean variables denoting whether their values are null or not. As a result, new boolean features have been included the classifier.
3.5 Dashboard Development

Since the dashboard was designed for DTTD’s internal use, the dashboard requires connections to their Databases containing the necessary data to display. Direct read queries from DTTD’s databases have been established and the data is cached locally to increase the responsiveness of the dashboard. Data visualization charts have been added to the dashboard to show the distribution of classification of NFT collections in the database. To facilitate more nuanced analytics in the dashboard, the feature to filter queries based on classifications and blockchains (e.g. Ethereum and Polygon) has also been also implemented.
An interface to display the model output has also been added to the dashboard (see figure 4). As the model development progresses, the dashboard will be updated to display the performance of the models, allowing DTTD’s staff to quickly and effectively identify potential NFT fraud.

### 3.6 Preliminary Model Training

#### 3.6.1 In-sample Performance

We have trained, tuned, and compared the performance of several different models with 90% of the NFT metadata as training set, using only the numerical and categorical variables. Five best performing models were identified according to their in-sample Area Under the Receiver Operating Characteristic Curve (AUC) score. On a scale from 0 to 1, with 1 being the best and 0.5 being as good as random choosing, the metric can be understood as the model’s capacity to accurately categorize classes. (Huyen, 2022)

From Table 3, we can see that tree-based classifiers random forest and light gradient boosting machines (LGBM) have the best performance. The random forest model had the highest accuracy of 0.9362, while the AUC of the LGBM model was the highest among all the models, being 0.9924. To better compare the models, we also looked into the recall, precision and F1 score.
<table>
<thead>
<tr>
<th>Model</th>
<th>Accuracy</th>
<th>AUC</th>
<th>Recall</th>
<th>Precision</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random Forest Classifier</td>
<td>0.9362</td>
<td>0.9919</td>
<td>0.8967</td>
<td>0.9373</td>
<td>0.9363</td>
</tr>
<tr>
<td>Light Gradient Boosting Machine</td>
<td>0.9350</td>
<td>0.9924</td>
<td>0.8990</td>
<td>0.9364</td>
<td>0.9354</td>
</tr>
<tr>
<td>Extra Trees Classifier</td>
<td>0.9311</td>
<td>0.9888</td>
<td>0.8835</td>
<td>0.9322</td>
<td>0.9311</td>
</tr>
<tr>
<td>Gradient Boosting Classifier</td>
<td>0.9263</td>
<td>0.9921</td>
<td>0.8975</td>
<td>0.9303</td>
<td>0.9273</td>
</tr>
<tr>
<td>Decision Tree Classifier</td>
<td>0.9138</td>
<td>0.9458</td>
<td>0.8709</td>
<td>0.9192</td>
<td>0.9158</td>
</tr>
</tbody>
</table>

Table 3: Model In-sample Performances (in descending order of accuracy)

Recall and precision are defined as

\[
Recall = \frac{TP}{TP + FN} \quad \text{and} \quad Precision = \frac{TP}{TP + FP},
\]

where TP is the number of true positives, FN is the number of false negatives, and FP is the number of false positives (Burkov, 2019).

F1 score measures the balance between precision and recall, and is calculated as the harmonic mean of precision and recall (Burkov, 2019).

\[
F1 = \frac{2 \cdot \text{Recall} \cdot \text{Precision}}{\text{Recall} + \text{Precision}}
\]

The precision and F1 score of the random forest model were 0.9373 and 0.9363 respectively, which were the highest among all the models. However, the LGBM model had a higher recall of 0.8990. Still, we cannot judge which model has the best performance. Hence, we further tuned the models with different hyperparameters such as learning rates. Table 4 shows the performance of the tuned models.

<table>
<thead>
<tr>
<th>Model</th>
<th>Accuracy</th>
<th>AUC</th>
<th>Recall</th>
<th>Precision</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Light Gradient Boosting Machine</td>
<td>0.9362</td>
<td>0.9928</td>
<td>0.9032</td>
<td>0.9378</td>
<td>0.9366</td>
</tr>
<tr>
<td>Random Forest Classifier</td>
<td>0.9158</td>
<td>0.9807</td>
<td>0.8599</td>
<td>0.9193</td>
<td>0.9155</td>
</tr>
</tbody>
</table>

Table 4: Model In-sample Performances after tuning (in descending order of accuracy)

After tuning the models, we found that the LGBM model outperformed the random forest model in terms of accuracy, AUC score, recall, precision and F1 score. Therefore, the LGBM model was the best performing model in terms of in-sample performance, and was able to correctly classify 94% of the NFTs according to their fraud category.
3.6.2 Out-of-sample Performance

The out-of-sample performance of the LGBM model was evaluated using the remaining 10% of the NFT metadata as the test set.

<table>
<thead>
<tr>
<th>Model</th>
<th>Accuracy</th>
<th>AUC</th>
<th>Recall</th>
<th>Precision</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Light Gradient Boosting Machine</td>
<td>0.9438</td>
<td>0.9945</td>
<td>0.8813</td>
<td>0.9420</td>
<td>0.9422</td>
</tr>
</tbody>
</table>

Table 5: LBGM Out-of-sample Performance

The out-of-sample performance of the LGBM model was roughly the same as the in-sample performance. This indicates that the model did not overfit to the training set, and was generalizable to unseen data. In short, the LGBM model was the best performing model in terms of in-sample and out-of-sample performance.

3.6.3 ROC curve

![ROC Curves for LGBM Classifier](image)

Figure 5: ROC curves for LGBM classifier

Figure 5 demonstrates the ROC curves of different classes. The AUC ranged from 0.98 to 1.0, showing that the classifier had a high diagnostic ability.
3.6.4 Classification report & confusion matrix

Figure 6: Classification report for LGBM classifier

Figure 7: Confusion matrix for LGBM classifier

Figure 6 and Figure 7 display the capability of the LGBM classifier to classify the NFTs. The classifier was good at classifying hidden, cautious, and suspicious NFTs, with the precision, recall, and F1 score of these classes all higher than 0.9. Yet, when
it comes to predicting safe and normal NFTs, there was a lot of room for improvement. Specifically, the recall of the predicted label normal was 0.729. Moreover, when NFTs were predicted as safe, the precision, recall, and F1 score were 0.683, 0.724, and 0.703 respectively.

The poor performances of the classifier on predicting safe and normal class NFTs were correlated. From figure fig:confusion-matrix, class normal accounted for 92% of the false positive and 63% of the false negative cases of class safe, while 64% of the false negative cases of class normal were classified as safe. Apparently, there is a need to better distinguish between these two classes.

3.6.5 Feature importance

The LightGBM classifier uses split feature importance to evaluate the contribution of each feature to the predicted results. It is based on the number of times the tree nodes split on the features. The premise is that importance of features is proportional to the times it is split.

![Feature Importance Plot](image)

Figure 8: Feature importance plot

Figure 8 shows that numerical features like token supply, number of holders, total trading volume, and token floor price are among the most important features, whereas categorical variables, such as whether the smart contracts have been verified and whether the collections have description, contribute less to the predicted results.
3.6.6 SHAP values

The split feature importance reflects the weights of each feature in the results, but it has one problem – binary features (boolean) only have two values, so they cannot be split more than once in each tree. For example, after splitting has_twitter into True and False, there is no way to split it further. Therefore, the importance of numerical features often outweighs that of categorical features.

Given this, we apply SHAP values (SHapley Additive exPlanations) to evaluate the feature importance of each class. SHAP value is a method using cooperative game theory to calculate the average marginal contribution of features (players) in the game (Vinícius, 2022). The contribution is known as Shapley value. Instead of weights, Shapley value focuses on the gain and cost of each feature. Thus, features with binary or continuous values are treated fairly under this method.

Figure 9: SHAP feature importance of class safe

Figure 10: SHAP feature importance of class suspicious

Figure 9 and figure 10 show the SHAP feature importance of class safe and suspicious respectively. From the comparison, we found that the most important features classifying safe NFTs are mostly numerical, while binary features contribute more to the predicted results when it comes to suspicious NFTs. Particularly, safe NFTs are generally more
active in the market, while suspicious NFTs may fail to fulfill some basic requirements of other verifiers.

3.7 Difficulties Encountered

In the process of data preprocessing and training, we encountered some problems related to data imbalance and rate limits. Thankfully, they had been resolved at the time of writing.

3.7.1 Data imbalance

After removing the noise, data labeled as “safe” only account for a little portion of the whole dataset. Thus, the classifier may get biased towards other classes, leading to a false sense of accuracy. In particular, the overall performance of the classifier may seem satisfactory, while it may be less capable of distinguishing safe NFTs from others. To solve this problem, we will (i) include more data labeled as “safe” into the dataset and (ii) tell pycaret to fix it before initializing the training environment. These methods mitigate data imbalance by upsampling the minority classes and downsampling the majority classes. With a balanced dataset, the classifier will be more impartial, while the performance metrics can better reflect the actual performance of the model.

3.7.2 Rate limits

A rate limit exceeded error occurs when too many requests and calls are made in a given amount of time. APIs and web service endpoints are often protected by website security tools, which provide a variety of web protection services, such as captcha. For instance, the discord API is protected by Cloudflare, which rejects requests and temporarily blocks the IP addresses when the rate limit is exceeded. To overcome it, we set a time interval between each request, which greatly increases the time cost.

3.8 Remaining Limitations and Future Work

There are still remaining issues to be addressed in future work.

Firstly, regarding the nature and the quality of data, the data labelled by DTTD are mostly based on heuristics and might not reflect the true nature of those contracts. With
only a few hundred collections with known classifications available for analysis, it can be challenging to achieve high prediction accuracy with our current dataset size.

In terms of textual data, the current model is only able to process numerical and categorical variables. As such, the model is unable to make use of the textual data in the NFT metadata, which can be a rich source of information. Future work will include the use of natural language processing techniques to extract meaningful information from the textual data such as NFT contract description.

There is also a need to transform and remodel the scrapped on-chain transaction data as a graph data structure to make inferences based on interactions and connections between NFT users. As such, future work will focus on expanding the dataset by incorporating additional data on NFT transactions and user behaviour.

To tackle these issues, we plan to collaborate closely with the DTTD technical team and will be exploring various other fraud detection techniques and approaches to stay up-to-date and ensure that our system will be as robust as possible.

4 Plans and Timeline

<table>
<thead>
<tr>
<th>Time Period</th>
<th>Task</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>2022 Sept</td>
<td>Detailed project plan, Project research</td>
<td>Done</td>
</tr>
<tr>
<td>2022 Oct</td>
<td>Data Collection, Data pipelines</td>
<td>Done</td>
</tr>
<tr>
<td>2022 Nov</td>
<td>Train and deploy preliminary classifier</td>
<td>Done</td>
</tr>
<tr>
<td>2022 Dec</td>
<td>Detailed interim report</td>
<td>Done</td>
</tr>
<tr>
<td>2023 Jan</td>
<td>Finish implementation of Phase 1</td>
<td>In progress</td>
</tr>
<tr>
<td>2023 Feb</td>
<td>Optimize model</td>
<td>TODO</td>
</tr>
<tr>
<td>2023 Mar</td>
<td>Further code refinement</td>
<td>TODO</td>
</tr>
<tr>
<td>2023 Apr</td>
<td>Project exhibition and final report</td>
<td>TODO</td>
</tr>
</tbody>
</table>

Table 6: Project Timeline

Table 6 outlines the timeline of our project. The upcoming key milestone is the complete infrastructure of the system pipeline outlined in Chapter 2. In the future, we will continue to refine the model and the system pipeline to improve the accuracy of the model.
5 Conclusions

With the recent rise in frauds and scams related to NFTs, efficient systems for detecting these kinds of crimes are now a necessity. Such systems should be able to identify potential red flags based on attributes and behaviour patterns. This report discusses the design of a proposed NFT fraud detection system that employs data analytics and machine learning algorithms. This system would cooperate with the DTTD, the project partner to safeguard NFT transactions and reduce fraudulent activities.

Although we have faced some challenges in developing this system, such as the inadequate quality of data, we are convinced that with more investigation and by working more closely with DTTD’s skilled staff, these obstacles can be surmounted.

As we continue to explore different algorithms and train our model classifier, we seek to enhance the accuracy and sensitivity of our system in detecting fraudulent NFTs. Our next steps will involve performing training and optimizing models using refined datasets, and incorporating feedback from relevant stakeholders on how we can further improve the performance of our system. Overall, we remain optimistic about the potential of our NFT fraud detection system and are committed to working together with DTTD in building a safer, more secure NFT ecosystem.

We believe that our proposed NFT fraud detection system has great potential to help safeguard the emerging NFT ecosystem. As we continue to develop our system and collaborate with DTTD, we are confident that together, we can create an ecosystem where users can safely and securely transact NFTs.
References


