ChainNet: Learning on Blockchain Graphs with Topological Features

Nazmiye Ceren Abay
University of Texas at Dallas
Richardson, Texas
nazmiye.abay@utdallas.edu

Cuneyt G. Akcora
University of Texas at Dallas
Richardson, Texas
cuneyt.akcora@utdallas.edu

Yulia R. Gel
University of Texas at Dallas
Richardson, Texas
ygl@utdallas.edu

Umar D. Islambekov
University of Texas at Dallas
Richardson, Texas
umar@utdallas.edu

Murat Kantarcioglu
University of Texas at Dallas
Richardson, Texas
muratk@utdallas.edu

Bhavani Thuraisingham
University of Texas at Dallas
Richardson, Texas
bxt043000@utdallas.edu

ABSTRACT

With emergence and rapid adoption of blockchain technologies and the associated cryptocurrencies, such as Bitcoin, understanding the network dynamics behind Blockchain graphs become an important research direction. Unlike other financial networks such as stock and currency trading, blockchain based cryptocurrencies have the entire transaction graph accessible to the public (i.e., all transactions can be downloaded and analyzed). A natural question to ask is whether the network dynamics impact the price of the underlying cryptocurrency. In this work, we show that on one hand, standard graph features such as degree distribution may not be enough to capture the network dynamics that impact the underlying cryptocurrency price. On the other hand, we show that persistent homology or analysis of properties of progressively finer simplicial complexes can explain the high level interactions among nodes in Blockchain graphs and can be used to build much more accurate price prediction models. Using persistent homology based ideas, we offer an elegant, easily extendable and computationally light approach for graph representation learning on Blockchain networks to predict cryptocurrency prices. Using extensive analysis, we show that our proposed approach can explain the price dynamics significantly better than the baseline graph based features.

CCS CONCEPTS

• **Mathematics of computing** → Network flows; • **Theory of computation** → Computational pricing and auctions;

KEYWORDS

Blockchain, Bitcoin, Betti numbers, Persistent Homology, Deep Learning

ACM Reference Format:
Nazmiye Ceren Abay, Cuneyt G. Akcora, Yulia R. Gel, Umar D. Islambekov, Murat Kantarcioglu, and Bhavani Thuraisingham. 1997. ChainNet: Learning on Blockchain Graphs with Topological Features. In *Proceedings of ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'18).* ACM, New York, NY, USA, Article 4, 9 pages. https://doi.org/10.475/123_4

1 INTRODUCTION

Recent rise and then the fall of Bitcoin price has created lots of discussion with respect to future of Bitcoin and cryptocurrencies and its potential impact in financial markets [20]. One interesting aspect of popular cryptocurrencies such as Bitcoin is that each transaction is recorded on a distributed public ledger called blockchain. The transactions recorded on the blockchain can be accessed and analyzed by everybody. Furthermore, all of the transactions could be represented by a graph which we refer to as the "blockchain graph". Existence of the blockchain graph raises important questions such as "How does the blockchain graph structure impact the underlying cryptocurrency price?" In this paper, we focus on answering this question by proposing different approaches to represent blockchain graph patterns and using these patterns to build deep learning models for price prediction.

First approach that comes to mind to leverage blockchain graph structure is to extract traditional graph features such as degree distribution, and use these graph features in machine learning models such as deep learning to see their effectiveness in predicting price information. As already observed by previous work (e.g., [31]), and confirmed by our experimental results as well, these standard graph based features fail to capture important properties such as transaction volumes, transaction amounts, and their relationships with the underlying graph structure. Since these basic approaches do not provide conclusive insights into the blockchain graph dynamics and its impact on cryptocurrency price, we propose novel approaches inspired by topological data analysis and persistent homology based techniques that can capture these higher order interactions.

Persistent homology, or analysis of properties of progressively finer simplicial complexes, unveils some critical characteristics behind functionality of a blockchain graph and interactions of its components at multi-scale levels, which are otherwise largely unaccessible with conventional analytical methods. Such an approach provides a number of important benefits. First, we systematically account for mesoscopic changes in the blockchain graph geometry, both in terms of transaction patterns and associated transaction volumes. Second, analysis of the combinatorial structure of the abstract simplicial complexes associated with a blockchain graph allows

KDD'18, August 2018, London, United Kingdom

ACM ISBN 123-4567-24-567/08/06…$15.00
https://doi.org/10.475/123_4

bypassing a stage of feature engineering; we no longer need to subjectively select topological features, such as degree distribution, but instead use an exhaustive knowledge on topological invariants of the blockchain graph and evaluate its predictive role in cryptocurrency price dynamics. Third, while persistent homologies and topological data analysis, in general, are found to be indispensable tools for understanding the role of hidden geometry in organization and functionality of many complex systems, from cancer research to material sciences, their utility in complex blockchain graph yet remains largely unexplored. Nevertheless, the limited studies on application of topological data analysis to other type of networks show that persistent homology features outperform conventional graph features such as betweenness centrality, clustering coefficient and nodal degree in network classification and segmentation [11]. In this work, we bring the power of topological data analysis, particularly, persistent homologies and associated network filtrations, to analysis of blockchain graphs and dynamics of cryptocurrency price volatility.

Our contributions in this work can be summarized as follows:

- To our knowledge, we are the first one to use persistent homology based features combined with deep learning techniques to predict cryptocurrency prices.
- We develop novel techniques to integrate persistent homology ideas with deep learning models by extracting Betti numbers and network filtration based features from the blockchain graph.
- Using extensive empirical analysis, we show that our proposed persistent homology based deep learning models can significantly outperform (i.e., in many setting more than 50% improvement in root mean squared error (RMSE)) models that use only past price and/or standard graph features such as degree distribution information.

The remainder of the paper is organized as follows: In Section 2, we discuss the related work and emphasize the differences of our proposed approach. In Section 3.2, we discuss the background information related to blockchain graph representations and persistent homology. In Section 4, we discuss how we leverage the persistent homology based techniques to build deep learning models for predicting crpytocurrency price information. In Section 5, we present the experimental results based on the Bitcoin blockchain graph and Bitcoin price. Finally, in Section 6, we conclude by discussing the implications of our results with respect to cryptocurrency price dynamics and underlying blockchain graph structure.

2 RELATED WORK

The success of Bitcoin [24] has encouraged hundreds of similar digital coins [34]. Furthermore, the underlying Blockchain technology has been adopted in many use cases and applications. With this rapidly increasing activity, there have numerous studies analyzing the technology from different perspectives.

The earliest studies tracked the transaction network to locate coins used in illegal activities, such as money laundering and blackmailing [3, 26]. These results are known as the taint analysis [9]. For example, Moser et al. [23] studied activities against money laundering on Bitcoin by looking at how successive transactions are used to transfer money.

The Bitcoin network itself has also been studied from multiple aspects. Dyhrberg [10] studied Bitcoin's similarities to gold and the dollar, finding hedging capabilities and advantages as a medium of exchange. From a graph perspective, Baumann et al. [4] analyzed centralities, and [18] found that since 2010 the Bitcoin network can be considered a scale-free network. Furthermore, [16] tracked the evolution of the Bitcoin transaction network, and modeled degree distributions with power laws. Although these studies analyzed the Bitcoin graphs, the primary focus was on global graph characteristics.

Furthermore, Kristoufek analyzed potential drivers of Bitcoin prices, such as the impact of speculative and technical sources [17]. A number of recent studies show the utility of global graph features to predict the price [13, 15, 19]. For instance, [30] studied the impact of average balance, clustering coefficient, and number of new edges on the Bitcoin price. These findings suggest that certain network features are correlated with price; for example, the number of transactions put into a block indicates a price increase. Two network flow measures were recently proposed by [36] to quantify the dynamics of the Bitcoin transaction network and to assess the relationship between flow complexity and Bitcoin market variables. Furthermore, [19] identified 16 features (e.g., number of Tx) for 30, 60 or 120 minute intervals and used random forest models to predict the price. The core idea behind all these approaches is to extract certain global network features and to employ them for predictions. Most recently, [1] introduced the notion of *chainlet* motifs to understand the impact of local topological structures on Bitcoin price dynamics, and showed that employing aggregated chainlet information leads to more competitive price prediction mechanisms. In contrast to global network features, chainlets provide a finer grained insight at the network transactions. In practice, chainlets can be used to refine the above-mentioned models, so that features are computed on selected subgraphs only.

3 PRELIMINARIES

We begin by providing a brief introduction to blockchain graphs [2]. Next, we present Topological Data Analysis features with a focus on persistent homology.

3.1 Blockchain Graph Representations

In a typical blockchain graph such as the one used by Bitcoin, an owner of multiple addresses (i.e., each address represents an account, each person may have many addresses/accounts) can combine them in a transaction and send cryptocurrencies such as Bitcoin to multiple output addresses. Therefore, the Bitcoin blockchain consists of two types of nodes: transactions and addresses that are input/output of transactions. Earlier works on Blockchain analysis constructed graphs with a single type of node: *transactions* or *addresses* constituted nodes and currency transfers created edges between nodes (see [2] for a Blockchain graph review). By choosing a single type of node, these works ignore either address or transaction information in the graph. In our approach we construct a heterogeneous Blockchain graph with both address and transaction nodes. Figure 1 shows a blockchain graph with transactions as rectangle and addresses as circle shaped nodes, respectively. Each directed edge connects an address to a transaction, and the edge direction denotes a transfer of currency. On some blockchains, such as the Ethereum, edges may transfer messages (which in turn may contain information on artifacts such as tokens) only. Blockchain edges are naturally ordered in time with respect to the block they appear in.

Once the graph is constructed, shapes of transactions, and how they connect addresses conveys information on how the graph further extends in time. For all purposes, a Blockchain graph can be thought as a forever forward branching forest where transaction nodes appear only once, and address nodes may appear multiple times (but in practice address reuse is discouraged on Bitcoin and other blockchains).

With input and output addresses, each transaction represents an immutable decision that is encoded as a subgraph on the blockchain graph. In an earlier work we developed the idea of graph **chainlets** to encode and aggregate this information [1]. On the directed, heterogeneous blockchain graph $\mathcal{G} = (V, E, B)$, V is a set of vertices, and $E \subseteq V \times V$ is a set of directed edges. The set $B = \{\textbf{Address}, \textbf{Transaction}\}$ represents node types. On the blockchain graph $\mathcal{G}$, we define k-chainlets as follows:

Definition 3.1 (The k-Chainlet). A blockchain subgraph $\mathcal{G}' = (V', E', B)$ is a *subgraph* of $\mathcal{G}$ (i.e., $\mathcal{G}' \subseteq \mathcal{G}$), if $V' \subseteq V$ and $E' \subseteq E$. Let $\mathcal{G}_k = (V_k, G_k, B)$ be a subgraph of $\mathcal{G}$ with k nodes of type $\{\textbf{Transaction}\}$. We call the $\mathcal{G}_k$ a graph *k-chainlet.*

For simplicity, we refer to 1-chainlets as *chainlets* in the rest of this work. As the k value increases, $k-$*chainlets* encode higher order structures on the graph. With an increasing k value, the number of distinct shaped chainlets also increases. As each transaction can have thousands of inputs and outputs, even for the most basic case of $k = 1$, $k - $ *chainlets* can have millions of distinct shapes.

For $k = 1$ chainlets we denote distinct shapes with two dimensions (in general $2k$ dimensions are required): for $|i|$ *input* addresses and $|o|$ *output* addresses, the chainlet is denoted as $\mathbb{C}_{i\rightarrow o}$. The example Figure 1 shows 3 distinct chainlets: 2 of shape $\mathbb{C}_{2\rightarrow 2}$ (around t_1 and t_4), 1 of shape $\mathbb{C}_{3\rightarrow 1}$ (around t_2) and 1 of shape $\mathbb{C}_{1\rightarrow 3}$ (around t_3).

In motif analysis of networks [21], each motif (such as a triangle among three nodes) is counted on the network, and network dynamics are linked to motif densities. Chainlet analysis provides a similar role on Blockchain graphs; by counting the occurrence of certain shapes, a graph can be summarized with chainlet densities.

For a graph chainlet if there exists an a $G_k \in G$, we say that there exists an **occurrence**, or *embedding* of $\mathcal{G}_k$ in $\mathcal{G}$.

Another aspect of a chainlet is the amount of currency (or artifacts) that is transferred from its inputs to outputs; we call this the **amount** information of the chainlet, and denote the transferred amount as $\mathbb{V}_{i\rightarrow o}$.

With occurrence and amount informations of chainlets, the blockchain graph is represented with $[i_{max} \times o_{max}]$ dimensional occurrence and amount matrices, where the cell of ith row and oth column represents information on the chainlet $\mathbb{C}_{i\rightarrow o}$.

Example 3.2. Consider the example Figure 1, where both $i_{max} = 3$ and $o_{max} = 3$. In total, there are four chainlets but only three distinct shapes. The occurrence and amount matrices of the figure is given as:

$$\mathcal{O} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 2 & 0 \\ 1 & 0 & 0 \end{bmatrix} \text{ and } \mathcal{A} = \begin{bmatrix} 0 & 0 & 0.8 \\ 0 & 6.1 & 0 \\ 4 & 0 & 0 \end{bmatrix}, \text{ respectively.}$$

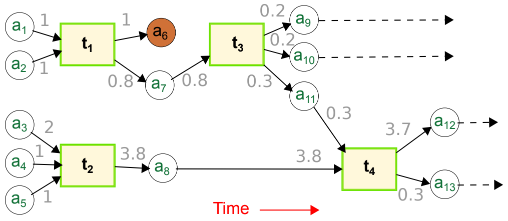


Figure 1: A bitcoin graph with 4 transactions and 13 addresses. Amounts on edges show currency transfers. The difference between input and outputs amounts, if exists, shows the transaction fee collected by miners.

In Section 4, we discuss how we use this chainlet information to extract persistent homology based deep learning model building.

3.2 Persistent Homology

Topological data analysis (TDA) and, persistent homologies, in particular, is an emerging methodology at the intersection of algebraic topology, statistics and machine learning that allow to systematically infer qualitative and quantitative mesoscopic geometric structures directly from the data and to enhance our understanding on the hidden role of geometry in functionality of a complex system [6], [35], [12], [8]. Yet, the utility of TDA in complex networks remains largely unexplored [27, 28]. However, the limited studies on application of TDA to random graphs indicate that analysis of topological invariants, (e.g., Betti numbers), outperform methods based on conventional graph features [11]. Our goal is to bring the persistent homologies tools to analysis of blockchain networks, and in this section we provide a general overview of the associated mathematical apparatus.

Let $\mathbb{X} = \{X_1, \ldots, X_n\}$ be a set of data points in a metric space (e.g., the Euclidean space or a manifold). Select a threshold ϵ_k and form a graph G_k with the associated adjacency matrix $A = \mathbb{1}_{d_{ij} \leq \epsilon_k}$, where d_{ij} is the distance between points X_i and X_j. Changing the threshold values $\epsilon_1 < \epsilon_2 < \ldots < \epsilon_N$ results in a hierarchical nested sequence of graphs $G_1 \subseteq G_2 \subseteq \ldots \subseteq G_N$ that is called as a *graph filtration*. That is, we glean the intrinsic geometry of $\{X_i\}_{i=1}^n$ from a multi-lens perspective, associated with a graph filtration.

Since it is generally hard to extract meaningful topological and geometric information from a discrete set of points, we associate an abstract simplicial complex with each G_k, $k = 1, \ldots, N$, which, in turn, allows to approximate the geometry underlying $\{X_i\}_{i=1}^n$ with a combinatorial structure. Furthermore, by quantifying all topological invariants associated with a simplicial complex, we bypass subjective selection of geometric features, or feature engineering. For instance, the Vietoris-Rips (VR) combinatorial complex is one of the the most popular choices in TDA due to its simplicity and computational advantages [7], [37].

Definition 3.3 (Vietoris-Rips complex). A *Vietoris-Rips complex* at threshold ϵ, denoted by VR_ϵ, is the abstract simplicial complex consisting of all k-element subsets of $\mathbb{X} = \{X_1, \ldots, X_n\}$, called $(k-1)$-simplices, $k = 1, \ldots, K$, whose points are pairwise within distance of ϵ. If $\mathbb{X} \subseteq \mathbb{R}^d$, a 0-simplex can be identified with a point, a 1-simplex with a segment, a 2-simplex is a triangle and a 3-simplex is a tetrahedron.

Now, armed with the associated *VR filtration*, $VR_1 \subseteq VR_2 \subseteq \ldots \subseteq VR_N$, we can track qualitative topological features such as connected components, 1-dimensional holes, 2-dimensional holes and their higher-order analogs, that appear and disappear with an increasing threshold ϵ.

In turn, analysis of evolution and lifespan of such topological features provides a multiscale quantitative insight into blockchain network geometry and its role in Bitcoin price formation. Systematic evaluation of patterns and dynamics of multiscale network geometry can be approached via an algebraic tool, based on the adaptation of a homology theory to applied data analysis and known as *persistent homology*. That is, the idea is to detect features which are long-lived, or persisting over varying thresholds ϵ_k and associated with refined scales of a graph filtration. One of the most widely used topological summaries of persistent features are Betti numbers.

Definition 3.4 (Betti numbers). The p-th Betti number $\beta_p, p \in Z^+$, of a simplicial complex (in our case the VR complex) is the rank of the associated p-th *homology* group defined as the quotient group of the *cycle* and *boundary* groups.

Getting into the technical details with regard to Betti numbers using the language of homology theory is beyond our scope here (see e.g. [8] for more rigorous treatment). Luckily, Betti numbers have a simpler interpretation - they represent the counts of connected components and p-dimensional holes. For instance, β_0 is the number of connected components; β_1 is the number of 1-dimensional holes, etc.

A convenient way to visualize persistent-homology-based summaries is by a *barcode* plot which is closely related to Betti numbers. A barcode is a set of stacked horizontal intervals (or bars), called *persistent intervals*, representing the birth and death of topological features of various dimensions (see Fig. 2).

4 METHODOLOGY

In this section we provide two predictive persistent homology inspired approaches for Blockchain graph analysis: Betti numbers (Betti) and graph filtration (FL). Our primary focus is on forecasting log returns of Bitcoin prices. Log returns are first differences of log prices recorded at the same unit time intervals. Log returns are widely used in mathematical finance, and there are multiple theoretic and algorithmic benefits of modeling and forecasting log returns vs. prices. In particular, such benefits include normalization, time-additivity, numerical stability, approximate raw-log equality, return-risk relationship, etc [5, 33].

In both newly proposed predictive approaches, the key insight is that by incorporating the edge weight information (i.e., the transferred amount), the Blockchain graph can be viewed through multiple perspectives. That is, in both methods we train a model on a set of data points $D = (X, Y)_{d,n}$ that holds the time series of d data pairs of length n, $\{(x_1, y_1), \ldots, (x_d, y_d)\}$, $x_d \in X$ with labels $y_d \in Y$ where Y is the daily Bitcoin price in dollars.

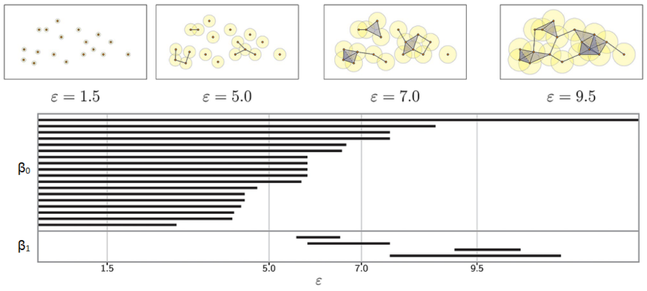


Figure 2: Barcode of a Vietoris-Rips complex built over 18 points in the plane. The top four figures are the snapshots of the evolving complex as threshold ϵ increases. The ϵ values corresponding to the two ends of horizontal bars mark the birth and death of topological features. To find Betti numbers we count the number of times respective horizontal bars intersect the vertical line through ϵ. For example, for $\epsilon = 7$, $\beta_0 = 4$ and $\beta_1 = 1$ [32].

The first approach uses Betti numbers discussed in Section 3.2 that pushes progressively finer simplicial complexes into the graph and captures graph properties. The second approach is based on graph filtration; we filter chainlets in the occurrence matrix with increasing thresholds of Bitcoin transaction amounts, and create multiple realizations of the occurrence matrix. Afterwards, we use these realizations to train multiple models, and combine their predictions.

The Betti approach is based on rigorous mathematical foundations of algebraic topology and provides a mesoscopic view of the system, whereas the graph filtration is a heuristic approach that allows selecting amount thresholds and associated filtering of occurrence matrices.

Next, we will describe these two approaches.

4.1 Learning with Persistent Topological Features

Our data model is based on the $n \times n$-occurrence matrix. As such, we consider n^2 different types of chainlets describing all possible transaction schemes. In the current study, following the notation of Section 3.2, with the chosen $n = 20$, the set $\mathbb{X}$ consists of 400 chainlets. Using the chainlet transaction data, we first define a suitable 'distance' d between the chainlets. We describe the main steps as follows:

(1) All the transferred amounts are converted from Satoshis to bitcoins (dividing by 10^8), then added one and log-transformed (added one so that the values after taking logarithm are non-negative): $y = \log(1 + x/10^8)$, where x is an amount in Satoshis.
(2) For each chainlet of a given day, we compute the sample q-quantiles for the associated log-transformed amounts [14]: a k-th q-quantile, $k = 0, 1, \ldots, q$, is the amount $Q(k)$ such

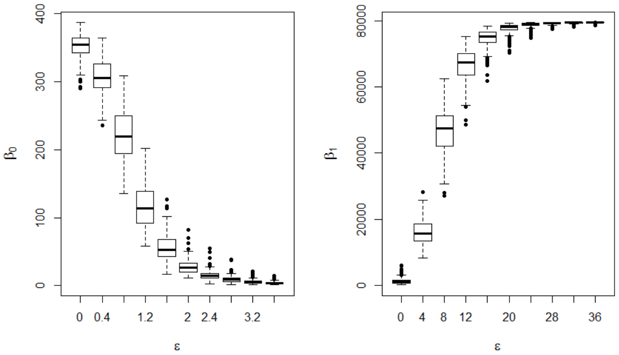


Figure 3: Boxplots of β_0 and β_1 numbers for various threshold ϵ values.

that

$$\sum_{i=1}^{n} \mathbb{1}_{y_i<Q(k)} \approx \frac{nk}{q} \text{ and } \sum_{i=1}^{n} \mathbb{1}_{y_i>Q(k)} \approx \frac{n(q-k)}{q},$$

where n is the total number of transactions. The (dis)similarity metric d_{ij} between chainlets indexed i and j is defined as the quantile-based distance

$$d_{ij} = \sqrt{\sum_{k=0}^{q}[Q_i(k) - Q_j(k)]^2}$$

In the present study we took $q = 20$.

(3) We construct a sequence of thresholds $\epsilon_1 < \epsilon_2 < \ldots < \epsilon_N$ covering a range of distances during the entire 365-day period. For each threshold ϵ_k, we build the corresponding Vietoris-Rips complex whose 0-simplices are single chainlets and 1-simplices are pairs of chainlets with distance less than or equal to ϵ_k. Thus, we obtain the filtration of VR complexes $VR_1 \subseteq VR_2 \subseteq \ldots \subseteq VR_N$ for each day.

(4) We compute β_0 and β_1 numbers on VR filtrations for each day. The dynamics of Betti numbers is depicted in Figures 3 and 4.

Figure 3 reveals visible variation in β_0 and β_1 numbers across 365 days for the initial values of ϵ and a general negative association between them – as ϵ increases, more simplices are added to the complex, thereby reducing the number of connected components and increasing the number of 1-dimensional holes. For the same reason, we see in Figure 4 that the spikes in average β_0 numbers match the plummets of the corresponding β_1 numbers and vice versa. Remarkably, we find that the spike in Bitcoin log returns in mid July 2017 have been preceded by an increase in β_0, and decreases in β_1 and average daily transactions. Moreover, extrema for Betti numbers β_0, β_1, and average daily transactions in July 2017 are well aligned. However, as shown by [1], daily transactions are outperformed by more fine grained chainlet motifs, in terms of their predictive utility of Bitcoin price. In this paper, we aim to get even a deeper insight into the role of local geometry in Blockchain on Bitcoin price formation.

4.2 Learning with Graph Filtration

In addition to the mathematically well defined Betti numbers over simplicial complexes, another way of describing the graph is by using a heuristic that filters the graph according to a set of user defined thresholds.

This approach leads to creating multiple realizations of a Bitcoin graph, and training a prediction model on each realization separately. Afterwards, multiple predictions for the same day is aggregated and a single prediction is created. In the rest of this work, we will refer to the graph filtration approach as the *FL*.

Example 4.1. Consider the scenario where we compute three filtrations of the example graph shown in Figure 1, with threshold values 0, 2 and 4. In the figure, chainlets transfer 2, 4, 0.8 and 4.1 bitcoins, respectively. As the filtration threshold is increased from 0 to 4, the three O matrices are created as follows:

$$O^0 = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 2 & 0 \\ 1 & 0 & 0 \end{bmatrix}, O^2 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 1 & 0 & 0 \end{bmatrix}, O^4 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

The FL captures persistent graph structures by retaining edges among nodes according to a set of threshold values. The FL starts with the occurrence O matrix of a day d. For a threshold value v chosen from the set filtration thresholds ϵ, we only record the occurrence of chainlet, if the amount transferred in the chainlet is more than or equal to v.

Algorithm 1 FL: Graph Filtration

```
Input: $Data : [X, Y]_{dxnxn}$ holds $nxn$ chainlets for $d$ days of a year
    where $Y$ is the daily Bitcoin price in dollars; s: training length;
    w: sliding window length; h: prediction horizon; $\epsilon$: set of $k$
    filtration thresholds $\epsilon_{1,..k}$.
Output: $Y_{predicted}$: Predicted log returns.
 1: $H \leftarrow \leftarrow \{\}$ //initialize model map
 2: for each threshold $\epsilon' \in \epsilon$ do
 3:     $O \leftarrow \{\}$
 4:     for each day $d$ do
 5:         Initialize $O^{\epsilon'}$ to hold $n \times n$ zero values.
 6:         for each $k = 1$ $\mathbb{C}_{i \to o} \in \mathcal{G}$ do
 7:             if Transferred amount $t_{\epsilon'} \leq \mathbb{V}_{i \to o}$ then
 8:                 Keep the occurrence in the $O^{\epsilon'}$
 9:         $O = O \cup O^{\epsilon'}$ //add the filtered chain
10:     $Y' \leftarrow SPred\left(O^{\epsilon'}, Y, w, h, s\right)$
11:     add $Y'$ to $H$
12: $model \leftarrow$ Combine filtration results in $H$
13: $Y_{predicted} \leftarrow$ Predict log return with $model$
14: return $Y_{predicted}$
```

The FL approach is given in Algorithm 1. Our framework aims to construct the graph filtration model with deep learning. Here, chainlets and their prices associated with each day d is given as an input. In line 2, for each threshold value $\epsilon' \in \epsilon_{1,..k}$ graph filtration is applied to $n \times n$ chainlets of a given day d. Each value of occurrence of the processed chainlet is checked and eliminated if it is below the threshold defined in the parameter of a model (Line 7-8 in Alg. 1).

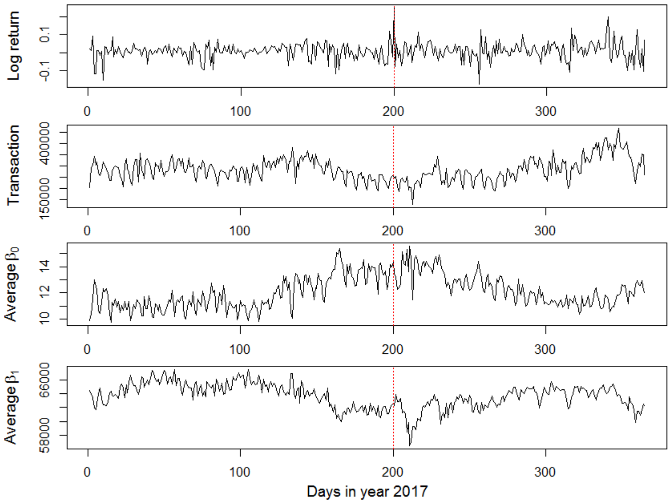


Figure 4: Time series of daily log returns, transactions, average β_0 and β_1 numbers in 2017.

Previously filtered elements are combined and given as an input of Alg. 2. The process is repeated for all threshold defined in a set ϵ and all filtration results are aggregated.

These filtration results are then used as an input to construct a new deep learning model that aggregates the results. Here, again the optimization is based on the predicted log return from the combined model and the log return of the day of the year (Line 12). At the end, forecasted log return values are outputted with the newly constructed model for the test days.

4.3 Prediction Model

In this section we describe the overall setting for predicting Bitcoin price in dollars.

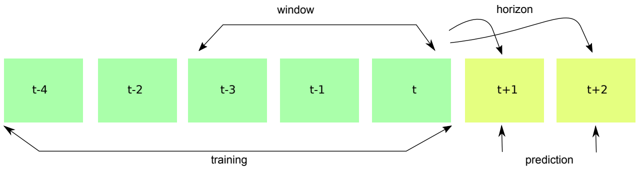


Figure 5: The sliding window model. The example model trains with data from the last $s = 5$ days, and uses the data from t, t_1 and t_2 (window=3) to make a prediction for either day $t + 1$ (horizon=1) or day $t + 2$ (horizon=2).

We use a sliding prediction scheme as shown in Figure 5. Our model is trained for each prediction separately. As we make use of lightweight occurrence and amount matrices, the computational cost of this choice over a batch prediction model is negligible.

After creating daily representations of the Bitcoin graph, we train the model on the preceding s days. In the most basic case of prediction horizon $h = 1$ and prediction window $w = 1$, the model learns to predict the price of day Y'_{d+1} by using the data X_d of day d. Similarly, for any window w, the model uses data from $\{X_{d-w} \dots x_d\}$ to predict the price Y'_{d+h} of day $d + h$.

Details of the sliding prediction approach is given in Algorithm 2. Here, input is assumed to be an ordered sequence of data associated with given day of the year and it label is the the daily Bitcoin price in dollars stated in Bitcoin system. In Line 1-9, sliding and processing the newly slided data is applied presented in Figure 5.

Algorithm 2 SPred: Sliding prediction

```
Input: Data: $D = (X, Y)_{d,n}$ holds the time series of d pairs, $\{(x_1, y_1), \dots, (x_d, y_d)\}$, $x_d \in X$ with labels $y_d \in Y$ where $Y$ is the daily Bitcoin price in dollars; $\eta$: learning rate; $b$: batch size; $T$: iteration number; $w$: sliding window length; $h$: prediction horizon; $s$: training length.
Output: $Y_{predicted}$: Predicted log return.
1:  for $i = 1$ to $d$ do
2:      $x_{training} \leftarrow \{\}$
3:      $y_{training} \leftarrow \{\}$
4:      for $j = 1$ to $j + s$ do
5:          $x' \leftarrow \frac{1}{w} \cdot \left(\sum_{k=i-w}^{i} x_k\right)$
6:          $x'' \leftarrow [y_{i-w}, \dots y_i]$
7:          $[x'; x''] \leftarrow$ concatenate $(x', x'')$
8:          append $[x'; x'']$ to $x_{training}$
9:          append log $\frac{y_{i+h}}{y_{i+h-1}}$ to $y_{training}$
10:     initialize $\theta_0$
11:     the objective function $\ell$
12:     the gradient of objective function $\nabla\ell$
13:     for $t = 0; t < T$ do
14:         $b_t \leftarrow$ random batch with size $b$ //from $x_{training}$
15:         $i_t \sim b$ where $x_{i_t} \in b_t$
16:         $\theta_{t+1} \leftarrow \theta_t - \eta \cdot \left(\frac{1}{b} \sum_{i_t} (\nabla\ell(\theta_t; x_{i_t}; y_{i_t}))\right)$
17:         t=t+1
18:     $y'_i \leftarrow$ Predict log return of $x_i$ with $\theta$
19:     append $y'_i$ to $Y_{predicted}$
```

Line 10-17 of Algorithm 2 infers the main optimization of the deep learning model while constructing it with the given input. To accelerate the optimization problem, stochastic gradient descent (SGD) is practiced in this algorithm [29]. Instead of computing the gradients based on all complete training set, SGD iterates only a subset of training instances. For a given training set of m samples, D=$\{x_i\}_{i=1}^{m}$ and $x_i \in \mathbb{R}^d$, objective problem can be given as:

$$\min_{\theta \in \mathbb{R}^d} \mathcal{L}(\theta) = \frac{1}{b} \sum_{x_i \in B} \ell(\theta; x_i; y_i), \qquad (1)$$

where θ is model parameter, and ℓ is the loss between an example, y_i and the predicted value of x_i made with model parameter.

At each step t, gradient of model is computed for a given batch b_t of size b, learning parameter η. Then, the model parameter is updated for next step as:

$$\theta_{t+1} = \theta_t - \eta \left(\frac{1}{b} \sum_{x \in b_t} \nabla_\theta \, \ell(\theta; x_i; y_i) \right). \qquad (2)$$

After constructing the model, we assess the model performance based on the root mean squared error (RMSE) metric defined below:

$$cost = \frac{1}{m} \sum_{i=1}^{m} \left\| p_i' - p_i \right\|. \qquad (3)$$

where m is the number of days, p_i' is the predicted log return from model, p_i is the original log return of i^{th} day of a required year.

5 EXPERIMENTS

In this section we demonstrate the effectiveness of our approach, which we will refer to as ChainNet, with experiments on the Bitcoin graph. To this end, we downloaded and parsed the entire Bitcoin transaction graph from 2009 to 2018 January. Using a time interval of 24 hours, we extracted daily transactions on the network and created the Bitcoin graph. Our code and datasets are available on Github at: https://github.com/cakcora/CoinWorks/.

We use the Betti numbers estimation routine from the Perseus [25] software which provides an efficient algorithm to compute Betti numbers and persistent intervals using discrete Morse theory. The algorithm pre-processes the complex in linear time and feeds it to the standard cubical algorithm based on Smith normal form [22].

On the Bitcoin graph % 90.50 of the chainlets have n of 5 (i.e., $\mathbb{C}_{i \to o}$ s.t., i < 5 and o < 5) in average for daily snapshots. This value reaches % 97.57 for n of 20. We chose to take $n = 20$, because it can distinguish a sufficiently large number (i.e., 400) of chainlets, and still offers a dense matrix.

In all experiments we chose to visualize results from the year 2017 because it is the latest complete year, and allows tracking seasonal effects in price. Figure 6 shows our outcome variable, i.e., the log return values in the network, for the year 2017. The log return values do not usually exceed ±0.1 but exhibit heavy tailed behavior.

The simplest baseline method for our work can be constructed by using past price values only in our sliding window prediction scheme. We will refer to this baseline method as m_0. The closest scholarly work to ChainNet is detailed in a report by Greaves et al. [13], where the authors extract both graph centric features (e.g., mean degree) and transaction features (e.g., mean amount) from the Bitcoin address graph, and use support vector machines to predict the Bitcoin price. As the authors also note at the end of their study, these features do not bring more information over a model that uses price data only. Indeed our experiments showed high error rates (RMSE=0.1) for predictions with the authors' experimental settings. However, we received better error rates when we plugged the same features to our sliding window prediction model. We will refer to this improved version of the Graeves' work by the term *feature* model.

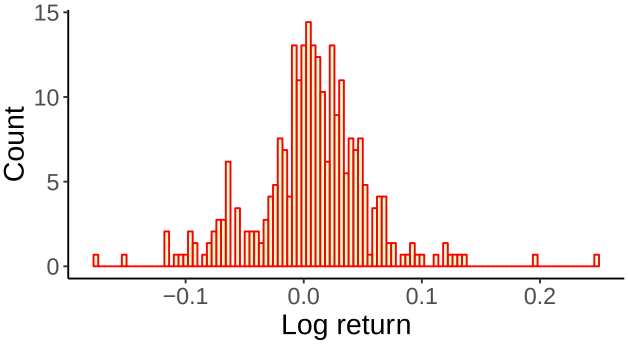


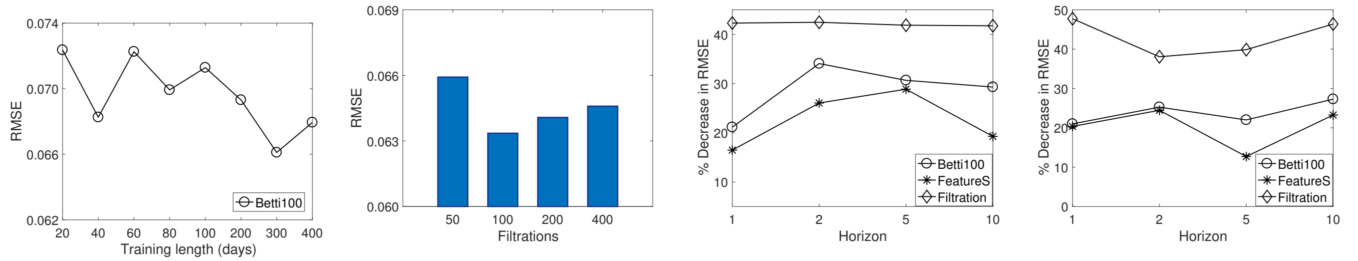
Figure 6: Histogram of log returns for the Bitcoin price in 2017.

We start discussing our results with a choice on the number of filtration levels in Betti results. In Section 4.1 we show how Betti numbers are computed on a graph of chainlets where the distribution of transferred values are used to create edges among the $n^2 = 400$ chainlet node types. The Betti filtration uses a sequence of thresholds $\epsilon_1 < \epsilon_2 < \ldots < \epsilon_N$ on these edges. In Figure 7b we change the filtration and report the corresponding root mean square (RMSE) values in predictions by using these filtrations. Here RMSE $= \sqrt{(1/n) \sum_{t=1}^{n} (y_t - \hat{y_t})^2}$, where y_t is the test set of bitcoin log return and $\hat{y_t}$ is the corresponding predicted value. As 100 filtrations give the best results, we use this value in rest of the experiments.

In the sliding prediction model that we use both in Betti Number (Betti100) and graph filtration (FL), we use three parameters to change our model.

The first parameter s is the training length, which indicates on how many days we train our model. Choosing a large value may prevent ChainNet from learning recent graph dynamics, whereas a small s value can lead to underfitting. In Figure 7a we report the RMSE values for various training lengths. Although the model reaches a local minimum around 40 days, we have the best result for 300 days.

The second parameter, window w, is the number of days whose data we use to predict a value after the model is learned. The third parameter is the number of days ahead in predictions. We refer to this parameter as the horizon h. As the interaction of horizon and window parameters may have non linear effects on the prediction,



(a) Impact of training length.

(b) Impact of filtration of Betti numbers.

(c) Impact of horizon days with window=3.

(d) Impact of horizon days with window=5

Figure 7: Changing ChainNet performance with changing window, horizon and training length parameters.

we show detailed results by varying both of them in Figures 7c and 7d.

In our studies, we report the percentage decrease in $RMSE$ for a specific approach m w.r.t. baseline approach m_0 as $\Delta_m = 100 \times (1 - \frac{RMSE_m}{RMSE_0})$, where $RMSE_0$ is computed by predictions with the baseline method m_0 (i.e., the price only model).

Predictive models for Bitcoin log returns, based on the new topological predictors such as Betti numbers and chainlet filtrations, substantially outperform baseline predictive models, based solely on price and conventional features, for all considered forecasting horizons from 1 to 10 days ahead (see Fig. 7).

As Fig. 7 indicates, the predictive models based on the new topological predictors provide 49% up improvement, whereas the models based on the conventional predictors such as Bitcoin price and global graph features reach 28% only. In particular, the most competitive performance is delivered by the predictive model based on the graph filtration, with 49% improvement vs. the conventional predictive model with model parameters window $w = 5$ and horizon $h = 1$ as shown in Figure 7. The next most competitive approach is based on the Betti numbers as topological predictor.

Figure 8 puts predictions of all models together for a visual comparison for window values 3 and 5 and horizon values 1, 2, 5 and 10. As seen in Figures 8c and 8d, the performance difference between our methods and baseline methods increase with increasing horizon, showing that our approach presents more information for longer term predictions. Overall, our methods outperform baseline methods consistently.

In Figure 9 we plot the Bitcoin price change in time along with predictions with the Betti method as well as the price only method. A visual inspection shows that a price only approach in price prediction suffers from sudden changes in prediction, whereas Betti smooths these effects and provides a better prediction. As the last days of 2017 showed extreme changes in price (e.g., plummeting from an all time high of 19K $), we show the price in the log scaled version in Figure 9b. This shows that baseline methods predict a momentum in log return movements; once price increases for a couple of days it is predicted to increase in the near future. Betti, on the other hand, can use the chainlet information and avoid making this mistake; although both are counted as transactions with traditional metrics, some chainlet shapes (e.g., $\mathbb{C}_{1\to20}$) show spending behavior (or a dispersion of bitcoins) whereas others (e.g., $\mathbb{C}_{20\to1}$) show a merging of funds into fewer addresses (saving behavior). Although our chainlet based methods can discern these two types, standard transaction counting cannot, and view every increased activity as a potentially price increasing move.

6 CONCLUSION

ChainNet is a deep learning platform that utilizes topological characteristics of a blockchain graph in explaining graph dynamics. ChainNet builds progressively expanding topological constructs, called simplicial complexes, over a graph and computes quantitative summaries in the form of Betti numbers which are then used in model building for the Bitcoin price prediction. Furthermore, ChainNet also offers a more heuristic based approach that allows user tailoring of system parameters for a finer grained look. The price prediction results on the full Bitcoin graph shows that these topological approaches capture the system dynamics well, and reduce the error in prediction by almost 50% over baseline approaches.

REFERENCES

[1] C. G. Akcora, A. K. Dey, Y. R. Gel, and M. Kantarcioglu. 2018. Forecasting Bitcoin Price with Graph Chainlets. *PaKDD* (2018).

[2] C. G. Akcora, Y. R. Gel, and M. Kantarcioglu. 2017. Blockchain: A Graph Primer. *arXiv preprint arXiv:1708.08749* (2017).

[3] Elli Androulaki, Ghassan O Karame, Marc Roeschlin, Tobias Scherer, and Srdjan Capkun. 2013. Evaluating user privacy in bitcoin. In *IFCA*. Springer, 34–51.

[4] A. Baumann, B. Fabian, and M. Lischke. 2014. Exploring the Bitcoin Network.. In *WEBIST (1)*. 369–374.

[5] T. Bollerslev, D. Osterrieder, N. Sizova, and G. Tauchen. 2013. Risk and Return: Long-Run Relationships, Fractional Cointegration, and Return Predictability. *Journal of Financial Economics* 108 (2013), 409–424.

[6] G. Carlsson. 2009. Topology and Data. *Bull. Amer. Math. Soc. (N.S.)* 46, 2 (2009), 255–308.

[7] G. Carlsson. 2009. Topology and Data. *Bull. Amer. Math. Soc.* 46, 2 (2009).

[8] F. Chazal and B. Michel. 2017. An introduction to Topological Data Analysis: fundamental and practical aspects for data scientists. *arXiv preprint arXiv:1710.04019* (2017).

[9] G. Di Battista, M. Di Donato, V.and Patrignani, M. Pizzonia, V. Roselli, and R. Tamassia. 2015. Bitconeview: visualization of flows in the bitcoin transaction graph. In *IEEE VizSec*. 1–8.

[10] A. H. Dyhrberg. 2016. Bitcoin, gold and the dollar–A GARCH volatility analysis. *Finance Research Letters* 16 (2016), 85–92.

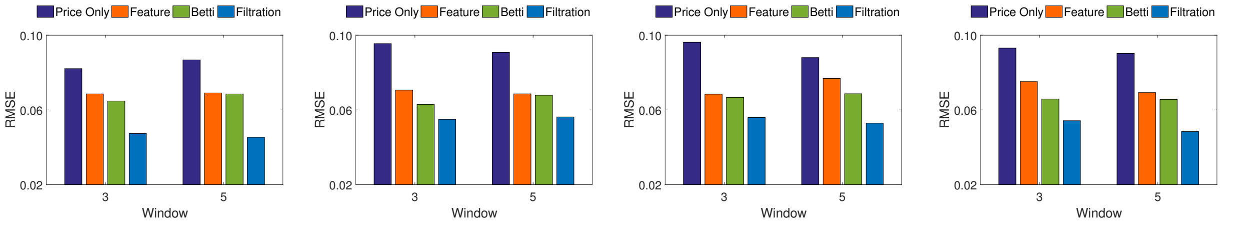
[11] A. Garg, D. Lu, K. Popuri, and M. F. Beg. 2016. Cortical Geometry Network and Topology Markers for ParkinsonâĂŹs Disease. *arXiv preprint arXiv:1611.04393* (2016).

[12] R. Ghrist. [n. d.]. Barcodes: The persistent topology of data. *Bull. Amer. Math. Soc.* 45 ([n. d.]), 61–75.

[13] A. Greaves and B. Au. 2015. Using the Bitcoin Transaction Graph to Predict the Price of Bitcoin. *No Data* (2015).

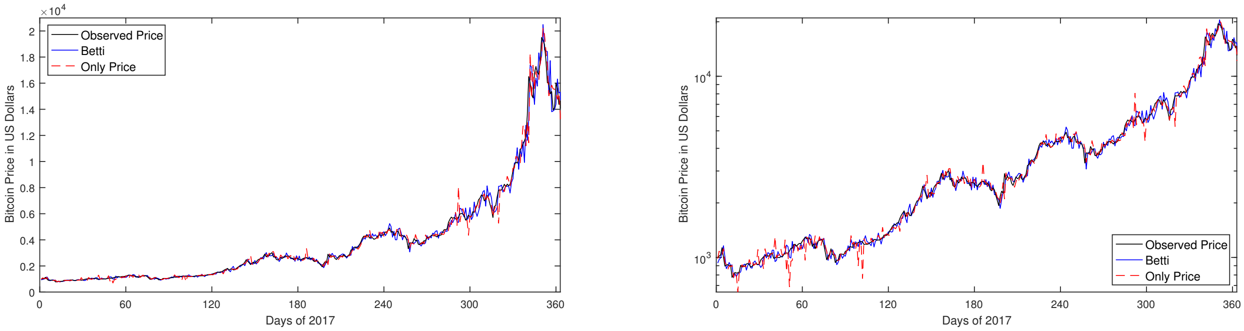
[14] R. Hyndman and Y. Fan. [n. d.]. Sample Quantiles in Statistical Packages. *The American Statistician* 50, 4 ([n. d.]), 361–365.

[15] D. Kondor, I. Csabai, J. Szüle, and G. Pósfai, M.and Vattay. 2014. Inferring the interplay between network structure and market effects in Bitcoin. *New J. of*



(a) Prediction horizon is 1 days (b) Prediction horizon is 2 days (c) Prediction horizon is 5 days (d) Prediction horizon is 10 days

Figure 8: [Color online]. Root Mean Squared Error (RMSE) for the two new proposed predictive approaches, i.e., based on Betti numbers and graph filtrations, and conventional predictive approaches, i.e., based solely on Bitcoin price and global graph features. Sliding window parameters are $w = 3$ and 5. Prediction horizons are $h = 1, 2, 5,$ and 10.



(a) Price prediction in 2017. (b) Price prediction in 2017 with a log scaled view of the y-axis.

Figure 9: [Color online]. The price of Bitcoin in 2017. Note the spikes in predictions with the baseline method.

Phys. 16, 12 (2014), 125003.

[16] D.l Kondor, M. Pósfai, I. Csabai, and G. Vattay. 2014. Do the rich get richer? An empirical analysis of the Bitcoin transaction network. *PloS one* 9, 2 (2014), e86197.

[17] L. Kristoufek. 2015. What are the main drivers of the Bitcoin price? Evidence from wavelet coherence analysis. *PloS One* 10, 4 (2015), e0123923.

[18] Matthias Lischke and Benjamin Fabian. 2016. Analyzing the bitcoin network: The first four years. *Future Internet* 8, 1 (2016), 7.

[19] S. Madan, I.and Saluja and A. Zhao. 2015. Automated Bitcoin Trading via Machine Learning Algorithms. (2015).

[20] Juri Mattila et al. 2016. *The Blockchain Phenomenon–The Disruptive Potential of Distributed Consensus Architectures.* Technical Report. The Research Institute of the Finnish Economy.

[21] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon. 2002. Network motifs: Simple building blocks of complex networks. *Science* 298, 5594 (2002), 824–827.

[22] Konstantin Mischaikow and Vidit Nanda. [n. d.]. Morse Theory for Filtrations and Efficient Computation of Persistent Homology. *Discrete Comput Geom* 50 ([n. d.]), 330–353.

[23] R. Moser, M.and Bohme and D. Breuker. 2013. An inquiry into money laundering tools in the Bitcoin ecosystem. In *eCRS*. IEEE, 1–14.

[24] Satoshi Nakamoto. 2008. Bitcoin: A peer-to-peer electronic cash system. (2008).

[25] V. Nanda. [n. d.]. Perseus: the persistent homology software. *http://people.maths.ox.ac.uk/nanda/perseus/index.html* ([n. d.]).

[26] M. Ober, S. Katzenbeisser, and K. Hamacher. 2013. Structure and anonymity of the bitcoin transaction graph. *Future internet* 5, 2 (2013), 237–250.

[27] N. Otter, M. A. Porter, U. Tillmann, P. Grindrod, and H. A. Harrington. 2017. A roadmap for the computation of persistent homology. *EPJ Data Science* 6, 1 (2017).

[28] A. Patania, F. Vaccarino, and G. Petri. 2017. Topological analysis of data. *EPJ Data Science* 6, 7 (2017).

[29] S. Song, K. Chaudhuri, and A. D. Sarwate. 2013. Stochastic gradient descent with differentially private updates. In *GlobalSIP, 2013 IEEE*. IEEE, 245–248.

[30] M. Sorgente and C. Cibils. 2014. The Reaction of a Network: Exploring the Relationship between the Bitcoin Network Structure and the Bitcoin Price. *No Data* (2014).

[31] T. Swanson. 2014. Learning from Bitcoin's past to improve its future. (2014).

[32] C. Topaz, L. Ziegelmeier, and T. Halverson. 2015. Topological Data Analysis of Biological Aggregation Models. *PLoS ONE* 10, 5 (2015).

[33] R. S. Tsay. 2010. *Analysis of Financial Time Series* (3rd ed.). Wiley.

[34] F. Tschorsch and journal=IEEE COMMUN SURV/TUT volume=18 number=3 pages=2084–2123 year=2016 publisher=IEEE Scheuermann, B. [n. d.]. Bitcoin and beyond: A technical survey on decentralized digital currencies. ([n. d.]).

[35] L. Wasserman. 2018. Topological Data Analysis. *Annual Review of Statistics and Its Application* 5, 1 (2018).

[36] S. Y Yang and J. Kim. 2015. Bitcoin Market Return and Volatility Forecasting Using Transaction Network Flow Properties. In *IEEE SSCI*. 1778–1785.

[37] A. Zomorodian. 2010. Fast construction of the Vietoris-Rips complex. *Computers and Graphics* 34, 3 (2010), 263–271.