

Clustering and analysis of user behaviour in blockchain: A case study of Planet IX

Dorottya Zelenyanszki^{a,*}, Zhé Hóu^a, Kamanashis Biswas^b and Vallipuram Muthukkumarasamy^a

^aGriffith University, Australia

^bAustralian Catholic University, Australia

ARTICLE INFO

Keywords:
blockchain
Non-fungible tokens
user behaviour
clustering

ABSTRACT

Decentralised applications (dApps) that run on public blockchains have the benefit of trustworthiness and transparency as every activity that happens on the blockchain can be publicly traced through the transaction data. However, this introduces a potential privacy problem as this data can be tracked and analysed, which can reveal user-behaviour information. A user behaviour analysis pipeline was proposed to present how this type of information can be extracted and analysed to identify separate behavioural clusters that can describe how users behave in the game. The pipeline starts with the collection of transaction data, involving smart contracts, that is collected from a blockchain-based game called Planet IX. Both the raw transaction information and the transaction events are considered in the data collection. From this data, separate game actions can be formed and those are leveraged to present how and when the users conducted their in-game activities in the form of user flows. An extended version of these user flows also presents how the Non-Fungible Tokens (NFTs) are being leveraged in the user actions. The latter is given as input for a Graph Neural Network (GNN) model to provide graph embeddings for these flows which then can be leveraged by clustering algorithms to cluster user behaviours into separate behavioural clusters. We benchmark and compare well-known clustering algorithms as a part of the proposed method. The user behaviour clusters were analysed and visualised in a graph format. It was found that behavioural information can be extracted regarding the users that belong to these clusters. Such information can be exploited by malicious users to their advantage. To demonstrate this, a privacy threat model was also presented based on the results that correspond to multiple potentially affected areas.

1. Introduction


Public blockchains have the benefits of immutability, decentralisation and anonymity [1], and have a wide range of potential use cases, such as distributed identity management [2], record linkage [3] and metaverse [4]. DApps that run on top of blockchains and NFTs that represent various types of physical and digital objects, such as arts [5], have become widely used in recent years. However, every submitted blockchain transaction can be publicly traced by anyone, thus, the transaction data can be analysed, which can uncover behavioural information. This is somewhat analogous to being able to analyse people's browser history and activities, which is a serious privacy concern. Zhang et al. [6] described this as the issue of linkability of transactions, which makes de-anonymisation inference attacks possible. There have been multiple research works conducted on blockchain transaction analysis, however, these have differing focuses and they usually do not involve the events that are emitted by the transactions in their analysis. Analysis in regards to NFTs is mainly limited to their market analysis and related works do not consider their actual usage in dApps. For instance, Tao et al. [7] presented a network analysis of the Bitcoin blockchain, and Pelechris et al. [8] provided an analysis of the NBA's TopShot NFT marketplace regarding illegal

activities such as money laundering or trading illicit goods. However, the type of user behaviour that can be extracted from dApps through transaction analysis remains unknown which prompted this research as there is a need to present the full extent of the previously mentioned privacy concern. Furthermore, behaviour analysis can be leveraged to identify malicious entities such as phishing scammer participants [9].

This work aims to analyse behavioural information regarding in-app activities, including NFT usage by users of blockchain-based games. We argue that this type of information can be considered problematic, as malicious users can use it to their advantage, for example, for impersonation attacks or targeted scams [10, 11]. It can also be used by related parties such as game providers, to track the users and their activities and implement changes based on the corresponding analysis. Tracking users in blockchain networks can be done in multiple application areas. Hu et al. [12] studied tracing users for blockchain data audit purposes.

In our work, a blockchain-based game called Planet IX¹ was chosen as a case study application. Transactions from multiple smart contracts were collected and used to get the general transaction information and the emitted event logs. The event logs were decoded to obtain all the event data which was then converted into a graph format, thereby allowing it to be queried to access information easily. All related event sequences of users and NFTs were extracted through database queries, which allowed the formation of general in-game actions. From the event sequences and formed actions,

¹<https://planetix.com/>

 dora.zelenyanszki@griffithuni.edu.au (D. Zelenyanszki);
z.hou@griffith.edu.au (Z. Hóu); kamanashis.biswas@acu.edu.au (K. Biswas);
v.muthu@griffith.edu.au (V. Muthukkumarasamy)
ORCID(s): 0009-0004-1300-0357 (D. Zelenyanszki);
0000-0001-7164-0580 (Z. Hóu); 0000-0003-3719-8607 (K. Biswas);
0000-0002-6787-6379 (V. Muthukkumarasamy)

the corresponding user action flows for every user were presented. Similarly, for every tokenId (which is a property that corresponds to an NFT), the associated actions were extracted. They were also linked to user wallet addresses, which enabled the presentation of extended user flows that not only show the conducted activities but also which tokens were used at what time by the performed actions.

These extended user flows served as input for a GNN model that assigned unique graph embeddings for each of them. A graph embedding is a vector representation of the graph that is suitable to be used for clustering. Following that, several clustering algorithms were utilised to cluster the user flows into separate unique behavioural clusters. The algorithms were evaluated based on common metrics, and one of them was selected as the current method for the clustering process. The output behavioural clusters were then described which presented the separate types of user behaviours that are possible in this chosen blockchain-based game. A corresponding privacy threat model was also included to present how this can potentially be utilised maliciously.

This research presents a new type of blockchain analysis that focuses on behavioural information that can inform users' activities, interactions and habits. The malicious actors or dApp developers do not need to fully identify the person behind the user as they can construct a profile based on the behavioural information and use that to their advantage to conduct activities like targeting the original user based on past actions or using the constructed profile to perform impersonation-related attacks. Behavioural information does not directly affect users' privacy but by linking multiple activities, relations and habits of users together, it can reveal sensitive information. Therefore, behavioural information is privacy-sensitive and has to be protected. The overall contributions of this research are summarised as follows:

- Proposed an action synthesis method that presented unique actions from a blockchain-based application which were used to construct users' action flows.
- A user behaviour analysis pipeline was proposed that adopts GNN. Multiple clustering algorithms were utilised within the pipeline to cluster the user flows into unique behavioural clusters. These clusters can be used to analyse different user groups.
- The results were compared from the clustering algorithms in order to identify the well performing ones for this task. Descriptions were provided of the different behavioural clusters that a chosen clustering algorithm generated. A privacy threat model was also presented that corresponded to the extracted behaviour clusters and was linked to potential application areas.

The rest of the paper is constructed as follows: Section 2 presents related research works. In section 3, the proposed user behaviour analysis pipeline is introduced in full detail. Section 4 shows the clustering results from the chosen

clustering algorithm and describes the privacy threat model. Finally, the paper is concluded in section 5.

2. Related work

In this section, the corresponding related works on blockchain transaction analysis, blockchain-based games and the usage of GNN are presented to highlight the differences and limitations in this area of research.

Transaction analysis has been utilised for multiple purposes in previous research. Wu et al. [13] proposed a novel transaction tracing tool for account-based blockchains called TRacer to trace illicit financial flows. They formed the accounts and their token transfer relationships into a graph format, which enables them to model Decentralised Finance (DeFi) actions. Those were categorised into two patterns: Xfer (transfer, minting, burning) and Swap (add liquidity, remove liquidity, trade), where Xfer refers to sending/receiving tokens and Swap means the exchange of a token for another token. Bonifazi et al. [14] proposed a new approach to classify Ethereum users. They built a social network that included the user addresses and their transactions, introduced multiple features to characterise the addresses, and utilised multivariate time series. Four classes of interest from information provided by Etherscan were also defined: Token Contract class involves addresses using tokens and not Ether. The Exchange class includes the addresses that buy and sell cryptocurrencies. The Bancor class is where users who allow clients to deposit and convert belong. Finally, the Uniswap class includes users who use the Uniswap protocol.

There has been research that focused on blockchain-based games. Jiang et al. [15] investigated how blockchain can introduce advantages regarding loot boxes. These boxes are goods involving a probability of obtaining one/more in-game virtual assets. They modelled in-game interactions between the provider and the players as a two-stage Stackelberg game. However, they utilised that to conduct market analysis, for example, the price set by the game provider, the users' utility and the impact of the gas fee. Jiang et al [16] collected a year of player operation data from a blockchain-based game called Aavegotchi. They analysed this data in aspects of gaming and finance. The analysis includes daily active addresses, functions and density distribution to reflect player behaviours. Finally, they applied an unsupervised Self-Organising Map (SOM) algorithm to divide user groups into different clusters. Their work presented a similar aim as the behavioural clusters in this work, as they provided information on user activity levels and included the functions from the smart contracts in their data collection. However, when presenting their clusters, they mainly focused on describing how much the corresponding users would invest. They included their interactions but not in detail and did not provide graph visualisation.

GNN is leveraged in the proposed pipeline. It can be applied to multiple research tasks as it is shown as follows. Wu et al. presented a survey on how GNN can be applied to recommender systems [17]. They presented multiple areas

where they can be utilised such as in user-item collaborative filtering. They apply it to exploit high-order connectivity from user-item interactions, or in social recommender systems. They use GNN to model social interactions. Jin et al. [18] similarly showed how GNN is being used in time series-related tasks, like classification and anomaly detection, as they are able to capture inter-variable and inter-temporal relationships. They can be applied to blockchain-related use cases as well. For example, Liu et al. [19] proposed a novel GNN framework, Filter and Augment Graph Neural Network, to provide an Ethereum transaction network embedding model. In the proposed pipeline, user flows are extracted, and they are subgraphs of the second local database's entire graph data. Alsentzer et al. [20] introduced SUBGNN that can learn subgraph representations, including if they are disentangled. They also detailed the challenges of utilising GNNs in regard to subgraphs, which we encountered when applying a GNN model to provide subgraph embeddings.

The gap: As presented, other blockchain analysis works may include the introduction and usage of actions in analysis, however, the application area is very different. Mainly finance-related. They can also introduce classes that are also a categorisation of the users, similar to the behavioural clusters that are introduced in this work, however, they do not have a direct relation to in-game activities. The proposed pipeline can be beneficial in describing user behaviour in multiple application areas as it can use data from various types of blockchain-based applications. For example, these clusters are based on in-game data, and as a result, they are less general and can be more expressive in the area of gaming.

Works on blockchain-based games are often market-related analyses and cannot be applied directly to this work's research area as they do not present user behaviour. Even if they include user categorisation in classes or clusters, the main focus still relies on potential investment in the game or other dApp and not on general application activities. This research addresses this gap by introducing a novel pipeline that can be used to describe different user behaviours, which can focus on non-finance-related areas as well such as NFT usage or which parts of the game are popular.

3. User behaviour analysis pipeline

In this section, the introduced user behaviour analysis pipeline is presented that can be used to reveal the type of behavioural information that can be extracted through transaction analysis. As can be seen in Figure 1, it has four components: data preprocessing, action formation and application, GNN and clustering. First, an overview of the pipeline is given and then each component is detailed.

3.1. Overview

Each step of Figure 1, namely Data preprocessing, Action formation and application, GNN model, and Clustering, is briefly described in this section to give a general overview.

These four are explained in detail later in their corresponding sections. The analysed outcomes are presented in section 4.

Data preprocessing: Transaction information is collected and stored in a graph format in a local database. By querying the graph database, event sequences can be extracted for user wallet addresses. There are two ways of extracting events that correspond to a certain wallet address: enquiring events from transactions that were submitted by the particular user who has that wallet address or extracting events based on event property information that corresponds to the address.

Action formation and user flow: The event sequences were examined, and certain patterns were identified that were used to form unique in-game actions. For every sequence, the pattern they belong to was checked, and the corresponding action steps were also added and associated with the particular user's wallet address. An action step refers to one in-game step a user makes at a given time that is represented by a timestamp where the user performs a specific action. The newly formed actions and the noted action steps enabled us to establish a new local database with two newly introduced node types: users and actions. The action steps are presented as edges between these nodes. By converting the data into a graph format, it is possible to present how and in which order the users performed these unique actions in the form of user flows. A similar process can be used to extract event sequences for NFT tokenIds. The tokenIds are unique numbers that identify an NFT within a collection. Querying the first local database, by using the particular tokenId as an event property value, lead to the extraction of the event sequences. After that, all the sequences were checked and assigned to the existing action patterns or formed new actions. The token action steps were also added. Every added action step included the wallet address of the user that was associated with the particular step (for example, the user that submitted the corresponding transaction). This enabled us to add the NFTs as nodes and the token action steps as a new type of edge in the second database. With this new information, not only the user's action activities but also their NFT usage in the form of an extended user flow can be presented. A second local database was added where the action-related information was stored as graphs.

GNN model: By examining multiple user flows, it can be seen that some users conduct very similar activities, which indicates that there is a possibility of analysing similar user behaviours if they are grouped together. In this research, this is formulated as an unsupervised clustering task. As a first step, a GNN model was leveraged to provide graph embeddings for its input graph representation. The extended user flows were passed as inputs for this model, however, there was an elimination step to reduce complexity. In the GNN model, both the node and edge features were considered. From the included wallet addresses 70% of them were randomly chosen for the training, and their corresponding user

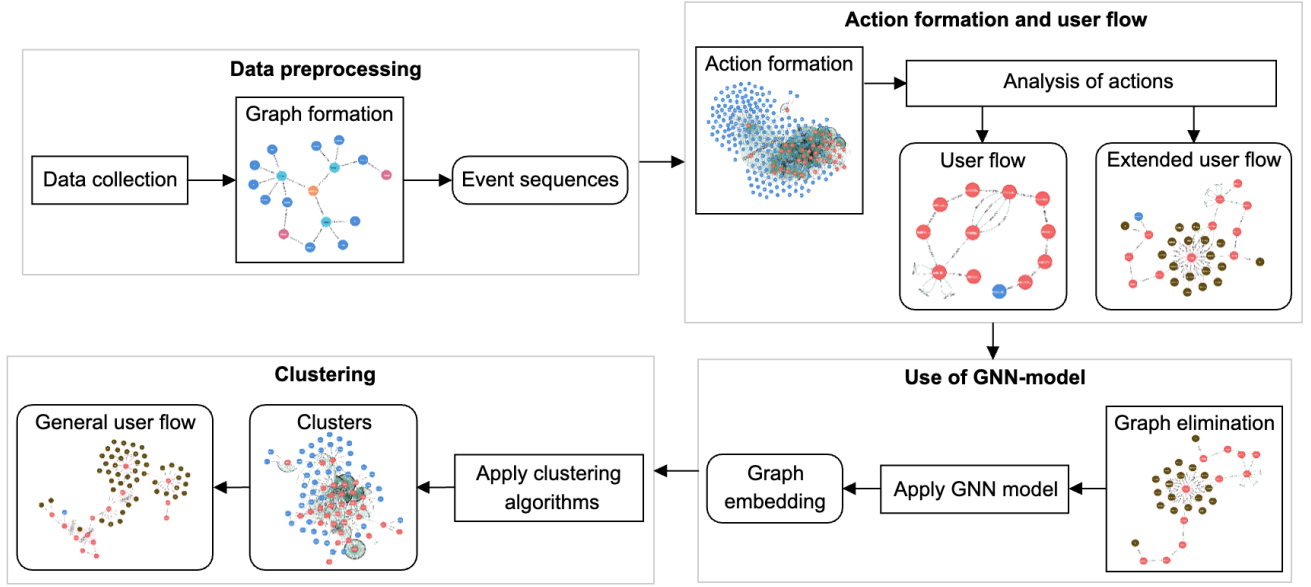


Figure 1: An overview of user behaviour analysis pipeline.

flows were added as input data. The remaining addresses were used for testing.

Clustering process: The elbow method [21] was used to estimate the number of clusters that were required because we did not have any predefined knowledge or labels. From the previous GNN step, we had two embedding arrays that were used as input in the clustering process. Multiple clustering algorithms were leveraged, and then their results were compared through multiple clustering metrics. One of the algorithms was chosen based on the comparison, and the different behaviours coming from the corresponding clusters were described and presented. Following that, the privacy threat model was also presented.

In what follows the technical details of the above approach are presented.

3.2. Data preprocessing

The first step of the methodology is described which includes the process of the data collection and how that transaction information is converted into graph format which then enables its analysis through various queries. The second part of this describes how the event sequences for the users and the NFTs can be extracted.

3.2.1. Data collection and graph formation

For this study, transaction data from a blockchain-based game called Planet IX was collected. Based on our previous work [22], APIs were leveraged, namely Polygonscan² and Alchemy³, to extract data from the Polygon blockchain where the game is deployed. In addition to the basic transaction information (such as transactionHash, blockHash and value), the events that are emitted within the transactions

are also encoded. The collected data is then transformed into graph format and stored in a local Neo4j database. This enables the process of conducting blockchain transaction analysis on the collected game data, which can reveal connections between transactions, events and event properties.

The data coming from the emitted events includes various types of information such as token and staking-related properties. It also presents a large number of involved wallet addresses which are the potential user addresses that can be examined to reveal user behaviour. By querying the established local database 12146 distinct addresses were extracted. However, this number is a combination of user, smart contract and game-related addresses.

3.2.2. Event sequencing

Since in-game behavioural information [23] is extracted, event sequences have to be linked to addresses. When building these sequences, the following has to be taken into account: the events that were emitted through the transactions that were submitted by the particular address, and the events that were part of other transactions, however, the specific wallet address is included as event property information. The latter transactions could have been initiated by a user as the differing wallet address can be just the result of the structure of the game, therefore, it has to be considered. For the first type of event sequence, the database can be queried based on the from address transaction information which corresponds to the address that submitted the transaction. The second type of event sequence is extracted by querying connections between events through event property information that has the address as the value. Besides the addresses' event sequences, the sequences that correspond to NFTs are also extracted, however, in this case, only the second type of event sequences are possible because the query is based on

²<https://docs.polygonscan.com/>

³<https://docs.alchemy.com/reference/api-overview>

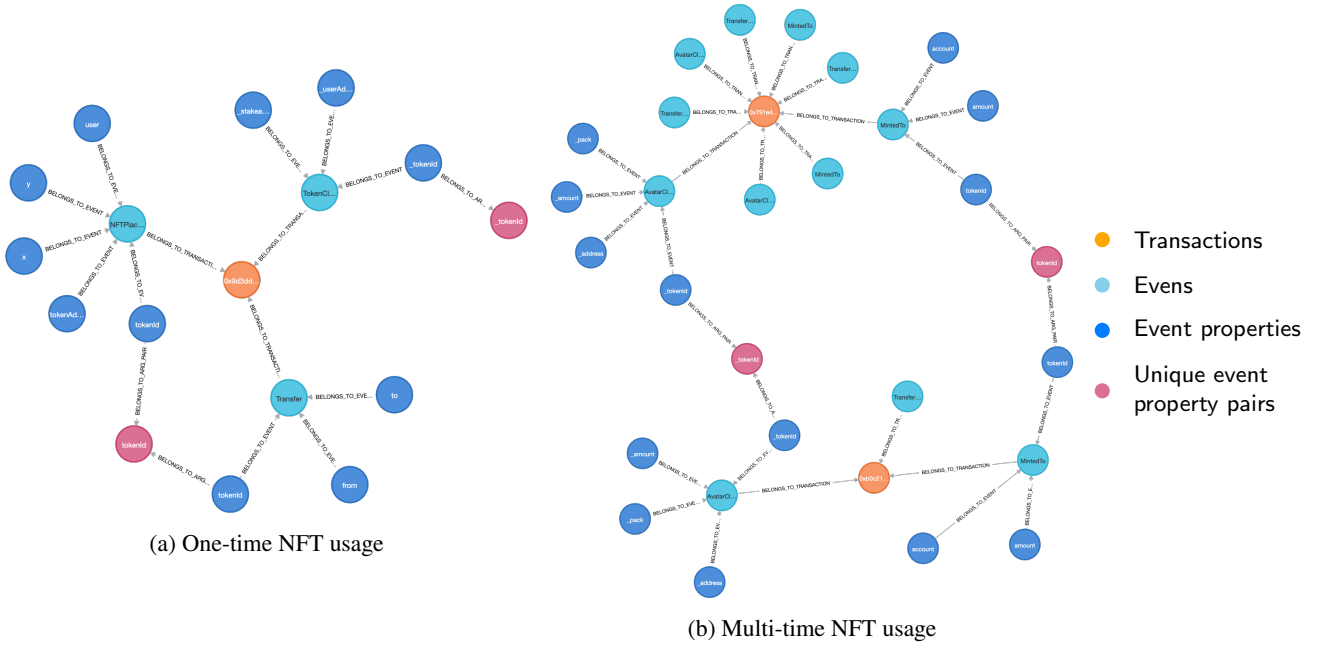


Figure 2: Categorisation of NFTs based on usage (unique event property pairs link event property information across multiple events).

the tokenIds of the NFTs. Following that, both the user and NFT event sequences are ordered based on the timestamp transaction information. By analysing these sequences, the type of activities the users or NFTs have been involved with can be learned.

When looking into the NFT event sequences, it can be seen that certain NFTs are only leveraged once by one address, whereas others can be part of several events and can be leveraged by multiple addresses. An example of the one-time NFT usage is given in Figure 2a, where an NFT is placed down to a certain location and is not referenced again. Figure 2b presents multiple-time NFT usage for a tokenId that represents an avatar that is claimed multiple times. We have to note, however, that a tokenId does not fully describe an NFT, as multiple collections can have the same tokenId value, but this still can indicate multiple usage of the same NFT. In the future, this is planned to be extended.

3.3. Action formation and user flow

The formation of the unique in-game actions from the address event sequences is presented. Two types of actions are introduced: primary and secondary. It also describes how the corresponding data can be converted into graph format. Following that, the way to present the corresponding user flows derived from the event sequences and also the method to extend them with the inclusion of the NFTs will be explained.

3.3.1. Formation of actions

In section 3.2, two types of event sequences were identified. The first type extracts them from the transactions that were submitted by the wallet address. Actions that are derived from these sequences were named as primary actions

since, in this case, it is certain that they describe address-related activities. The second type acquires the events based on connecting events through the address as a common event property. Every action that is constructed from these sequences, is secondary. Even though the address is included as event information, the action may or may not describe an activity the actor behind the address personally has been a part of. Some of the identified actions can be considered both primary and secondary as they were derived from both types of event sequences. We identified 48 actions where 38 of them were primary, one was secondary and 9 were determined as both.

When examining the transactions the events were extracted from, regardless of which case the transactions were considered in, it can be seen that the events are repeating across multiple similar transactions in unique recognisable patterns. These patterns were identified and used to extract the unique actions that all actors conduct within the game. These patterns are explained as follows:

1. Unique events: This corresponds to those transactions that have a unique set of events emitted that is fixed at every occurrence. This transaction is submitted by a wallet address.
2. Every event repeating the same amount of times: This presents transactions where, within the event set, event sequence chunks can be discovered that consist of the same set of events. The event sequence chunk is added as the new action. The sequence chunks can be clearly grouped based on their event property values.
3. Every event repeats the same amount of times except for one event, which only occurs once: This is similar to the

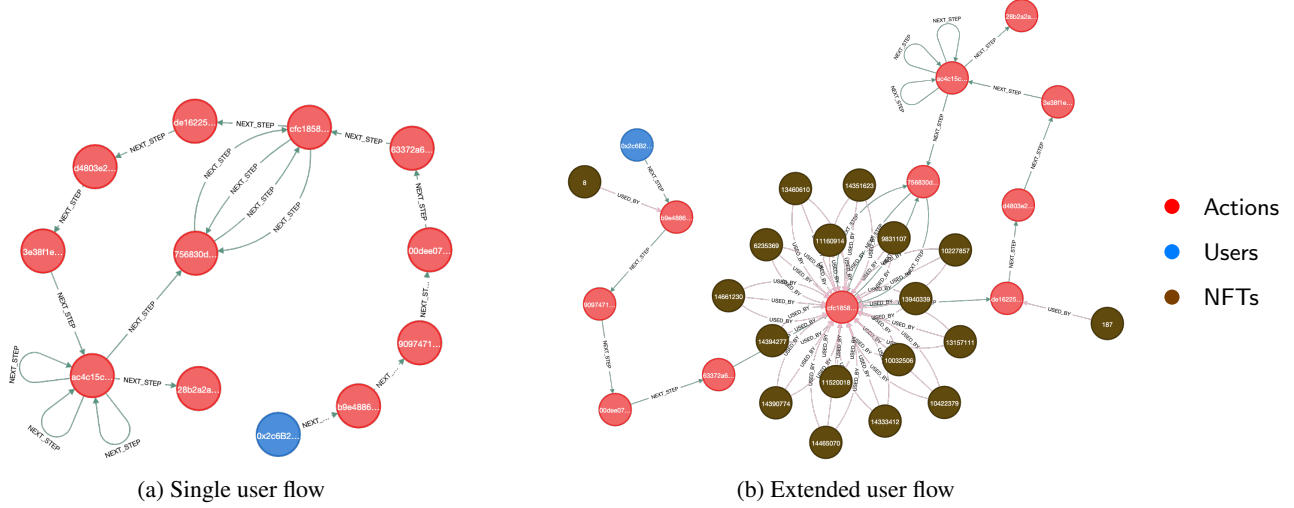


Figure 3: Examples of user flows.

Algorithm 1 Action formation from event sequences

```

1: procedure FORM_ACTIONS(type, addresses, patterns)
2:   sequences  $\leftarrow$  []
3:   actions  $\leftarrow$  []
4:   action_steps  $\leftarrow$  []
5:   for addr in addresses do
6:     if type = primary then
7:       sequences  $\leftarrow$  primary event sequences
8:     else
9:       sequences  $\leftarrow$  secondary event sequences
10:    end if
11:    order  $\leftarrow$  0
12:    for seq in sequences do
13:      current_pattern  $\leftarrow$  call match_pattern(seq, patterns)
14:      current_events  $\leftarrow$  call get_events(seq.events, current_pattern)
15:      for each action in actions do
16:        if action.events == current_events then
17:          uuid  $\leftarrow$  action.id
18:        else
19:          uuid  $\leftarrow$  generate uuid
20:          new_action  $\leftarrow$  {uuid: uuid, type: type, events: current_events}
21:          actions.push(new_action)
22:        end if
23:        new_step  $\leftarrow$  {order: order, uuid: uuid, prev_uuid: action_steps[length(action_steps) - 1].uuid, address: addr, transactionHash: seq.transactionHash, timestamp: seq.timestamp, data: seq.data}
24:        action_steps.push(new_step)
25:      end for
26:      order  $\leftarrow$  order + 1
27:    end for
28:  end for
29:  return actions, action_steps
30: end procedure

```

previous pattern except for the single count event. It is recognised that the latter event is actually a bulk event, which means that it includes event property values from the event sequence chunks initiated from the repeating events. This is added as a new action formed as the combination of the event sequence chunk and the bulk event.

- One event repeats a large number of times, whereas the rest of them only occur once: In the corresponding transactions, it can be noticed that there is one event that occurs a large but varying number of times, while other events are unique per transaction. This is added as a new action by only adding the repeating event once but by combining the event property values.
- Some events repeat at the same amount of times, but there are single count events as well: This groups all the remaining other transactions as there are no recognisable patterns. In this case, the actions are added with all the events included.

These unique actions were added by going through the event sequences. For every sequence, it was first checked which pattern they belonged to. Following that, the then-current action list (as the actions were formed, the action list was continuously growing) was examined to investigate whether the current sequence corresponded to an already identified action. If not, the sequence was added as a new action with a generated uuid (unique ID that is generated and then assigned to the action), the action type (primary, secondary or both) and the associated events. The action step was also noted with the following information: The order of the step (the order of the steps were able to be ascertained because the sequences were previously ordered based on the timestamps), the corresponding wallet address (in the NFTs' case this is the from address), in case of an NFT-related sequence the tokenId, the transactionHash, the timestamp, the previous and current action uuids and

Table 1

Properties of the action nodes.

Property	Description
Total count	total # times action is called
Min call	min # times action is called
Max call	max # times action is called
Mean call	avg # times action is called
Min assets	min # assets associated with action
Max assets	max # assets associated with action
Mean assets	mean # assets associated with action
Min tickets	min # tickets associated with action
Max tickets	max # tickets associated with action
Mean tickets	mean # tickets associated with action
Min packs	min # packs associated with action
Max packs	max # packs associated with action
Mean packs	mean # packs associated with action
Min timestamp	first time action is called
Max timestamp	last time action is called

the event data information. In the latter, the tokenIds or IDs for the assets (NFTs), tickets (the game includes a lottery activity) and packs (openable objects that include additional items like avatar NFTs), the related addresses, the values (such as reward amount) and additional IDs (like IDs associated with waste management which is one of the in-game activities) were noted. In the NFT's case, only the related addresses were noted. The action formation from sequences can be seen in Algorithm 1 where the function *match_pattern* matches the events with the established patterns and function *get_events* shortens the event list based on the matched pattern.

3.3.2. Graph representation

Similarly to our previous work [22], the extracted information was converted into a graph format and stored in a separate local Neo4j database. Three types of nodes were introduced: 1. User nodes that correspond to the unique wallet addresses, as those are the only node property they have. 2. Action nodes that present the unique set of actions that were formed from the event sequences based on the patterns from the previous sub-section. They have multiple properties that were extracted from the examination of the address and NFT action steps. They are summarised those in Table 1. 3. NFT nodes that correspond to a unique token and have only one other property (*isMultiple*), which describes whether they are one-time or multiple-time usage NFT. This new local database includes two types of relations: NEXT_STEP, which presents one step from a specific address at a given timestamp, and USED_BY, which describes one NFT usage at a given timestamp by a certain address (which can be non-user). Both of them have the same three edge properties: the corresponding wallet address, the order in which the address conducted the action or the NFT was used and the timestamp.

3.3.3. Extracting user flows

As mentioned in the previous sub-section, action steps from both the users and the NFTs are converted into a graph format in the form of the NEXT_STEP and the USED_BY relations. Since these relationships have the address as a property, the database graph data can be filtered based on that. This results in the user flow graphs that correspond to the filter address only. Since the edges also include the action step order number and the associated timestamps, by examining these user flows, the type of activities the participants have conducted can be described, and assumptions can be made in regard to their behaviour. For example, what time they conduct certain actions, in which order they perform a subset of actions or how active they are in general. Shorter user flows indicate lesser engagement in the game. Figure 3a presents an example of a single user-flow that only shows how the actions have followed each other. If the NFT nodes and their USED_BY relations are also included in the filtering, an extended user flow can be presented, which also considers how the assets were leveraged by the particular corresponding address. An example of this can be seen in Figure 3b.

3.4. Embedding and clustering

In the following, it is explained how the individual user flows are categorised into multiple separate clusters to show not only a single user behaviour but also to enable the analysis of group user behaviour. For this purpose, a GNN model was included that provides unique embeddings for every user flow. These embeddings are then given as input to the clustering algorithms, which results in clusters for all the user flows and, therefore, enables them to be analysed and visualised together. The entire process is visualised in Figure 4.

3.4.1. Definition of user flow subgraph and elimination

Let $G(V, E)$ be a graph with a set V of vertices and a set E of edges. There are three types of vertices (nodes): $A \subseteq V$ is the set of action nodes, $U \subseteq V$ is the set of user nodes, and $N \subseteq V$ is the set of NFT nodes. For $\forall u \in U$ there is a user flow S , which is a subgraph of G . For every such user flow (sub)graph, a suitable graph representation S' is generated and then given as input for the GNN model to get the corresponding subgraph embedding.

The graph representation S' for each S is being constructed as follows: As S corresponds to one user u , the related nodes consist of $V_S = \{u\} \cup \{a \mid a \in A_u\} \cup \{n \mid n \in N_u\}$ and $E = \{e \mid e \in E_u\}$ where the single user node u and its associated subset of the action nodes A_u , subset of the NFT nodes N_u and the related edges E_u are considered. The user nodes are emitted from the input graph representations for the GNN model as there is only one user node for every user flow subgraph and the user nodes only have the address property which makes them negligible when it comes to learning. The corresponding edges are also emitted as we assume that by eliminating only one edge, these user flow subgraphs are still suitable

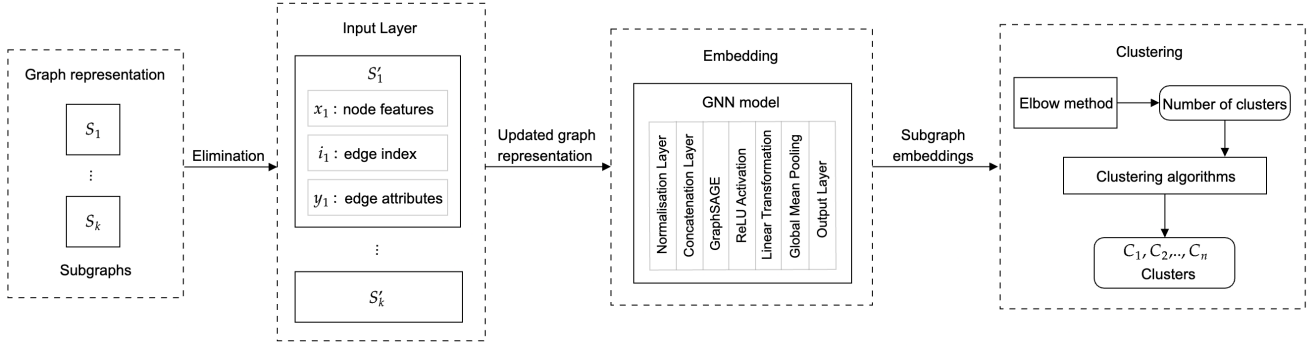


Figure 4: Overall embedding and clustering process.

for clustering into separate diverse clusters. However, this elimination reduces the complexity of the model because the dimension differences between the user nodes and the other nodes do not have to be handled. Furthermore, every \mathbf{S} that interacted with less than four unique actions is excluded from the embedding and clustering process as it is presumed that an actual user would interact with multiple parts of the game whereas a game-related address will perform the same set of actions continuously. With this step, the included address list is reduced to wallet addresses that are more likely to be actual user addresses. From the extracted 12146 addresses, only 8296 had associated event sequences and solely 716 were considered for clustering. The burn address⁴ was also eliminated. For the action nodes, the features presented in Table 1 are considered. The NFT nodes only have the *isMultiple* property to be included. The edge attributes are fully leveraged. Based on this, GNN graph representations were added for the 716 addresses.

3.4.2. GNN model

This sub-section presents the GNN model that is leveraged to produce graph embeddings. It consists of multiple layers to provide an embedding that takes into account all the necessary node features and also the edge attributes. By leveraging all, the produced embedding is clearly unique for $\forall \mathbf{S}$, therefore, is suitable to be given as input for clustering algorithms. The following is defined for the model:

- Let $\mathbf{X} \in \mathbb{R}^{N \times d_{\text{in_node}}}$ be the input node features
- Let $\mathbf{Y} \in \mathbb{R}^{M \times d_{\text{in_edge}}}$ be the input edge attributes
- Let $\mathbf{I} \in \mathbb{R}^{2 \times M}$ be the edge index
- Let $\mathbf{B} \in \mathbb{R}^N$ be the batch vector
- Let $\mathbf{H} \in \mathbb{R}^{N \times d_{\text{hidden}}}$ be the hidden node embeddings
- Let $\mathbf{Z} \in \mathbb{R}^{G \times d_{\text{out}}}$ be the output graph embeddings

As users perform different actions in varying order, the sizes of the node features, edge index and edge attributes tensors are different. In the following, all the layers the GNN model consists of are being presented:

Input Layer: The input consists of node features \mathbf{X} , edge index \mathbf{I} and the edge attributes \mathbf{Y} . The \mathbf{X} that is given as input is the concatenation of the padded node features matrices of the action and NFT nodes. As they have differing dimensions, this is an addition to reduce the complexity because this way, the GNN model still behaves as a homogenous model instead of the complexity of a heterogenous model. The edge attributes are the same for both relations so there was no need for an adjustment in that regard. The input layer consists of the following:

$$\mathbf{X} \in \mathbb{R}^{N \times d_{\text{in_node}}}, \quad \mathbf{I} \in \mathbb{R}^{2 \times M}, \quad \mathbf{Y} \in \mathbb{R}^{M \times d_{\text{in_edge}}}$$

Normalization Layer. Node features are standardised using layer normalisation [24] as a pre-processing step, which was added to improve the performance of the training:

$$\mathbf{X}_{\text{norm}} = \text{LayerNorm}(\mathbf{X}) \in \mathbb{R}^{N \times d_{\text{in_node}}}$$

Concatenation Layer: Because the edge attributes are also required to be considered when generating the subgraph embeddings, they are concatenated to the node features. Scatter Mean is applied to calculate the average of the edge attributes and that is then concatenated to the node features. This way all graph information is passed down through the other layers. The applied Scatter Mean is described as follows:

$$\mathbf{I}_{\text{edge}} = \text{scatter_mean}(\mathbf{Y}, \mathbf{I}[0], \text{dim} = 0, \text{dim_size} = N) \in \mathbb{R}^{N \times d_{\text{in_edge}}}$$

The concatenated node features \mathbf{X}_{cat} that is passed down to the GraphSAGE layer is explained as follows:

$$\mathbf{X}_{\text{cat}} = \text{concat}(\mathbf{X}_{\text{norm}}, \mathbf{I}_{\text{edge}}, \text{dim} = 1) \in \mathbb{R}^{N \times (d_{\text{in_node}} + d_{\text{in_edge}})}$$

GraphSAGE: A GraphSAGE [25] operator⁵ is utilised to perform the neighbourhood aggregation where every node aggregates features from its neighbouring nodes:

$$\mathbf{H} = \text{SAGEConv}(\mathbf{X}_{\text{cat}}, \mathbf{I}) \in \mathbb{R}^{N \times d_{\text{hidden}}}$$

⁵<https://bitly.cx/oo1b>

⁴0x00

ReLU Activation: The previously defined hidden features are passed through a ReLU activation function:

$$\mathbf{H}_{\text{relu}} = \text{ReLU}(\mathbf{H}) \in \mathbb{R}^{N \times d_{\text{hidden}}}$$

Linear Transformation: A linear transformation is applied to the activated features to put them into the required shape for the next step:

$$\mathbf{H}_{\text{linear}} = \mathbf{H}_{\text{relu}} \mathbf{W}_{\text{linear}} + \mathbf{b}_{\text{linear}} \in \mathbb{R}^{N \times d_{\text{out}}}$$

Global Mean Pooling: Since the subgraph embeddings from both the training and the testing of this model are required to obtain, global mean pooling was applied to obtain graph-level embeddings:

$$\mathbf{Z} = \text{global_mean_pool}(\mathbf{H}_{\text{linear}}, \mathbf{B}) \in \mathbb{R}^{G \times d_{\text{out}}}$$

Output Layer: Finally, the model returns the total graph-level embeddings for the input \mathbf{S}' :

$$\mathbf{Z} \in \mathbb{R}^{G \times d_{\text{out}}}$$

3.4.3. Embedding and clustering of user-flow graphs

The user (sub)graphs were partitioned into a training and testing dataset. 501 user addresses were chosen, and their corresponding graph representations were constructed and then added to the training dataset. The graph representations were calculated for the remaining 215 users as well and then placed in the testing dataset. The training dataset was used to train the GNN model, and that was validated on the testing dataset by graph embedding similarity. Each unique subgraph embedding was extracted and grouped into one array, which was given as input for the clustering algorithms.

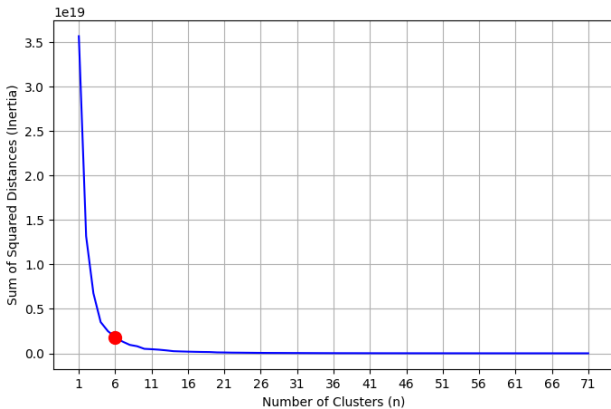


Figure 5: Elbow method for optimal n .

Without pre-defined knowledge, we also did not have any indicators of the suitable clustering algorithm that should be applied. Therefore, clustering was performed by leveraging multiple clustering algorithms, ranging from widely used to state-of-the-art, and comparing their results. The applied algorithms are as follows: k -means [26], mean-shift [27], spectral clustering [28], agglomerative clustering [29], BIRCH

Table 2

Comparison of clustering algorithms.

Algorithm	SC	DBI	CHI
k -means	0.6122	0.4605	2757.17
Mean-shift	0.5592	0.5456	987.51
Spectral clustering	0.1463	2.8559	605.45
Agglomerative clustering	0.5273	0.5712	2030.50
BIRCH	0.5447	0.4321	2232.54
Bisecting k -means	0.5685	0.4670	2412.91
Affinity propagation	-0.6096	646.2999	1.65

[30], bisecting k -means [31] and affinity propagation [32]. We have also tried DBSCAN [33] and HDBSCAN [34], but they did not yield comparable results, so we excluded them from our experiment tables.

Prior to this clustering process, we did not have any predefined knowledge of potential user behaviour or any grand truth labels that we could use. Therefore, unsupervised clustering was conducted to place the unique subgraph embeddings into multiple diverse clusters. We also did not have any indication of the potentially required number of clusters (n), so the elbow method was leveraged. Based on Figure 5, 6 was chosen as the number of clusters for those clustering algorithms that require such an input.

4. Results and discussion

In this section, the results of the clustering process are presented. At first, a comparison of the clustering algorithms is provided. Based on that, one of the clustering methods was chosen, and behavioural clusters that were produced by this algorithm were presented. For each cluster, descriptions were provided based on multiple aspects and some visualisation examples are also presented.

4.1. Comparison of the clustering algorithms

Multiple clustering metrics were leveraged: the Silhouette Coefficient (SC) [35], the Davies-Bouldin Index (DBI) [36] and the Calinski-Harabasz Index (CHI) [37], to evaluate the performance of the various clustering algorithms. SC gives a value between -1 and 1 and the values closer to 1 present better clustering results. DBI ranges between 0 and ∞ and shows good clustering results when the value is close to 0. In CHI there is no specific range but higher values mean good clustering and lower values present poor results. The results are presented in Table 2. They show that k -means performed the best based on SC and CHI and BIRCH outperformed all based on DBI. However, agglomerative clustering and bisecting k -means produced very similar values as well which present a good performance so the usage of any of those is recommended. On the other hand, spectral clustering and affinity propagation presented poor results. We chose k -means to present clustering results but any of the recommended clustering algorithms can be utilised in related research.

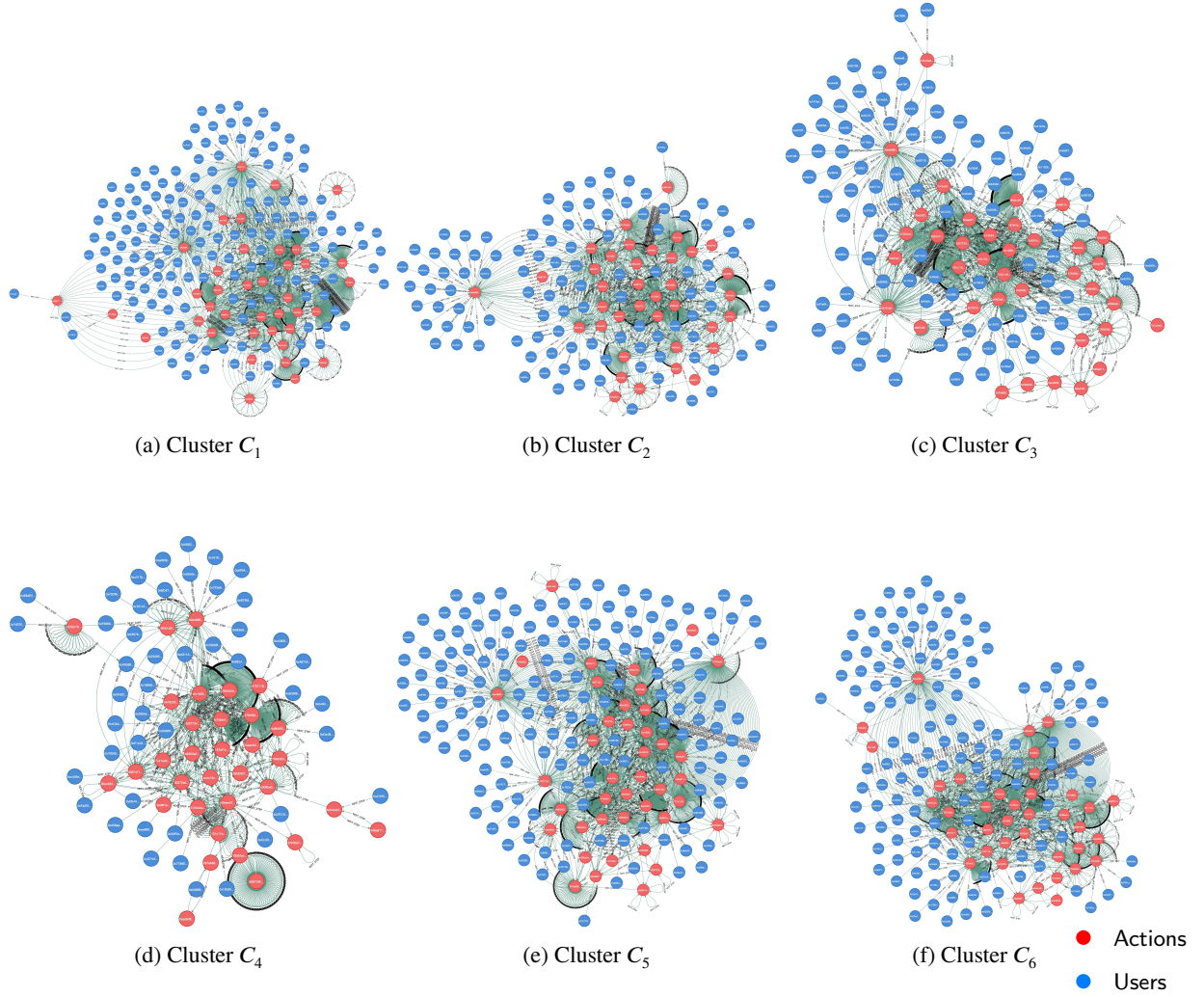


Figure 6: Behavioural clusters from k -means.

4.2. Behavioural clusters

In order to describe the different behavioural clusters that were constructed by k -means, certain information was extracted as a base for their comparison. This can be seen in Table 3. Multiple features are presented for all 6 clusters, they are named as follows in left to right order: the number of users in the cluster, the mean number of assets, tickets and packs that were used in any action, the length of the general user flow, the number of unique NFTs that the corresponding users utilised, the average time (in hours) the users spent in the game during the examined time period, two of our introduced metrics, asset usage (ρ) and NFT distribution (Φ) that we introduce later and the action sequences for the general user flows. Those present how the actions follow each other in the corresponding general user flows. In order to show visualisation results, the clustering algorithm results were added as edge properties, which enables us to filter based on the clustering algorithm and the cluster number. The clusters generated by k -means are visualised in Figure 6. This presents how all user flow subgraphs of users of the clusters are merged together to a new subgraph of G .

For better visibility, a general user flow was also generated for each cluster. For every possible step (based on the average amount of steps the included users made within the examined period) in that cluster, a check was made for which action was the mode action to determine which one was the most frequently performed by the users of that cluster at the specific step, and that was added as the action for that specific step. In order to visualise this, a virtual user node was added as the starting user for all general user flows and the edges in between nodes were added as a new type of edge called VIRTUAL_STEP which has the k -means cluster number as property. In Figure 7, the general user flows for the behavioural clusters are presented. However, these graphs also include all the NFT assets that were used by the actions of the general user flows. In the following, all produced clusters are described, which we summarise in Table 4. In both Table 3 and Table 4, # refers to the cluster number.

In the summary Table 4, the asset usage is calculated by two metrics: ρ and Φ . ρ refers to high/low asset usage by calculating the ratio of the actions that have any type of asset usage as shown in Eq 1, while Φ presents whether these NFT assets are well distributed among the actions by

Table 3Comparison of k -means clusters.

#	Users	Mean# asset	Mean# ticket	Mean# pack	Flow length	NFTs	Time (hr)	Usage (ρ)	Distr. (Φ)	Typical action sequence
1	162	1.5578	0.0491	0.1639	11	441	52.19	0.50	2	$a_0, a_1, a_2, a_3, a_0, a_4, a_5, a_6, a_7, a_8, a_9$
2	111	1.6725	0.0508	0.1525	11	574	64.53	0.40	2	$a_0, a_0, a_5, a_7, a_{10}, a_{11}, a_{12}, a_{13}, a_6, a_{14}, a_9$
3	100	1.4878	0.0508	0.1525	11	188	69.62	0.54	2	$a_0, a_1, a_5, a_{15}, a_7, a_9, a_{16}, a_{17}, a_{12}, a_6, a_{18}$
4	46	1.6011	0.0625	0.1875	15	820	247.20	0.50	3	$a_0, a_{19}, a_0, a_{15}, a_{20}, a_{10}, a_{21}, a_{16}, a_{22}, a_6, a_{23}, a_{18}, a_{24}, a_{25}, a_{26}$
5	161	1.2900	0.0769	0.1230	9	282	41.61	0.33	1	$a_0, a_{27}, a_5, a_{15}, a_{17}, a_{10}, a_{16}, a_7, a_3$
6	136	1.2849	0.0517	0.1034	9	280	69.36	0.62	1	$a_0, a_{16}, a_5, a_2, a_{10}, a_{16}, a_6, a_{22}, a_{28}$

Table 4

Summary of cluster descriptions.

#	Active	Asset usage	Time
1	No	High, not distributed	Low
2	No	Low, not distributed	Medium
3	Semi-active	High, not distributed	Medium
4	Yes	High, distributed	Long
5	No	Low, not distributed	Short
6	No	High, not distributed	Medium

checking the number of nodes that have the majority of the USED_BY relations. This is detailed in Eq 2. In both cases, if a node has multiple USED_BY relations to a single NFT, we only counted one relation. Asset usage is high if $\rho \geq \alpha$ where α is the usability factor which determines the ratio of the action nodes that must have NFTs associated with them. NFTs are well distributed if the number of nodes, that have the majority ratio of the USED_BY relations where this ratio is determined by the distribution factor β , are higher than the distribution number of nodes that are represented by γ . They are calculated as follows:

$$\rho = \frac{\sum_{i=1}^m x_i \cdot \mathbf{I}(x_i > 0)}{m} \quad (1)$$

where $\mathbf{I}(x_i > 0)$ is the indicator function that equals 1 if $x_i > 0$ and 0 otherwise, where x_i is the number of USED_BY relations associated with a particular node \mathbf{n} and m is the number of nodes in the general user flow.

$$\Phi = \min \left\{ k \mid \sum_{i=1}^k x_i \geq \beta \cdot \sum_{i=1}^m x_i \right\} \quad (2)$$

where x_i represents the number of USED_BY relations for each node, m is the total number of nodes in the general user flow and k is the minimum number of nodes needed to reach or exceed the set threshold.

Based on the examination of the general user flows visualised in Figure 7, we chose 0.5 for α , 0.6 for β and 2 for γ . In Table 3, values for both metrics can be seen.

Table 4 also presents information regarding user activity. When determining whether users in a cluster were active in the game or not, the following was considered:

- Distributed NFT usage which suggests that they have participated in multiple parts of the game and not just focused on one aspect.
- The time the users spent in the game and the amount of actions they performed within the examined period.
- Whether there is any element of the game that influences the users' performed actions.

The behavioural clusters generated by k -means are described as follows in the order of the ascending engagement level of the included users:

Cluster C_5 - Inactive Users with interest: The short general user flow visualised in Figure 7e, the low asset usage, mean asset and pack values and the lowest associated time period shows that these users were highly inactive within the period. However, the mean ticket value is the highest among all clusters which means that lottery-related activities may interest them and can be a key element in increasing potential engagement. As a result of this the users who belong to this cluster can be called inactive users with interested.

Cluster C_6 - Inactive Users with no interest: Similar behaviour to Cluster C_5 but with an increased amount of time spent in the application. The overall asset usage is higher but it is extremely low in all actions but one that has the majority of the involved assets as presented in Figure 7f. There is no clear indication what could increase the users' activity level, therefore, further behavioural analysis can be crucial for the developers to gain insights on the users' potential interests and habits that can be utilised to make changes in the application that can be beneficial for them. As there is no defined interest, users of this cluster are named inactive users with no interest.

Cluster C_2 - Brief Engagers: Users in this cluster can be considered not active because although they performed multiple types of actions within a time period that is considered average, they did not use NFTs in a lot of actions. The involved assets are crucial parts of this application thus low usage presents low user participation. Figure 7b presents this low asset usage. Application providers can also leverage analysis of this cluster to determine causes of low engagement after trial. The included users can be called brief

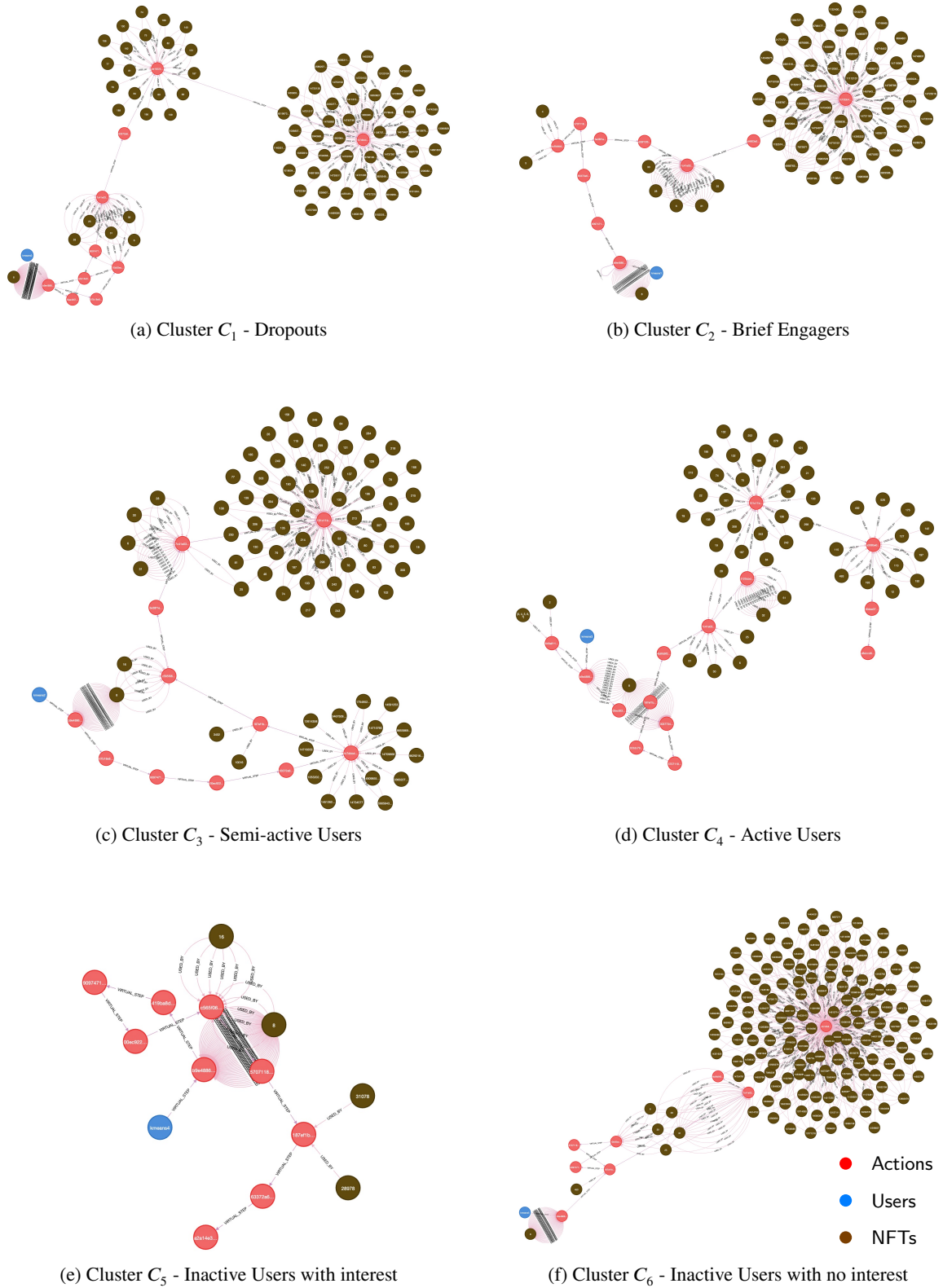


Figure 7: Generated general user flows.

engages as they try the application but do not get involved in all parts.

Cluster C_1 - Dropouts: This cluster presents a common behaviour where the included users conducted various types of activities within a short time with high asset usage but they did not get invested in the application. Figure 7a

presents a not well distributed asset usage as the majority of the assets are centered around only two action nodes. NFT distribution is important as the application consists of multiple activities offered by in-game corporations and some of those involve NFTs and others are being performed without the need for any asset usage. If the NFTs are well distributed among actions, it can be assumed that the corresponding users are invested in multiple parts of the application. Analysis of this behavioural cluster can provide information on why certain users did not get engaged in the application after conducting various types of activities in it. These users are called dropouts.

Cluster C_3 - Semi-active Users: Users of this cluster performed multiple actions with high asset usage within an average time period but with a poor distribution as presented in Figure 7c. They can be considered semi-active users as they engaged in the game but the distribution presents that they did not get fully involved even though they spent more time in the application than users of Cluster C_2 . Analysis of this cluster can help developers to adopt strategies that can potentially increase the corresponding users' engagement.

Cluster C_4 - Active Users: This cluster groups the users that presented active user behaviour within the examined period. Figure 7d presents the longest general user flow that shows high asset usage with good distribution. The values for mean asset, ticket and pack also present that they utilised the game assets when participating in multiple types of in-game activities. This cluster has the longest time period which means that these users not only engaged but presented continuous active behaviour. However, this cluster has the lowest associated user base thereby, it can be stated that active user behaviour is rare in this dApp. Behavioural analysis of the involved users can enable the establishment of engagement models of active user behaviour which can be utilised by the game providers to check which parts of the application boost active engagement.

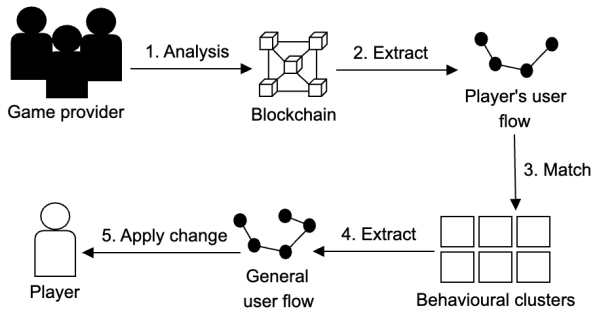


Figure 8: Privacy threat model.

4.3. Privacy threat model

The modelled version of a potential privacy threat is presented in this sub-section, which can be seen in Figure 8. Game providers can conduct similar types of blockchain analysis of their games, establishing behavioural clusters.

If a game provider would like to deduce what to add or remove in the game in order to increase the engagement of a certain player, the provider can extract the player's corresponding user flow. Then that can be matched to the behavioural clusters' general user flows. From the cluster that matches, the game provider can extract the general user flow, which shows the usual activities of the belonging users and other in-game behavioural information in regard to how the corresponding users usually behave. The provider then can form and apply a change based on that which specifically targets the specific player. In the following, a couple of affected areas are presented:

GameFi: This threat model presents that this type of research has a use case in the GameFi sector. This is further supported by the examples given in the previous sub-section. For example, behavioural information from an active cluster can be utilised to reveal which activities boost interest and information extracted from lesser active clusters can be leveraged to conduct churn rate-related analysis. Results from these analyses can enable game providers to design more effective engagement and incentive models.

Secondary NFT marketplace: Users can buy assets in the primary NFT marketplace that is associated with the particular dApp. However, because of full ownership of assets on the blockchain, these users can sell these assets on other platforms that are not related to the original dApp as well. We refer to these as secondary NFT marketplaces. Analysis of the primary marketplace, secondary marketplace or both can be also beneficial if a malicious actor would like to learn more about a certain user's behaviour. Therefore, the pipeline can be utilised in this area as well. Data from the smart contracts that are associated with the marketplaces can be collected and behavioural clusters can be established based on it. Targeted users' user flows can then be matched to the general user flows of the clusters and information from the matched cluster can be utilised against the targeted user. For example, it can be learned whether he/she is willing to participate in an auction or not.

Metaverse: In a metaverse setting malicious attackers can utilise similar techniques to construct social profiles of users based on their activities and interactions. They can extract the users' user flows and match them to general user flows thereby, gaining knowledge of common activities and habits that are present in the matching cluster. These profiles can be leveraged to create fake avatars, which can be utilised to impersonate the original users or perform scams to a targeted user base (users within a certain cluster), similarly to how they are presented in [38]. To present an example, they can check which cluster includes the most active users and target them with scams, as they are more likely to respond because of their higher participation rate. As a result of the possibility of establishing behavioural clusters or user profiles, blockchain applications in privacy-sensitive areas such as healthcare or governance could face similar problems.

5. Conclusion

Public blockchains have the potential to be used in multiple application areas. However, the traceability of transaction data by anyone presents a privacy challenge as blockchain data analysis can potentially be used to extract user behaviour. This work presented a process to extract and define user behaviour clusters based on blockchain transaction data that was collected from a chosen dApp which is a blockchain-based game. From the data collection, unique actions were identified and leveraged to present user action flows that described the user's activities and asset usage. These user flows were utilised as input to a GNN model to provide embeddings, which were then used in clustering algorithms to put them into diverse clusters. These clusters can be visualised and analysed to discuss the related user behaviour. The first resulting behavioural clusters present multiple information regarding the participating users like their activity level and asset usage.

In future work, the time period provided for the user flows will be increased so long-term user behaviour can also be analysed. This means a wider time period that corresponds to the current timestamps and also the examination of user behaviour from a time period that is very distant from the current timestamps. We assume that the analysis between multiple periods may provide more expressive results. Currently, we only consider data from game-related smart contracts, but combining that with transaction information from secondary NFT markets and market analysis may extend information presented in the form of behavioural clusters. Data from other blockchain-based applications DeFi services will be collected and it will be investigated whether there are general behavioural patterns across multiple dApps.

References

- [1] N. Deepa, Q.-V. Pham, D. C. Nguyen, S. Bhattacharya, B. Prabadevi, T. R. Gadekallu, P. K. R. Maddikunta, F. Fang, P. N. Pathirana, A survey on blockchain for big data: Approaches, opportunities, and future directions, *Future Generation Computer Systems* 131 (2022) 209–226. doi:<https://doi.org/10.1016/j.future.2022.01.017>. URL <https://www.sciencedirect.com/science/article/pii/S0167739X22000243>
- [2] X. Yang, W. Li, A zero-knowledge-proof-based digital identity management scheme in blockchain, *Computers & Security* 99 (2020) 102050. doi:<https://doi.org/10.1016/j.cose.2020.102050>. URL <https://www.sciencedirect.com/science/article/pii/S0167404820303230>
- [3] T. Nóbrega, C. E. S. Pires, D. C. Nascimento, Blockchain-based privacy-preserving record linkage: enhancing data privacy in an untrusted environment, *Information Systems* 102 (2021) 101826. doi:<https://doi.org/10.1016/j.is.2021.101826>. URL <https://www.sciencedirect.com/science/article/pii/S0306437921000661>
- [4] T. Huynh-The, T. R. Gadekallu, W. Wang, G. Yenduri, P. Ranaweera, Q.-V. Pham, D. B. da Costa, M. Liyanage, Blockchain for the metaverse: A review, *Future Generation Computer Systems* 143 (2023) 401–419. doi:<https://doi.org/10.1016/j.future.2023.02.008>. URL <https://www.sciencedirect.com/science/article/pii/S0167739X23000493>
- [5] L. Kugler, Non-fungible tokens and the future of art, *Commun. ACM* 64 (9) (2021) 19–20. doi:[10.1145/3474355](https://doi.org/10.1145/3474355). URL <https://doi.org/10.1145/3474355>
- [6] R. Zhang, R. Xue, L. Liu, Security and privacy on blockchain, *ACM Comput. Surv.* 52 (3) (jul 2019). doi:[10.1145/3316481](https://doi.org/10.1145/3316481). URL <https://doi.org/10.1145/3316481>
- [7] B. Tao, H.-N. Dai, J. Wu, I. W.-H. Ho, Z. Zheng, C. F. Cheang, Complex network analysis of the bitcoin transaction network, *IEEE Transactions on Circuits and Systems II: Express Briefs* 69 (3) (2022) 1009–1013. doi:[10.1109/TCSII.2021.3127952](https://doi.org/10.1109/TCSII.2021.3127952).
- [8] K. Pelechris, X. Liu, P. Krishnamurthy, A. Babay, Spotting anomalous trades in nft markets: The case of nba topshot, *PLOS ONE* 18 (6) (2023) 1–17. doi:[10.1371/journal.pone.0287262](https://doi.org/10.1371/journal.pone.0287262). URL <https://doi.org/10.1371/journal.pone.0287262>
- [9] M. Ghosh, D. Ghosh, R. Halder, J. Chandra, Investigating the impact of structural and temporal behaviors in ethereum phishing users detection, *Blockchain: Research and Applications* 4 (4) (2023) 100153. doi:<https://doi.org/10.1016/j.bcr.2023.100153>. URL <https://www.sciencedirect.com/science/article/pii/S2096720923000283>
- [10] Y. Chen, H. Chen, Y. Zhang, M. Han, M. Siddula, Z. Cai, A survey on blockchain systems: Attacks, defenses, and privacy preservation, *High-Confidence Computing* 2 (2) (2022) 100048. doi:<https://doi.org/10.1016/j.hcc.2021.100048>. URL <https://www.sciencedirect.com/science/article/pii/S2667295221000386>
- [11] R. Phillips, H. Wilder, Tracing cryptocurrency scams: Clustering replicated advance-fee and phishing websites, in: *2020 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, 2020, pp. 1–8. doi:[10.1109/ICBC48266.2020.9169433](https://doi.org/10.1109/ICBC48266.2020.9169433).
- [12] J. Hu, K. Huang, G. Bian, Y. Cui, Redact4trace: A solution for auditing the data and tracing the users in the redactable blockchain, *Computer Networks* 245 (2024) 110360. doi:<https://doi.org/10.1016/j.comnet.2024.110360>. URL <https://www.sciencedirect.com/science/article/pii/S1389128624001920>
- [13] Z. Wu, J. Liu, J. Wu, Z. Zheng, Tracer: Scalable graph-based transaction tracing for account-based blockchain trading systems (2022). arXiv:2201.05757.
- [14] G. Bonifazi, E. Corradini, D. Ursino, L. Virgili, Defining user spectra to classify ethereum users based on their behavior, *Journal of Big Data* 9 (04 2022). doi:[10.1186/s40537-022-00586-3](https://doi.org/10.1186/s40537-022-00586-3).
- [15] Y. Jiang, S. Fan, W. Cai, Economic analysis of loot box market in blockchain games, in: *Proceedings of the Fourth ACM International Symposium on Blockchain and Secure Critical Infrastructure, BSCI '22*, Association for Computing Machinery, New York, NY, USA, 2022, p. 35–46. doi:[10.1145/3494106.3528675](https://doi.org/10.1145/3494106.3528675). URL <https://doi-org.libraryproxy.griffith.edu.au/10.1145/3494106.3528675>
- [16] Y. Jiang, T. Min, S. Fan, R. Tao, W. Cai, Towards understanding player behavior in blockchain games: A case study of aavegotchi, in: *Proceedings of the 17th International Conference on the Foundations of Digital Games, FDG '22*, Association for Computing Machinery, New York, NY, USA, 2022. doi:[10.1145/3555858.3555883](https://doi.org/10.1145/3555858.3555883). URL <https://doi-org.libraryproxy.griffith.edu.au/10.1145/3555858.3555883>
- [17] S. Wu, F. Sun, W. Zhang, X. Xie, B. Cui, Graph neural networks in recommender systems: A survey, *ACM Comput. Surv.* 55 (5) (dec 2022). doi:[10.1145/3535101](https://doi.org/10.1145/3535101). URL <https://doi.org/10.1145/3535101>
- [18] M. Jin, H. Y. Koh, Q. Wen, D. Zambon, C. Alippi, G. I. Webb, I. King, S. Pan, A survey on graph neural networks for time series: Forecasting, classification, imputation, and anomaly detection (2023). arXiv:2307.03759.
- [19] J. Liu, J. Zheng, J. Wu, Z. Zheng, Fa-gnn: Filter and augment graph neural networks for account classification in ethereum, *IEEE Transactions on Network Science and Engineering* 9 (4) (2022) 2579–2588. doi:[10.1109/TNSE.2022.3166655](https://doi.org/10.1109/TNSE.2022.3166655).
- [20] E. Alsentzer, S. Finlayson, M. Li, M. Zitnik, Subgraph neural networks, in: H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, H. Lin

- (Eds.), *Advances in Neural Information Processing Systems*, Vol. 33, Curran Associates, Inc., 2020, pp. 8017–8029.
URL https://proceedings.neurips.cc/paper_files/paper/2020/file/5bca8566db79f3788be9efd96c9ed70d-Paper.pdf
- [21] R. L. Thorndike, Who belongs in the family?, *Psychometrika* 18 (4) (1953) 267–276.
- [22] D. Zelenyanszki, Z. Hóu, K. Biswas, V. Muthukkumarasamy, Linking nft transaction events to identify privacy risks, in: N. Dong, B. Pillai, G. Bai, M. Utting (Eds.), *Distributed Ledger Technology*, Springer Nature Singapore, Singapore, 2024, pp. 82–97.
- [23] E. Kleinman, S. Ahmad, Z. Teng, A. Bryant, T.-H. D. Nguyen, C. Hartevelde, M. Seif El-Nasr, "and then they died": Using action sequences for data driven, context aware gameplay analysis, in: *Proceedings of the 15th International Conference on the Foundations of Digital Games, FDG '20*, Association for Computing Machinery, New York, NY, USA, 2020. doi:10.1145/3402942.3402962.
URL <https://doi-org.libraryproxy.griffith.edu.au/10.1145/3402942.3402962>
- [24] J. L. Ba, J. R. Kiros, G. E. Hinton, Layer normalization (2016). arXiv:1607.06450.
- [25] W. Hamilton, Z. Ying, J. Leskovec, Inductive representation learning on large graphs, in: I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (Eds.), *Advances in Neural Information Processing Systems*, Vol. 30, Curran Associates, Inc., 2017.
URL https://proceedings.neurips.cc/paper_files/paper/2017/file/5dd9db5e033da9c6fb5ba83c7a7ebee9-Paper.pdf
- [26] J. MacQueen, Some methods for classification and analysis of multivariate observations, in: *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, Vol. 1, Oakland, CA, USA, 1967, pp. 281–297.
- [27] K. Fukunaga, L. Hostetler, The estimation of the gradient of a density function, with applications in pattern recognition, *IEEE Transactions on Information Theory* 21 (1) (1975) 32–40. doi:10.1109/TIT.1975.1055330.
- [28] M. C. Nascimento, A. C. de Carvalho, Spectral methods for graph clustering – a survey, *European Journal of Operational Research* 211 (2) (2011) 221–231. doi:<https://doi.org/10.1016/j.ejor.2010.08.012>.
URL <https://www.sciencedirect.com/science/article/pii/S0377221710005497>
- [29] R. Xu, D. Wunsch, Survey of clustering algorithms, *IEEE Transactions on Neural Networks* 16 (3) (2005) 645–678. doi:10.1109/TNN.2005.845141.
- [30] T. Zhang, R. Ramakrishnan, M. Livny, Birch: an efficient data clustering method for very large databases, in: *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data, SIGMOD '96*, Association for Computing Machinery, New York, NY, USA, 1996, p. 103–114. doi:10.1145/233269.233324.
URL <https://doi.org/10.1145/233269.233324>
- [31] S. M. Savaresi, D. L. Boley, On the performance of bisecting K-means and PDDP, pp. 1–14. arXiv:<https://epubs.siam.org/doi/pdf/10.1137/1.9781611972719.5>, doi:10.1137/1.9781611972719.5.
URL <https://epubs.siam.org/doi/abs/10.1137/1.9781611972719.5>
- [32] B. J. Frey, D. Dueck, Clustering by passing messages between data points, *Science* 315 (5814) (2007) 972–976. arXiv:<https://www.science.org/doi/pdf/10.1126/science.1136800>, doi:10.1126/science.1136800.
URL <https://www.science.org/doi/abs/10.1126/science.1136800>
- [33] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise, in: *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, KDD'96*, AAAI Press, 1996, p. 226–231.
- [34] R. J. G. B. Campello, D. Moulavi, J. Sander, Density-based clustering based on hierarchical density estimates, in: J. Pei, V. S. Tseng, L. Cao, H. Motoda, G. Xu (Eds.), *Advances in Knowledge Discovery and Data Mining*, Springer Berlin Heidelberg, Berlin, Heidelberg, 2013, pp. 160–172.
- [35] P. J. Rousseeuw, Silhouettes: A graphical aid to the interpretation and validation of cluster analysis, *Journal of Computational and Applied Mathematics* 20 (1987) 53–65. doi:[https://doi.org/10.1016/0377-0427\(87\)90125-7](https://doi.org/10.1016/0377-0427(87)90125-7).
URL <https://www.sciencedirect.com/science/article/pii/0377042787901257>
- [36] D. L. Davies, D. W. Bouldin, A cluster separation measure, *IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI-1* (2) (1979) 224–227. doi:10.1109/TPAMI.1979.4766909.
- [37] T. Caliński, J. Harabasz, A dendrite method for cluster analysis, *Communications in Statistics* 3 (1) (1974) 1–27. arXiv:<https://www.tandfonline.com/doi/pdf/10.1080/03610927408827101>, doi:10.1080/03610927408827101.
URL <https://www.tandfonline.com/doi/abs/10.1080/03610927408827101>
- [38] Y. Wang, Z. Su, N. Zhang, R. Xing, D. Liu, T. H. Luan, X. Shen, A survey on metaverse: Fundamentals, security, and privacy, *IEEE Communications Surveys & Tutorials* 25 (1) (2023) 319–352. doi:10.1109/COMST.2022.3202047.