Toward Intention Discovery for Early Malice Detection in Cryptocurrency

1st Ling Cheng
Singapore Management University
Singapore
lingcheng.2020@phdcs.smu.edu.sg

2nd Feida Zhu
Singapore Management University
Singapore
fdzhu@smu.edu.sg

3rd Yong Wang
Singapore Management University
Singapore
yongwang@smu.edu.sg

4th Ruicheng Liang
Hefei University of Technology
Hefei, China
rcliang@mail.hfut.edu.cn

5th Huiwen Liu
Singapore Management University
Singapore
hwliu.2018@phdcs.smu.edu.sg

***Abstract*—Cryptocurrency's pseudo-anonymous nature makes it vulnerable to malicious activities. However, existing deep learning solutions lack interpretability and only support retrospective analysis of specific malice types. To address these challenges, we propose Intention-Monitor for early malice detection in Bitcoin. Our model, utilizing Decision-Tree based feature Selection and Complement (DT-SC), builds different feature sets for different malice types. The Status Proposal Module (SPM) and hierarchical self-attention predictor provide real-time global status and address label predictions. A survival module determines the stopping point and proposes the status sequence (intention). Our model detects various malicious activities with strong interpretability, outperforming state-of-the-art methods in extensive experiments on three real-world datasets. It also explains existing malicious patterns and identifies new suspicious characteristics through additional case studies.**

***Index Terms*—Malicious Address, Cybercrime, Early Detection, Intention-aware, Cryptocurrency, Bitcoin**

I. Introduction

The growing popularity of cryptocurrency has led to a rise in cybercrimes, such as hacking, extortion, and money laundering. These criminal activities are often carried out by individuals or entities referred to as "malicious addresses" in the cryptocurrency realm [1], [2]. Detecting and diagnosing these malicious addresses pose significant challenges, especially in the case of Bitcoin, the most prominent cryptocurrency.

There are three key characteristics of cryptocurrencies that make the detection of malicious behavior more challenging compared to traditional financial fraud detection: **Early detection** is crucial due to the continuous and fast-paced nature of cryptocurrency trading. Malicious behaviors are often short-lived and can cause substantial damage if not identified early. Retrospective analysis offers limited value as it cannot prevent financial losses once the malicious activity is completed. **Manually-engineered features** are insufficient for detecting unknown types of malicious behavior or applying them to different cryptocurrencies. Malicious behaviors in cryptocurrencies constantly evolve, requiring a more versatile set of features that capture fundamental characteristics across various types. **Interpretability** is essential in detecting malicious behavior in cryptocurrency. Malicious activities often disguise themselves as legitimate projects, making it challenging for investors to differentiate between credible and fraudulent ventures. Deep learning methods used for detecting malicious activity often lack interpretability, which is crucial for accurately identifying such behavior. Investors and regulators require deeper insights into the underlying intention behind malicious behavior.

To address these challenges, we propose a system called Intention Monitor, which utilizes asset transition paths to identify patterns indicating malicious intent. Our system analyzes significant asset transitions between innocent and malicious addresses to detect malicious behavior across different types of cryptocurrencies. The proposed solution consists of four stages: 1. Feature formation: Long-Term (LT) and Short-Term (ST) transition paths are generated to capture transaction patterns for both long-term and short-term structures. 2. Feature filtering and temporal assembly: A Decision Tree-based Feature Selection and Complement model (DT-SC) identifies the most powerful features for different types of malicious behavior. The Status Proposal Module (SPM) organizes these features into coherent segments based on their importance scores. 3. Semantic mapping: Temporal feature segments are mapped to global statuses through clustering, representing the intentions of malicious behaviors after temporal grouping by survival analysis. 4. Intention motif as prediction witness: A hierarchical transformer with a survival module predicts malicious addresses in real-time. The survival module segments the status sequence into intention motifs, which serve as a witness to the prediction result.

In summary, this paper's key contributions are as follows: We propose two novel definitions of asset transfer paths that effectively capture Bitcoin transaction patterns for early malice detection and can be applied to other cryptocurrencies, enabling the model's versatility across different types of malicious behavior. We provide interpretability for our malice detection results using intention motifs as prediction witnesses, which is not achievable by deep-learning-dominated models. Interpretability is facilitated through a decision tree-based strategy for feature selection and assembly, as well as a hierarchical transformer encoder with a survival module for grouping statuses into a sequence of intention motifs. We conduct extensive evaluations and achieve significantly better performance than the state-of-the-art on three malicious datasets. Additionally, we present a deep-dive case study on the 2017 Binance hack incident, illustrating corroborating transaction patterns and uncovering hidden insights for early-stage malice detection that would otherwise be unattainable.

II. Related Work

Existing malicious address detection methods can be categorized into three groups based on the types of features utilized:

Case-Related features: These features focus on modeling addresses and activities in specific events. Examples include combining topological structures with IP addresses to investigate theft cases [3], using transaction-graph annotation systems to extract information from social media [4], and linking users to hidden services with social media [5]. While Case-Related features provide valuable insights in case studies, their generalizability to other issues is limited.

General address features: Machine learning techniques are commonly employed in this category for malicious activity detection [6]. Examples include using time and transaction value to reveal address identities [7], performing temporal analysis to identify repeating patterns [8], detecting Ponzi schemes on ETH using address features and operation codes [9] [10], and classifying entities involved in cybercriminal activities [11]. General address features significantly enhance the model's generality, although characterizing asset inflow and outflow remains challenging.

Network-based features: Cryptocurrencies inherently provide transaction networks between addresses. Network-based methods utilize these networks for detection purposes. Examples include detecting suspicious users using power degree laws and local outlier factor [12], analyzing motifs in directed hypergraphs [13], and detecting BTC mixing service addresses using temporal motifs [14]. Network-based methods perform well for retrospective analysis, leveraging structural information encoded in the trading network. However, in the early stages, when the trading network is small, discriminative topological structures may not be formed.

In summary, each category has its strengths and limitations in terms of interpretability, generalizability, and applicability to early-stage detection.

III. Problem Formulation

A BTC transaction tx has a set of inputs $I = \{i_1, i_2, \dots i_{|I|}\}$ and outputs $J = \{j_1, j_2, \dots j_{|J|}\}$. The essences of input and output are still transactions. tx records token distribution between I and J. The incoming tokens will flow into a pool and then to the outgoing transactions according to the prior agreement proportion. There is no record of how many tokens flow from an Input i to an Output j. Thus, we have to build a complete transaction bipartite graph in tx and get total $|I| \times |J|$ transaction pairs. Let $D_{t_m} = \{d^i_{t_m}\}^N_{i=1} = \{(l^i, T^i_{in,t_m}, T^i_{out,t_m})\}^N_{i=1}$, where $l^i \in \{0, 1\}$ is the label of Address i, and 0 or 1 stands for the label of regular and malicious. $T^i_{in,t_m} = [tx^i_{in,1}, tx^i_{in,2}, \dots tx^i_{N_{in,t_m}}]$ is the set of all transactions with Address i as the input address by the t_m-th time step, and T^i_{out,t_m} is the set of all transactions with Address i as the output address by the t_m-th time step. For ease of understanding, we denote these two transaction sets as $Receive$ set and $Spend$ set, respectively. Given a set of addresses A, and D_{t_m} at t_m-th time-step, the problem is to find a binary classifier F such that

$$F(d^i_{t_m}) = \begin{cases} 1 & \text{if Address } i \text{ is illicit} \\ 0 & \text{Otherwise} \end{cases}. \tag{1}$$

IV. Asset Transfer Path

In the early stages of malicious behaviors, the address-based network may not have grown to a size that allows for credible predictions. Instead, the transaction flow itself can provide crucial information. As mentioned in Sec. III, each BTC transaction consists of $|I| \times |J|$ transaction pairs. However, not all transaction pairs are relevant for malicious address detection. The significant transactions, which make up a considerable portion of the total transaction amount, play a crucial role. Fig. 1 illustrates the concept of asset transition paths. Each node represents a transaction, and the labels LT, ST, FR, and BK represent Long-Term, Short-Term, Forward, and Backward, respectively. The transaction nodes within the dashed gray box represent transactions in which the given address is involved. In this example, the left node represents a transaction where the address receives tokens from other transactions, referred to as the address's "Receive Transaction." In this specific transaction, there are three inputs contributing 5%, 70%, and 25%, respectively, to the total transaction amount. Similarly, the right node represents a transaction where the address spends its tokens to the outputs, known as the address's *Spend Transaction*.

A. Influence and Trust Transaction Pair

Let $I = i_1, i_2, \dots, i_{|I|}$ be a set of $|I|$ spend transactions to a receive transaction j. We define an *Influence Transaction Pair* as follows: given an influence activation threshold θ, (i_k, j) is called an **Influence Transaction Pair** for transaction j if there exists a k ($1 \leq k \leq |I|$) such that the amount of transaction pair (i_k, j) contributes at least a certain proportion of the received amount of transaction j, i.e., $\hat{A}(i_k, j) \geq \theta \times \hat{A}(I \rightarrow j)$, where $\hat{A}(\cdot)$ denotes the amount of a transaction pair or the sum of all transaction pairs. Similarly, let $J = j_1, j_2, \dots, j_{|J|}$ be a set of $|J|$ transactions, and let i be a transaction. We define a **Trust Transaction Pair** for transaction i as follows: if there exists a receive transaction j_k ($1 \leq k \leq |J|$) such that transaction i transfers at least a certain proportion of its spend amount to j_k, the transaction

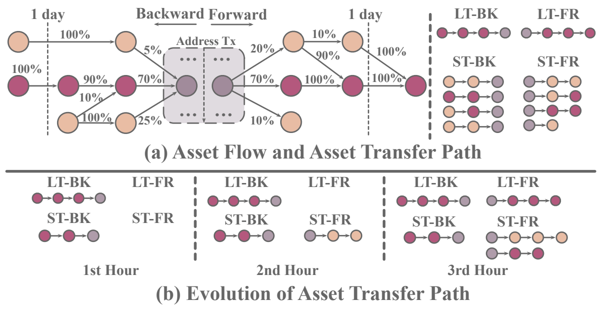

Fig. 1. (a) Address transaction flow and Asset transfer paths. (b) Evolution of Asset Transfer Path. New asset transfer paths are generated if the address participates in new transactions. Forward paths can be extended if asset flows within the late stage.

pair (i, j_k) is called a **Trust Transaction Pair** for transaction i, indicating a form of trust from i to j_k in terms of asset transfer.

Given an influence transaction pair (i_k, j), we can conclude that transaction j obtains a significant amount (based on the threshold) of the asset from transaction i_k. Furthermore, if there exists a sequence of transaction pairs satisfying the following conditions: (I) each pair is an influence transaction pair, (II) the spend transaction of each pair is the receive transaction of the previous pair, and (III) the receive transaction of the last pair is transaction j, we call such a sequence a **Backward Path** for transaction j. It reveals the source of the asset for transaction j. Similarly, we can define a **Forward Path** to trace the destinations of transaction i's asset flow. For simplicity, both the *Backward Path* and *Forward Path* are referred to as *Asset Transfer Paths*, and the activation threshold in both directions is denoted as the *activation threshold*.

B. Long-Term and Short-Term Path

To simplify, we use the terms *Asset Transfer Paths* to refer to both the *Backward Path* (BK) and *Forward Path* (FR), and the term *activation threshold* to denote the threshold in both directions. To capture transaction patterns at both macro and micro levels for the *Asset Transfer Paths*, we introduce two types of time spans: long-term (LT) and short-term (ST). The LT asset transfer paths have a larger maximum observation period and a higher activation threshold. They are designed to identify the primary source of assets in a transaction. On the other hand, the ST asset transfer paths have a shorter observation period and a lower activation threshold. They provide insights into transition patterns and structures within a shorter time frame.

V. Feature Selection and Complement & Status Proposal Module

A. Address & Transaction Features

We utilized 16 address features to capture address behaviors. Additionally, to characterize a specific path set, we selected 13 path features from four perspectives: (1) path number, (2) path length (hop/height), (3) maximum (minimum) input (output) amount (quantity) for each node on the path, and (4) maximum (minimum) activation score of the path. To represent the overall properties, we computed the maximum (max), minimum (min), average (avg), and standard deviation (std) values for each feature, except for the path number. This results in a total of 49 path features for a single path set, hence a total of 196 path features (49 * 4) as an address has four path sets.

B. DT-based feature Selection and Complement

In this module, we have three sets: complement list, reserve list, and deletion list. In the initial round, address features and all path features' mean values are set as seed features. We feed these features into the decision tree model and select the best-performing (the performing score will be elaborated in Sec. VII-B) model from 10 independent training models. We sort the model's feature importance scores and denote the maximum importance score as s_{imp}^M. Given a feature j with an importance score $s_{imp,j}$, if $s_{imp,j} \geq \theta * s_{imp}^M$, we append it into the complement list. θ is the complement threshold. If $0 < s_{imp,j} < \theta * s_{imp}^M$, we append it into the reserve list. If $s_{imp,j}$=0, we append it into the deletion list.

In the second round of training, we first complement the features in the complement list. Here, by complement, we mean including its maximum, minimum, and standard deviation. Then, we append reserve features to the input feature list and delete features in the delete list.

C. Status Proposal Module

To describe an address's behaviors, we propose the Status Proposal Module (SPM). A status is a segment where features should be stable enough.We calculate all addresses' all features' average change ratio and denote the highest change ratio as C^H and the change ratio at the j-th time step as C_j. If $C_j > \theta * C^H$, the former stable segment ends, and we add the corresponding time step j to the splitting time list.

We then define *segments representation* to represent these segments. Due to the feature temporal shifting, we train a decision tree on each segment. For the j-th segment of a single address, we record its beginning and end time points as b_j and e_j, and the corresponding feature sequence as $[f_{b_j}, ..., f_{e_j}]$, where f_i is the feature vector at i-th hour. The feature importance score list of the corresponding decision tree for the j-th segment is $S_{imp,j}$ in which each element stands for the importance score for the corresponding feature. Then address's segment vector of this period is calculated as follows:

$$g^j = S_{imp,j} * \sum_{i=b_j}^{e_j} f_i/(e_j - b_j). \quad (2)$$

we cluster all addresses' all segment representations as the status vectors to obtain status clusters with global semantic meanings. We then build a status decision tree to label the status for the input segment vector. Finally, each address has three vector sequences, namely *features*, *segments*, and *statuses*.

VI. Hierarchical Survival Transformer

A. Hierarchical Transformer Encoder

The decision tree group fails to utilize temporal patterns, and the subsequent redundant noise will make the prediction

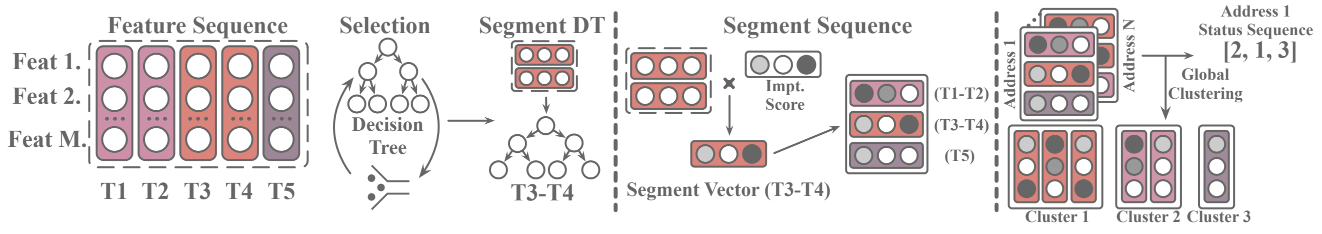

Fig. 2. The overview of Decision Tree based feature Selection and Complement (DT-SC) and Status Proposal Module (SPM).

inconsistent. Thus we propose the Hierarchical Survival Transformer to tackle these issues. For the p-th segment, we denote the beginning and ending time points as b^p and e^p, respectively. In this segment, an address has e^p-b^p+1 features $\{f_i\}_{i=b^p}^{e^p}$ ($f_i \in \mathbb{R}^d$), one segment vector $g^p \in \mathbb{R}^d$ and the index of status o^p, where d is the feature dimension. We transform the status index into a learnable embedding vector via embedding u^p. To select the most representative information inside a segment, we use an attention layer to focus on those significant time steps and prepare the weighted feature vector f^p.

To encode these three representation vectors, we apply a multi-head self-attention encoder with N_h parallel independent heads. Take feature-level self-attention as an example. After multi-head self-attention, the feature list by the p-th segment is F^p can be defined as follows:

$$F^p = \{\hat{f}^i\}_{i=1}^p = Concat(H_1^p, \cdots, H_h^p, \cdots, H_{N_h}^p)W^O. \tag{3}$$

Both the segment and status self-attention encoder are performed in the similar way. the only difference is that we bridge feature, segment and status hierarchically with fully connected layers. $\tilde{g}^i$ and $\tilde{u}^i$ is given by:

$$\begin{aligned} \tilde{g}^i &= W^g \tanh(W^{f,g}[g^i, \hat{f}^i]), \\ \tilde{u}^i &= W^u \tanh(W^{g,u}[u^i, \hat{g}^i]), \end{aligned} \tag{4}$$

where $[\cdot,\cdot]$ stands for concatenation, $\hat{g}^i$ is the segment representation at the i-th split after self-attention. $W^{f,g}, W^{g,u} \in \mathbb{R}^{d\times 2d}$ and $W^g, W^u \in \mathbb{R}^{d\times d}$ are learnable matrices. After encoding all the three levels of representations, the final output vector $\bar{u}^p$ is given by the averaged feature vector of the final self-attended status sequence $U^p = \{\hat{u}^i\}_{i=1}^p$. Finally, based on $\bar{u}^p$, the model gives the final prediction y^p of the p-th splitting period with a learnable fully connected layer.

B. Survival Prediction Analysis

A robust predictor should perform better consistency and robust to noise in the late stage. Thus, we resort survival analysis to boost our hierarchical transformer predictor. As in survival analysis [15], the survival function $S(t)$ of an event represents the probability that this event has not occurred by time t. Here, we define the event as "the model has collected enough information to predict the address label". Since, the observation time is discrete in our case, we use t to denote a timestamp. The association between $S(t)$ and λ_t can be calculated as:

$$S(t) = exp(-\sum_{k=1}^{t}\lambda_k), \quad \lambda_t = ln(1+\exp(W^{hz}\bar{u}^p)), \tag{5}$$

where W^{hz} is the linear projection matrices. A *softplus(x)* $= ln(1+\exp(x))$ function is usually deployed to guarantee the hazard rate is always positive. Thus, the final prediction $\hat{y}^t$ at t-th split is given by:

$$\hat{y}^t = S(t) * y^t + (1 - S(t)) * \hat{y}^{t-1}. \tag{6}$$

The survival analysis can accelerate the prediction and group statuses into an intention sequence. The time step when the survival probability equals 0 is t_{die}, and the status sequence $\{\hat{u}^i\}_{i=1}^{t_{die}}$ as the addresses' intention sequences.

C. Consistent and Early Boost Loss Function

For an address i at t-th segment, model should give the correct prediction and consistent with the previous prediction. the negative logarithm prediction $loss^P$ and the consistency loss $loss^C$ are defined below:

$$\begin{aligned} loss_{i,t}^P &= (l_i - 1) * \log(\hat{y}_i^t) - l_i * \log(1 - \hat{y}_i^t), \\ loss_{i,t}^C &= \begin{cases} 0 & (\hat{y}_i^t - 0.5) * (\hat{y}_i^{t-1} - 0.5) >= 0. \\ 1 & else, \end{cases} \end{aligned} \tag{7}$$

To accelerate prediction speed and ensure early detection, we introduce an earliness loss denoted as $loss^E$. The goal is to minimize the survival probability at each time split. For Address i at Time Split t, the earliness loss $loss_{i,t}^E$ is equal to the survival probability $S_i(t)$. However, predicting the correct labels at the early stage is challenging due to limited data availability. The model can be influenced by incorrect predictions during this period. To mitigate this issue, all loss components are weighted by $\sqrt{t}$, where t represents the time step. The overall loss function is defined as follows, considering N as the number of training samples, and γ_1 and γ_2 as the coefficients used to weight each loss component:

$$L = \sum_{t=1}^{t_M}\sum_{i=1}^{N}\sqrt{t}(loss_{i,t}^P + \gamma_1 loss_{i,t}^C + \gamma_2 loss_{i,t}^E). \tag{8}$$

VII. Experiment and Analysis

A. Data Preparation

Raw Data and Label Collection For higher high credibility, we only select data verified by many participants from the 1-st block to the first block of 2020. To get the labels for three different types of malicious addresses, namely, Hack, Ransomware, and Darknet. We performed a manual search on public forums, datasets, and prior studies [16] and [14]. For regular addresses, we collected four types of addresses as "negative samples", namely Exchange, Mining, Merchant, and Gambling. The negative dataset is also augmented as prior study [14]. The positive smaple number for different malicious types are 341, 1903, and 7696. The negative smaple numbers are 79765, 50617, and 89318. The segment numbers for different malicious types are 27, 34, and 36.

B. Settings and Metrics

We used the first 200 hours of data with a 1 hour interval for training. The max time spans for LT and ST paths are one week and one day, respectively. We set 0.5 and 0.01 as the thresholds for LT and ST paths. We averaged the metrics (accuracy, precision, and recall) on all time steps. And we introduced the early-weighted F1 score $F1^E$ and consistency-weighted F1 score $F1^C$ to evaluate early detection and consistency of the prediction.

$$
\begin{aligned}
F1^E &= \frac{\sum_{i=1}^{N} F1_i/\sqrt{i}}{\sum_{i=1}^{N} 1/\sqrt{i}}, \\
F1^C &= \frac{\sum_{i=1}^{N-1} \sqrt{i} \times F1_i \times \mathbb{1}_{y_c}(y_i)}{\sum_{i=1}^{N-1} \sqrt{i}},
\end{aligned} \tag{9}
$$

where i is the time split index, y_c is the prediction set in which $(y_i - 0.5) * (y_{i+1} - 0.5) > 0$. $\mathbb{1}_{y_c}(y_i) = 1$ when $y_i \in y_c$. $F1_i$ is the $F1$ score at the i-th time split.

TABLE I
SCORES OF DIFFERENT FEATURES (ADDRESS FEATURES (**AF**), LONG-TERM AND SHORT-TERM PATH FEATURES (**+LT/ST**), AND SELECT / COMPLEMENT SCHEME (**S/C**), **+X** MEANS ADD **X** TO PREVIOUS MODEL

	Model	*Acc.*	*Prec.*	*Rec.*	$F1^C$	$F1^E$
H	AF	**0.997**	**0.863**	0.164	0.284	0.246
	+LT/ST	0.996	0.737	0.192	0.303	0.303
	+S/C	**0.997**	0.831	**0.207**	**0.331**	**0.323**
R	AF	0.971	0.457	0.056	0.085	0.076
	+LT/ST	0.973	**0.624**	0.220	0.309	0.290
	+S/C	**0.974**	0.604	**0.290**	**0.387**	**0.349**
D	AF	0.932	0.628	0.292	0.436	0.287
	+LT/ST	**0.933**	**0.639**	0.398	0.497	0.427
	+S/C	**0.933**	0.613	**0.455**	**0.528**	**0.470**

C. Feature Selection and Component Contribution

As shown in Table I, path features **+LT/ST** significantly improve performance, as most malicious addresses require victims to transfer money with particular requirements. Moreover, feature selection can mitigate the input noise and substantially improves $F1^C$ and $F1^E$ scores by an average of 13.6% and 12.3%, respectively. Besides, As shown in Fig. 3, the proportion difference is relatively small for single status (1-Gram), and it becomes bigger when we analyze 2 grams or 3 grams. This means the status sequences can reflect different addresses' intentions. The proportion difference justifies the effectiveness of temporal patterns.

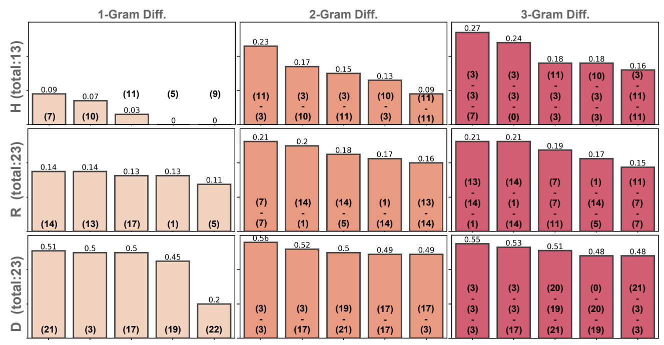

Fig. 3. Top-5 differentiated status n-gram. The value above each bar is the n-gram proportion difference between the two classes.

D. Performance Analysis

As shown in Table II, to demonstrate the validity of the temporal information, we compare **XGB** [17]. Then we compare the address graph-based models, Namely **Evolve-GCN** [18]. To verify the validity of our prediction model, we compare four sequential-based models applied in the "Early Rumor Detection" task, namely **M-LSTM** [19], **SAFE** [15], and **CED** [20]. In addition, we perform a set of ablation studies to verify the modules' effectiveness. **H-T** means we only use the hierarchical transformer encoder. **+Imp** stands for using the importance scores of each segment decision tree. As shown in Table. II, **XGB** does not perform well as they cannot encode temporal information due to distribution shifting. For Address Graph methods, the address **GCN** may lead to Over-Smoothing issues and the dilution of the minority class.For the sequential model, **M-LSTM** have great improvement compared to non-sequential model. However, the **SAFE** model cannot rectify the previous wrong prediction, resulting in lower precision scores. Besides, **CED** models cannot use previous predictions directly and thus cannot guarantee prediction consistency. On the contrary, the **H-T** model achieves an average improvement of 5.00% and 4.92% on $F1^C$ and $F1^E$. Further improvement can be achieved by the survival analysis module and the importance score, compared with the **H-T** model,

E. Dynamically Predict and Scalability

Our system dynamically updates transaction features to predict labels, as shown in Fig. 1(b), and is scalable. We randomly selected 1,000 blocks (from 2020 to 2022), collected daily BTC prices, and filtered transactions and addresses. According to experiments, only 3.7s is required for data preparation during each one-hour interval for each address. Our model is primarily used for predictions within 200 hours, but it can still perform well beyond that if users generate segments based on feature change ratios without importance scores. As shown in Table II, our model still outperforms others.

TABLE II
PERFORMANCE COMPARISON. BOLD STANDS FOR THE BEST SCORE IN ALL GROUPS.

Model	Hack					Ransomware					Darknet				
	$Acc.$	$Prec.$	$Rec.$	$F1^C$	$F1^E$	$Acc.$	$Prec.$	$Rec.$	$F1^C$	$F1^E$	$Acc.$	$Prec.$	$Rec.$	$F1^C$	$F1^E$
XGB	0.996	**0.834**	0.223	0.351	0.349	0.975	0.622	0.342	0.437	0.415	0.940	**0.666**	0.475	0.578	0.502
Evo-GCN	0.760	0.145	0.335	0.196	0.146	0.866	0.200	0.871	0.322	0.326	0.737	0.216	0.822	0.342	0.343
M-LSTM	**0.997**	0.574	0.998	0.725	0.731	0.986	0.716	**1.000**	0.834	0.835	0.907	0.465	0.890	0.575	0.620
SAFE	**0.997**	0.610	**1.000**	0.758	0.757	0.985	0.711	**1.000**	0.831	0.831	0.871	0.363	0.848	0.481	0.512
CED	0.987	0.339	0.695	0.379	0.474	0.986	0.724	0.985	0.827	0.837	0.903	0.453	**0.904**	0.565	0.615
H-T	**0.997**	0.615	**1.000**	0.762	0.762	0.986	0.722	**1.000**	0.838	0.838	0.930	0.540	0.872	0.642	0.674
+Imp.	**0.997**	0.634	**1.000**	**0.776**	**0.776**	**0.988**	**0.747**	**1.000**	**0.855**	**0.855**	**0.941**	0.594	0.833	**0.668**	**0.701**

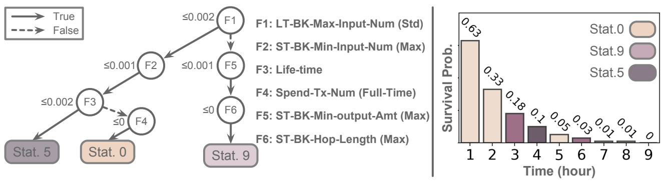

Fig. 4. Sample's status decision tree and survival probability.

F. Case Analysis

Our study utilizes the Binance hack of May 7, 2019[1]. The hacker address(bc1q***v3wm) was detected by our method within the first 9 hours, *12 hours before the stolen bitcoins were transferred*. As shown in Fig. 4, this prediction was based on the status sequence [0-0-9-5-0-9-0-0-0], which indicates that the asset associated with the address came from a single source and there were no spend transactions. **Status 0** essentially indicates (I) The asset associated with the address comes from a single source, and (II) No spend transaction (A small F4 means no transaction). **Status 9** indicated that the asset had been obtained from a single source but through a sequence of transitions. **Status 5** confirmed that there were no spend transactions after receiving the initial asset from a single source. The malicious address received a transfer of 567.997 BTCs at creation through 71 input transitions. By the 13th hour, it received another transfer of about 0.00008642 BTC, followed by a bulk transfer of all its BTC assets at the 21st hour.

VIII. CONCLUSION

This paper introduces a novel framework for early detection of malicious addresses using asset transfer paths, segment and status vectors, and a hierarchical transformer with a survival prediction module. The model's effectiveness was evaluated, and a detailed case study demonstrated its ability to identify suspicious transaction patterns and hidden abnormal signals.

IX. ACKNOWLEDGMENTS

This research is supported by the National Research Foundation, Singapore under its Emerging Areas Research Projects (EARP) Funding Initiative, the NSFC (72171071, 72271084, 72101079) and the Excellent Fund of HFUT (JZ2021HGPA0060). Any opinions, findings and conclusions or recommendations expressed in this material do not reflect the views of National Research Foundation, Singapore.

[1]hack-binance-cryptocurrency-exchange

REFERENCES

[1] H. Chen, M. Pendleton, L. Njilla, and S. Xu, "A survey on ethereum systems security: Vulnerabilities, attacks, and defenses," *CSUR*, 2020.
[2] S. Foley, J. R. Karlsen, and T. J. Putniņš, "Sex, drugs, and bitcoin: How much illegal activity is financed through cryptocurrencies?" *The Review of Financial Studies*, 2019.
[3] F. Reid and M. Harrigan, "An analysis of anonymity in the bitcoin system," in *Security and privacy in social networks*, 2013.
[4] M. Fleder, M. S. Kester, and S. Pillai, "Bitcoin transaction graph analysis," *arXiv preprint arXiv:1502.01657*, 2015.
[5] H. Al Jawaheri, M. Al Sabah, Y. Boshmaf, and A. Erbad, "Deanonymizing tor hidden service users through bitcoin transactions analysis," *Computers & Security*, 2020.
[6] J. V. Monaco, "Identifying bitcoin users by transaction behavior," in *Biometric and surveillance technology for human and activity identification XII*. SPIE, 2015.
[7] E. Androulaki, G. O. Karame, M. Roeschlin, T. Scherer, and S. Capkun, "Evaluating user privacy in bitcoin," in *FC*, 2013.
[8] F. Zola, J. L. Bruse, M. Eguimendia, M. Galar, and R. Orduna Urrutia, "Bitcoin and cybersecurity: Temporal dissection of blockchain data to unveil changes in entity behavioral patterns," *Applied Sciences*, 2019.
[9] W. Chen, Z. Zheng, J. Cui, E. Ngai, P. Zheng, and Y. Zhou, "Detecting ponzi schemes on ethereum: Towards healthier blockchain technology," in *WWW*, 2018.
[10] W. Chen, Z. Zheng, E. C.-H. Ngai, P. Zheng, and Y. Zhou, "Exploiting blockchain data to detect smart ponzi schemes on ethereum," *IEEE Access*, 2019.
[11] H. S. Yin and R. Vatrapu, "A first estimation of the proportion of cybercriminal entities in the bitcoin ecosystem using supervised machine learning," in *IEEE Big Data*, 2017.
[12] T. Pham and S. Lee, "Anomaly detection in the bitcoin system-a network perspective," *arXiv preprint arXiv:1611.03942*, 2016.
[13] S. Ranshous, C. A. Joslyn, S. Kreyling, K. Nowak, N. F. Samatova, C. L. West, and S. Winters, "Exchange pattern mining in the bitcoin transaction directed hypergraph," in *FC*. Springer, 2017.
[14] J. Wu, J. Liu, W. Chen, H. Huang, Z. Zheng, and Y. Zhang, "Detecting mixing services via mining bitcoin transaction network with hybrid motifs," *Transactions On SMC*, 2021.
[15] P. Zheng, S. Yuan, and X. Wu, "Safe: A neural survival analysis model for fraud early detection," in *AAAI*, 2019.
[16] M. Paquet-Clouston, B. Haslhofer, and B. Dupont, "Ransomware payments in the bitcoin ecosystem," *Journal of Cybersecurity*, 2019.
[17] M. A. Harlev, H. Sun Yin, K. C. Langenheldt, R. Mukkamala, and R. Vatrapu, "Breaking bad: De-anonymising entity types on the bitcoin blockchain using supervised machine learning," in *HICSS*, 2018.
[18] M. Weber, G. Domeniconi, J. Chen, D. K. I. Weidele, C. Bellei, T. Robinson, and C. E. Leiserson, "Anti-money laundering in bitcoin: Experimenting with graph convolutional networks for financial forensics," *arXiv preprint arXiv:1908.02591*, 2019.
[19] S. Yuan, P. Zheng, X. Wu, and Y. Xiang, "Wikipedia vandal early detection: from user behavior to user embedding," in *ECML PKDD*, 2017.
[20] C. Song, C. Yang, H. Chen, C. Tu, Z. Liu, and M. Sun, "Ced: Credible early detection of social media rumors," *IEEE TKDE*, 2019.