Unmasking Cryptocurrency Pump and Dump Schemes: An Autoencoder Approach for Detection

Denis Reibel
denis.reibel@uni-ulm.de
ORCID: 0009-0007-1590-1820
Ulm University
Baden-Württemberg, Germany

ABSTRACT

Fraud patterns, like pump and dump schemes, pose a significant challenge to market integrity and investor trust in the cryptocurrency market. In this paper, we propose a novel approach to detect pump and dumps in cryptocurrency markets using autoencoder models for anomaly detection. Further, we evaluate how depicting a more realistic market scenario in the data influences the capabilities of our proposed models and current stat-of-the-art models. Findings show that our lightweight autoencoder models can match or even outperform more complex deep-learning models but confronting the models with a more realistic market scenario results in drastic loss of performance. We suggest an improved dataset for further research on pump and dump to evaluate their model capabilities.

KEYWORDS

Cryptocurrencies, Pump and Dumps, Anomaly Detection, Autoencoder

1 INTRODUCTION

Global adoption of cryptocurrencies like Bitcoin, Ethereum, and others is on the rise. A research report by Deutsche Bank Research from 2020 says the adoption speed is similar to the Internet adoption. It predicts that "there could be two hundred million blockchain wallet users by 2030" [11]. A downside is that cryptocurrencies are an easy target for scammers or fraudsters and are often connected with criminal activity. Fraudsters abuse crypto-assets for their benefit in many ways, like money laundering, rug pulls [17], or market manipulation.

This paper focuses on the pump and dump scheme which is a form of market manipulation that inflates asset prices for a short period of time only to let them fall sharply afterwards [12]. The pattern originates in the traditional stock market (where it is illegal nowadays) and was imported to the crypto-world. The idea is that a lot of people arrange a timeslot (often using platforms like Discord, Telegram, or Reddit) and then buy the same (mostly small-capped) asset simultaneously. Due to this the price of the asset rises and other uninformed investors get attracted. When the price rises, the initiators sell their investment with a profit. This causes the price to fall down sharply and leaves others with losses. Kamps and Kleinberg [8] divide pump and dumps into three phases: the accumulation phase, the pump phase, and the dump phase.

Figure 1 shows a pump and dump on the cryptocurrency SingularDTV (SNGLS) conducted by the "BigPumpSignal" telegram group on April 10th, 2018. The event was tracked in a public pump and dump dataset from La Morgia et al. [10], and the historical

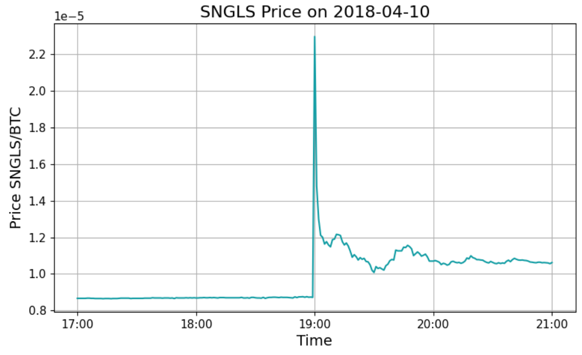


Figure 1: A pump and dump conducted by the "BigPumpSignal" telegram group targeting the SNGLS cryptocurrency on 10th, April 2028

price data was obtained using the Binance API[1]. Since the API only allowed the pairing with Bitcoin (BTC), prices are given in SNGLS/BTC.

As visible in the figure, the pump and dump was announced at 19:00. The price of the coin jumped immediately from 0.87e-05 SNGLS/BTC per coin to 2.30e-05 SNGLS/BTC per coin, reaching its highest point only seconds after the start of the event (pump phase). Directly afterward, the price rapidly fell to reach 1.15e-05 SNGLS/BTC per coin (dump phase). It is not uncommon to see the price fall even below the level it had before the pump. So most investors that jumped onto the pump only shortly after it began, did lose money on this investment. It is not unlikely that only the group admins and informed insiders made profits from this event. They know which is the next target coin and can therefore buy at low prices before the pump gets announced (accumulation phase).

Cryptocurrencies are an interesting target for market manipulation especially pump and dumps for multiple reasons. First of all, anyone can participate and the barrier to entry is relatively low. Participants can stay pseudonymous so they do not fear any legal consequences. While centralized exchanges often require some sort of identification, trading on uncentralized exchanges often has no restrictions. Further, the amount of purchase power needed to move prices by a relevant margin is relatively low on crypto-assets compared to the traditional stock market. Especially small-capped coins or small exchanges with low liquidity pools are an interesting target due to this reason.

[1]https://www.binance.com/en/binance-api

To tackle these problems, this paper contributes towards building a pump and dump detection model that can recognize when characteristic market patterns evolve. Therefore we propose a chunk-based approach using an autoencoder model that can classify time chunks into pump and dumps or natural movement. The question that will be answered is phrased as follows:

RQ1 *What is the effectiveness of an autoencoder model in the detection of pump and dump schemes in cryptocurrency markets, and how does its performance compare to established baselines in this field?*

Another important aspect that past works have not discussed so far, is how a model can differentiate between a planned pump and dump event and a natural price spike. Cryptocurrencies can be very volatile so large up or down movements of prices are not uncommon and do not necessarily come from pump and dumps. Thus the second goal of this paper is to answer the question:

RQ2: *What performance changes can be observed when our data also embodies events that are not an artificial pushing of the price (pump and dump) but caused by a natural movement?*

.

In the following section, the current state of research around pump and dumps on cryptocurrencies is introduced. Thereafter, a glimpse into the real world is given with an overview of the regulatory situation and how a detection model could be used outside of research. In section four the experimental setup, the data used, and the detection methods are explained. The results are described and discussed in section five. The last section summarizes the results and concludes this work.

2 BACKGROUND & RELATED WORK

Pump and dumps can be confused with rug pulls or the two terms are sometimes used as equivalents. Yet there is a clear distinction between them. With rug pulls, the developers of a cryptocurrency project create hype about a new asset. When their target is reached they lock all investors out of selling their coins or they clear the liquidity pool. The project was meant as a scam from the beginning and was left dead afterwards. This pattern mostly occurs on NFTs and the hype around the project is often created through social media platforms [17]. On the other hand, pump and dumps are more common on crypto-assets and require a large group of people who are willing to assist in conducting the event.

This paper focuses on pump and dump schemes executed on the Ethereum blockchain. NFTs are excluded because as explained they usually get rug pulled [17]. The reason behind the focus on Ethereum-based tokens is to keep this research in a clearly defined setting and keep the data-gathering process manageable.

Previous works have already explored different methods for building a detection system for pump and dumps on cryptocurrencies. Kamps et al. [8] built a detection system for pump and dumps that recognizes anomalies with a pure statistical method based on dynamic thresholds. They attempted to set thresholds for the price and volume change over time. This was outperformed by the work of La Morgia et al. [10] with their Random Forest detection model, which is based on features they derived from order book data. With the publication of this data, they released a well-structured open-source dataset of pump and dump events. Kamps et al. [8] and Victor et al. [18] also used pump and dump events from Telegram as the ground truth for their models. However, the dataset published by La Morgia et al. [13] is, as of now, the most comprehensive open-source dataset for pump and dump detection.

An approach using Deep Learning methods was done by Chadalapaka et al. [3]. They used the dataset and features provided by La Morgia et al. [10] to train their deep learning models. As architectures, they evaluated a convolutional long-short-term memory model (C-LSTM) and also an Anomaly Transformer model. The C-LSTM model they used consisted of 997,851 learnable parameters and was trained by them for 200 epochs. The Anomaly Transformer model was originally introduced by Xu et al. [21] for unsupervised anomaly detection settings. Chadalapaka et al. slightly changed the implementation to fit the supervised setting La Morgia's Dataset provides and, although not stated in their work, the model was applied to a point-global problem. Chadalapaka et al. reported their Anomaly Transformer has 155,665 learnable parameters and was trained for 50 epochs. Both models did outperform Kamps et al. as well as La Morgia et al. They criticized that the dataset was preprocessed and argued, that deep learning models have the capability of dealing with raw data in a much more effective way than basic statistic models could.

In addition to the task of detecting pump and dumps, there is other notable research around pump and dumps on cryptocurrencies. Xu and Livshits [20] tested basic classification and regression models for the task of predicting the next target coin a group will pick. Nghiem et al. [15] tried to solve the same task by using market and social media signals, and Hu et al. [7] recently published a sequence-based deep learning model for this task.

Pump and dump detection on cryptocurrencies poses specific challenges. First of all, the anomaly is not caused by any natural reason, but instead by a group of people that appoint a specific time and cryptocurrency for the event. Therefore, there are no warning signs or patterns in the data that could presage a coming pump and dump beforehand. It could be argued that the accumulation phase, as described above, is such a warning sign but it is so indistinctive from usual trading that it cannot be differentiated and just blurs into the regular noise [8]. Another characteristic is that the start of pump and dump events can be determined by the announcements in the respective channel. In contrast, the end often can not be clearly identified, not even by looking at the data to manually label each end, as minor subsequent pump and dumps are not uncommon. Further, an important goal of the task is to detect pump and dumps as soon as they arise. Therefore, it is not reasonable to look at the whole time series and let the model label anomalies in the aftermath. Instead, the approach needs to be able to continuously look at the data and decide whether or not a pump and dump is going on at the moment. Given these characteristics, the problem can be formulated as a classification problem, since the time buckets can be viewed independently and judged whether or not they are an anomaly. This way, any time bucket can be fed into the model and the output tells whether or not a pump and dump is taking place.

When trying to find anomalies, autoencoder architectures are widely used and have proven to be very successful for anomaly classification by previous works [1, 19, 22]. With this paper, we aim to expand the perspective on cryptocurrency pump and dump detection by approaching the problem with an autoencoder-based approach. Further, we discuss what is necessary for a detection system that could also be useful for real-world application and therefore start bridging the gap between research and real-world use.

3 A GLIMPSE INTO THE REAL WORLD

This section provides an impression of what the real-world situation around pump and dumps looks like. Other papers have already looked at pump and dump groups and conducted detailed case studies [8, 10, 12], so providing yet another case study would not contribute new knowledge. Instead with this paper, we want to open the discussion of what is necessary to make pump and dumps and detection systems ready for real-world usage.

3.1 Time to Spike

In forums where pump and dump schemes are discussed, a common question is how easy it is to make money. The usual response is that it is important to be really fast, as the peak (spike) of the event is typically reached shortly after the announcement. To determine the necessary speed to make a profit, the time between the announcement and the highest spike can be measured.

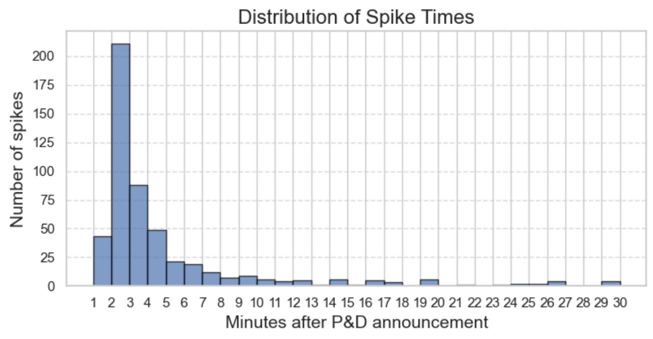


Figure 2: Distribution of time the pump and dump events need to reach their spike after the initial announcement

Figure 2 shows the distribution of minutes until a spike is reached. For this analysis, 513 pump and dump events from the dataset of La Morgia et al. [10] that were carried out on the Binance exchange were analyzed. The price data was gathered using the public API of Binance[1], which provided the closing prices in one-minute intervals.

It shows that most of the events reach their highest spike between two to three minutes after the announcement. The median spike time is at 3 minutes and the interquartile range is 2 minutes. These descriptive measures have been chosen because they are robust against outliers which can be seen in the figure and would distort measures like the mean. So in fact, to make a profit in the majority of pump and dump events, a participant would have to place a buy order for the asset, wait a short period, and sell the asset again all in under three minutes. Even then the majority of the profits are modest because the participant was "late to the party".

3.2 Regulatory Background

To understand why pump and dumps are so widespread on crypto-assets, the legal situation is summarized in the following. On the traditional stock market, different nations have laws that declare market manipulation (such as pump and dumps) illegal and authorities that control the exchanges, banking institutions, and other market participants. On cryptocurrencies, this is far more complicated and different problems arise. First of all, cryptocurrencies are still a relatively new field compared to the traditional banking system (although the original bitcoin paper [14] was published 15 years ago). There are simply not enough clear regulations globally and precedent cases are missing. Further, the distributed nature of cryptocurrencies makes it difficult to determine the territorial connections of the emitters and find out which laws apply to them. Additionally, the pseudonymous (respectively even anonymous) character of cryptocurrencies can make it difficult to identify responsible persons behind the fraud. The next problem poses the diversity of token types which impedes defining regulations for all tokens.

If we look at German law as an example case, there is a binding EU regulation called the Market Abuse Regulation (short: MAR) [4] which targets the stock market and regulates market manipulations. A newer EU regulation from early 2023 called the Markets in Crypto-Assets Regulation (short: MiCAR) [5] specifically targets the crypto market and regulates market manipulation there. However, it makes big differences between different token types (i.e. utility tokens, asset tokens, currency tokens, and NFTs). Grimm and Kreuter [6] wrote an essay about the regulation, commenting that the MiCAR closes a regulatory gap and enables EU authorities to stop and punish market manipulation on cryptocurrencies. However, in their opinion, the real-world effect of the regulation needs to be awaited and it is unclear to what extent the regulation can keep pace with the dynamic ecosystem of crypto markets [6]. In summary, while the global nature of both the crypto market and the stock market inherit regulatory challenges, the traditional stock market benefits from a long-established regulatory framework, clearly identified market participants, and a clearer structure of the assets traded.

3.3 Real World Uses Cases and Implementation Ideas

Given that regulation by law is complicated, a detection system could help to find and report pump and dump events. To answer this question sufficiently, the real-world use cases for a pump and dump detection system need to be identified. An idea for a use case could be finding all pump and dumps that have taken place on a crypto-asset by searching through the historical data. This generates knowledge about how many pump and dumps have taken place on a specific crypto-asset and which assets are especially prone to being a target of such frauds. However, probably the most interesting use case is protecting unknowing investors from losing their money in a pump and dump fraud. For this the education of the investors is key. Yet, even for well-informed investors, it can be difficult to detect a pump and dump scheme and even harder to differentiate it from a natural price volatility which is common on the crypto market. Therefore a system that could detect an ongoing pump and dump in real time and warn investors is needed.

There are different scenarios possible to implement such a system. One option would be to scan through all Telegram channels, Discord groups, or Reddit forums and calculate the warnings based on when they post a pump and dump starts. However, it is simply not possible to keep track of all pump and dump groups and the idea does not scale for long-term usage. Also, scanning through news or social media data is not feasible because it is too slow and unreliable [16].

Because of these reasons, the current state of research focuses on data like order books and price data. Based on this, the pump and dumps are detected using anomaly detection techniques. When detected, an exchange could display warnings for the targeted coin. To do so, an exchange platform would have to scan continuously through all crypto-assets they offer to trade. To give an example, Binance, as the biggest exchange platform by market share, currently offers 372 tradeable cryptocurrencies[2]. CoinGecko, an API provider for crypto-prices and structured data, has 10,116 crypto-assets listed[3]. Other platforms even provide more tradeable cryptos and since new assets are very easy to create, the number is growing rapidly. Scanning through data for all those assets would be a challenging task in terms of the computation power needed. To solve this problem, one could decide to restrict the search space to specific coins or blockchains.

The current detection models (such as La Morgia et al. or Chadalapaka et al.) rely on features that need the order book data, which mostly only enables the exchange platforms to efficiently implement such a detection model. To surpass this, models that can detect pump and dumps based on features that derive from open-source data, like on-chain information or price data, are needed.

In summary, the current detection models use data that is offered by the exchange platforms and therefore rely on the exchanges. To decouple the pump and dump detection from these, features independent of exchange data need to be evaluated. Further, the detection needs to be capable of differentiating between pump and dumps and naturally triggered movements to generate reliable warnings and inform investors.

4 EXPERIMENTAL SETUP

4.1 Datasets

4.1.1 La Morgia Dataset

The dataset used for evaluating our autoencoder models is the pump and dump dataset published by La Morgia et al. [10], which was also used in the work of Chadalapaka et al. [3]. Because both of these works used the same data, we can compare our performance on the dataset and get meaningful results. La Morgias dataset includes 1,111 pump and dump events collected by observing 20 different Telegram groups from July 2017 until January 2021. The pump and dump events are distributed among 372 cryptocurrencies on five different exchange platforms. The dataset also includes the processed labeled feature data points. These were created by downloading the order book data from the 317 unique events carried out on the Binance exchange and then calculating the features. For each pump and dump event on Binance, they split the order book into time slices and calculated the features for each slice. In the end, three versions of the dataset were released with either 5-, 15- or 25-second sized time slices. For each pump and dump event, they labeled the slice that matched the announcement time of the event as an anomaly. Since the duration of the event was unclear, all slices afterwards were dropped and all slices beforehand were labeled as normal slices.

In this paper, we are only using the 15-second-based dataset. This is because the comparison between the chunk sizes has already been done by other works [3, 10] which showed that the performance does not change much over the different chunk sizes.

The 15s-sized dataset includes 317 anomaly data points (from 317 pump and dump events) and 583,787 normal data points. This is a highly imbalanced dataset, with an anomaly to normal ratio of only 0.0386%. Such an imbalance poses a high risk of influencing the models and distorting the final metrics. In the worst case, the models could learn to ignore the anomaly class completely and would still score high on most metrics. Therefore, this imbalance has to be kept in mind when training and evaluating final metrics.

We also tried downloading the data for pump and dump events that were carried out on other exchanges than Binance to extend the dataset. However, as of now, only Binance has a public API that offers its historical order book data. So retrieving data from the other exchanges was not possible.

4.1.2 NatSD Dataset

To evaluate how well a model could differentiate between pump and dumps and naturally occurring spikes on cryptocurrencies, a dataset of such natural spikes is needed. However, there is currently no public dataset available. Thus, to create such a dataset the historical prices for Bitcoin (BTC), Ethereum (ETH), and Ripple (XRP) were downloaded for a three-year window (September 2020 until September 2023) using the Binance API. This was the best source for providing minutely historical price data for different coin pairs. However, there were some minor price gaps where Binance did not provide any data. Bitcoin, Ethereum, and Ripple were selected because of their leading positions in market capitalization. Naturally, there is still a risk that the downloaded data contains pump and dump events, but by choosing such large capped coins the risk is minimized.

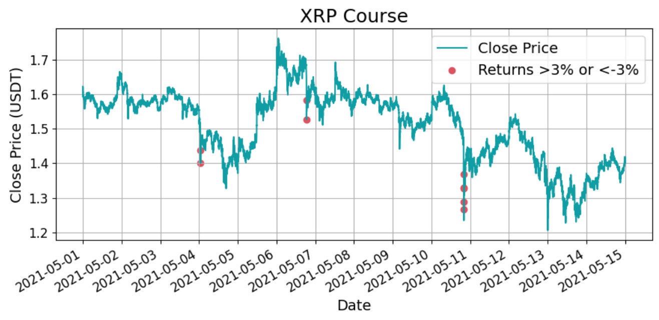


Figure 3: Visualization of the XRP course in 2021 with highlighted returns over 3% or under -3%

Using this data, the returns for each minute slice compared to the slice before were calculated. We used the closing price for each

[2]https://www.binance.com/en/markets/overview
[3]https://www.coingecko.com/en/api

minute to calculate the returns to keep the numbers comparable. To find natural spikes and drawdowns, each event with a return over three percent was labeled as a spike, and under minus three percent was labeled as a drawdown. Figure 3 shows an excerpt of the XRP course with highlighted spikes and drawdowns. Some points (for example 2021-05-13) look like they have sharp drawdowns but when analyzing the minute-based prices the return change rate does not exceed minus three percent.

The final dataset contained 324 natural spikes or drawdowns distributed between Bitcoin (26 events), Ethereum (52 events), and Ripple (246 events) as shown in Figure 4. The high representation of Ripple can be explained by the currency being more volatile in comparison to Bitcoin or Ethereum.

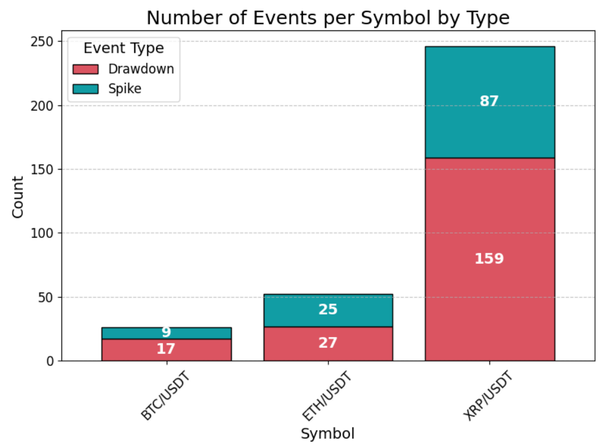


Figure 4: Distribution of spikes and drawdowns per coin pairing in the dataset

The order book data for the natural spike and drawdowns was downloaded and the same features as in the dataset of La Morgia et al. were calculated by using their provided code. We consider two minute ranges before and after each natural spike or drawdown event. As example, for the 15S-chunks NatSD data, this results in 3,040 data points. All new data points are labeled as normal since the goal is to detect pump and dumps and minimize false positives (as they would likely be produced by natural spikes or drawdowns). This way, it is ensured our dataset extension was put together in a manner so that it fits in well with the data from La Morgia et al. [10] and the new points are not distorted with a bias. Throughout this paper, our data extension is referred to as NatSD dataset.

4.2 Features

The dataset of La Morgia et al. and our NatSD data include eight features that have been calculated for each chunk. Each data point has the following features [10]:

- **StdRushOrders and AvgRushOrders:** Moving standard deviation and average of the number of rush orders in each chunk of the moving window.
- **StdTrades:** Moving standard deviation of the number of trades.
- **StdVolumes and AvgVolumes:** Moving standard deviation and average of volume of trades in each chunk of the moving window.
- **StdPrice and AvgPrice:** Moving standard deviation and average of closing price.
- **AvgPriceMax:** Moving average of maximal and minimum price in each chunk.

The features had been calculated based on the order book data provided by the Binance API. Since Binance only provides information about sells or buys, but not about the kind of order, La Morgia et al. had to infere this information. They named their infered order type "rush orders", which are orders that are filled in the second they are placed.

Some problems arise with that kind of feature creation for detecting pump and dumps. First of all, a dependence on the trading platforms or exchanges is created. Detecting pump and dumps is now dependent on their order book data and the quality of the data. As mentioned above, among the exchanges included in their dataset, Binance is the only one that offers access to its order book data. Further, the process of inferring the order type (like La Morgia et al. did) inherits some problems, as they described in their paper, and can result in faulty data.

4.3 Methods

Autoencoders are a neural network architecture that is commonly used for anomaly detection. Many research works [1, 19, 22] have proven their usefulness for detecting data points that differ from regular patterns.

4.3.1 Autoencoder

Autoencoders consist of two parts, the encoder and the decoder net [2]. Figure 5 shows the schema of a typical autoencoder architecture. The dimension of the input features $x_1, ..., x_i$ gets reduced by the encoder net down to the code size, often also referred to as the latent space. The latent space in autoencoders is usually unregularized. Variational autoencoders aim at a more regularized latent space by introducing this into the total loss function. In the next step, the decoder net tries to recreate the input from the given code, so that in the end the reconstructions $\hat{x}_1, ..., \hat{x}_i$ are obtained. The reconstructions are compared to the input and an error is calculated, based on which the weights of the network are adjusted to train the model.

To use an autoencoder model for anomaly detection, it is important to train it only on normal data points. This way, the network learns how to properly represent the normal data. The testing set should contain normal and anomaly data points. Using the trained model, the reconstruction errors for each point can be calculated. Now an error threshold needs to be defined which is used to separate normal data from anomalies. In theory, normal data can be represented well by the model and therefore has low errors compared to anomaly data. To choose a threshold, there are different strategies but a very common way is to use the mean μ with a weighted standard deviation std like this $\mu + k * std$. When aiming for the best performance, it is important to optimize the model towards the desired metric. For anomaly detection, it is difficult to do this. Instead, the model often gets optimized towards a minimum loss during the training phase. An important hyperparameter to

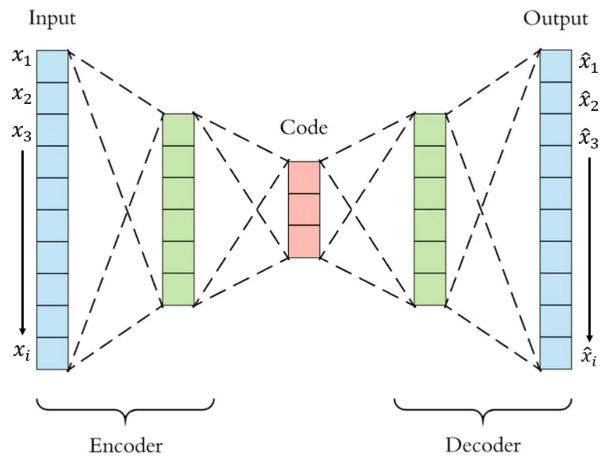


Figure 5: Schema of an autoencoder architecture

tune is the code size, as well as the depth of the model itself. Other parameters that can be looked at, are the learning rate, the number of epochs, the batch size per epoch, or the chosen optimizer.

A major benefit of using autoencoders is that for anomaly detection, the model only needs normal data points for training. Therefore, all the anomaly data points and a random sample of the unseen normal data points can be used for testing.

4.3.2 *Variational Autoencoder*

Variational autoencoders (VAEs) [9] differ from traditional autoencoders in how they handle the latent space. While autoencoders reduce input features to a fixed code or latent space, VAEs introduce a probabilistic approach to this reduction. In a VAE, the encoder network learns to map input data to a probability distribution in the latent space, typically a Gaussian distribution with a mean and a variance. This means that instead of obtaining a single fixed code, a probability distribution for each data point is created.

The decoder network in a VAE then samples from these distributions to generate data points in the latent space. This sampling introduces a stochastic element, making VAEs generative models that can produce diverse outputs from the same input. In the context of anomaly detection, VAEs are used because they can capture the inherent uncertainty in the data and can perform better when it comes to modeling complex data distributions [9].

To use a VAE for anomaly detection, it is trained similarly to traditional autoencoders. During testing, the VAE computes the reconstruction error but instead of a deterministic error, a probability distribution of errors is obtained. Anomalies are often detected by considering how likely a data point is given this error distribution, and points with low likelihoods are flagged as anomalies.

The training of VAEs also involves optimizing two objectives: the reconstruction loss and a regularization term that encourages the latent space to follow a specific distribution (usually Gaussian). The regularization term helps ensure that the latent space is continuous and well-structured. A commonly used method for this is the Kullback-Leibler divergence between the resulting distribution and the aspired distribution.

In summary, VAEs differ from traditional autoencoders by introducing probabilistic elements to the encoding and decoding processes, making them more suitable for modeling complex data distributions and capturing uncertainty.

4.3.3 *Feature Importance*

The reconstruction error for autoencoders is often calculated using the mean squared error which is defined by Equation 1. It is used to define the difference between the input x_i and the reconstruction of the input $\hat{x}_i$ for every feature N. With this error, the model can evaluate its performance and adjust the weights.

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(x_i - \hat{x}_i)^2 \tag{1}$$

To determine how influential each feature was for the final reconstruction error, we can use Equation 2. This term describes the mean reconstruction error $e_{m,i}$ for the feature i from sample m.

$$e_{m,i} = (x_{m,i} - \hat{x}_{m,i})^2 \tag{2}$$

As a last step, we have to calculate the mean error for feature i, by taking the mean of $e_{m,i}$ over all samples m from M, as shown in Equation 3.

$$\text{Mean Error}_i = \frac{1}{M}\sum_{m=1}^{M} e_{m,i} \tag{3}$$

This way the mean reconstruction error for feature i is calculated. Doing this for all features, the values can be compared against each other, so that the influence of a feature on the final classification result can be derived.

4.4 Procedure

4.4.1 *Anomaly Detection on AE's / VAE's*

The first experiment targets to answer the research question ***RQ1:*** *"What is the effectiveness of an autoencoder model in the detection of pump and dump schemes in cryptocurrency markets, and how does its performance compare to established baselines in this field?"*. To answer this, we developed an autoencoder as well as a variational autoencoder model. Those are trained and tested on the La Morgia pump and dump dataset using the labeled features they provided. This ensures, that the achieved results are reliable and comparable to other works that used this dataset. Since autoencoders only need normal data points when training for anomaly detection, the data was split into normal and anomaly data points initially. The train set was built using 95% of the normal data points. The leftover 5% of normal data points as well as 100% of the anomaly data points were combined to create the test set. The normal data split ratio may sound high at first but as described in section 4.1 the dataset is highly skewed between anomaly and normal data points. For the 15-second-chunks-based dataset, this data split creates a train set of 554,598 normal data points and a test set including 29,189 normal data points and 317 anomaly data points. With these, the autoencoder and the variational autoencoder models were trained and evaluated.

The final trained models were evaluated using the test set. First, the test set is run through the model, and the reconstruction errors are stored. The decision of whether or not a data point is normal or an anomaly is then made by an error threshold that separates the two classes. This threshold was defined as $\mu + std$ for the autoencoder and as 99.65%-quantile for the variational autoencoder. With the assigned classes, the evaluation metrics can be calculated. As metrics, we measured the Precision and Recall. Because the dataset is highly skewed, we omitted the Accuracy since even if all anomalies were misclassified the Accuracy would be 99.94%, so reporting it would be misleading. Instead, it is important to measure the F1-macro, since this is calculated with respect to the different classes. To estimate how good the final results are, they are compared against other models. For this, the chosen baselines are the Random Forest from La Morgia et al. [10] and the C-LSTM as well as the Anomaly Transformer evaluated by Chadalapaka et al. [3]. Their models were chosen for comparison because both used the same dataset and achieved strong results which gives a reliable testimony of how qualitative the results achieved by our models are.

The numbers for the C-LSTM, the Anomaly Transformer, and the Random Forest were calculated by using 5-fold cross-validation. Since autoencoders are trained with only normal data points a typical cross-validation is not applicable. Instead, we used the best hyperparameter combination found by our random search. This was run five times with different random states (influencing the split of the normal data and the weight initialization). We report the average of those five runs.

4.4.2 Performance Evaluation on NatSD Data

An aspect that can be criticized about the dataset of La Morgia et al. is the setting being artificially simplified by only including the pump and dump event and data from a few hours ago. To test the limits of a detection model, it is therefore necessary to evaluate its robustness with more challenging data. The question that has to be answered is ***RQ2:*** *What performance changes can be observed when our data also embodies events that are not an artificial pushing of the price (pump and dump) but caused by a natural movement?*. For this, we built the NatSD dataset containing natural spikes and drawdowns, as described in section 4.1. The data points in this dataset were all classified as normal, and each data point includes the same features as the original dataset published by La Morgia. This data was randomly split and distributed into the training and testing sets equally. We then trained and optimized our autoencoder models as well as the baseline models on the new dataset and measured the same metrics as beforehand. These are compared with the values measured in the first experiment. The differences then allow conclusions on how well the model can decide between pump and dumps and natural movements.

4.5 Hyperparameter Optimization

To tune the model size and other hyperparameters of our (variational) autoencoders, a random search through the parameter space was conducted. The goal of the search was to minimize the loss (defined as mean squared error loss) of the models throughout the training process. We aimed at keeping the latent dimension as well as the overall model size low while still having the best possible performance. This way, our models can be trained, evaluated, and re-trained in a short period. We also tried to optimize the performance of the Anomaly Transformer and the C-LSTM model by tuning the learning rate and number of epochs. The search space as well as the best configurations for each model can be found in the appendix.

5 RESULTS & DISCUSSION

In Table 1 the performance of the baselines and our models are compared. We also ran Gaussian Naive Bayes to have a purely statistical method to compare against. However, even with the help of the authors, we were not able to achieve the same numbers Chadalapaka et al. [3] published in their paper. The code for their models contained too much uncaught randomness, i.e. in weight initialization for their models. Because of that, we decided to publish the numbers from their paper next to the numbers we were able to produce with their models. Concerning the numbers from the Random Forest model by La Morgia et al. [10], we even outperform the numbers they reported in their paper. However, Chadalapaka et al. report equally strong numbers for the Random Forest, as we achieved.

5.1 Comparison of Different Models

The left column of Table 1 shows the performance of all models on the La Morgia dataset. The best Precision was achieved by the Random Forest model of La Morgia. Our variational autoencoder model was able to achieve the best Recall, while the autoencoder model had the best F1-Macro across all models. If all of the measures are taken into account, the overall best-performing model was the autoencoder, being the only model which achieved more than 90% on all metrics.

The right column of Table 1 shows the performance of the models on the La Morgia dataset combined with our NatSD data. Not surprisingly, the performance dropped for all models in comparison to the pure La Morgia dataset. The Anomaly Transformer and especially our variational autoencoder still have a recall of over 90% but at the expense of the other metrics which dropped drastically. The overall best performance on the NatSD data was achieved by the C-LSTM model which still achieved numbers around 80% on all metrics. The performance of the autoencoder dropped by around 30% for Precision and Recall, and around 17% for F1-Macro.

It is worth taking a look at the hyperparameters of our models. When evaluating the random search we used for hyperparameter optimization, we were not able to identify a correlation between model size and performance. Lightweight models were able to achieve high numbers and keep up with more complex architectures. Our best autoencoder architecture only includes 170 learnable parameters, and the best-performing variational autoencoder includes 282 learnable parameters. Especially in comparison to the C-LSTM (997,851 learnable parameters) and the Anomaly Transformer (155,665 learnable parameters) [3], our models are astonishingly lightweight. Further, we did not find a coherence between the number of neurons in the latent space and the model performance. Our best-performing autoencoder has a latent space of two neurons, and the best-performing variational autoencoder has a latent space of four neurons. We were able to achieve these results with only 20 epochs of training. This allows us to re-train the models very fast without the need for high computational power which is a major benefit compared to the C-LSTM or the Anomaly Transformer. However, it has to be mentioned that also the Random

	La Morgia Data			La Morgia Data + NatSD Data		
	Precision	Recall	F1-Macro	Precision	Recall	F1-Macro
Random Forest (reproduced)	**96.78%**	85.48%	95.39%	62.94%	78.23%	84.87%
Random Forest [10]	91.3%	84.4%	Binary F1: 87.7%			
C-LSTM (reproduced)	96.08%	80.32%	93.74%	81.93%	76.84%	79.03%
C-LSTM [3]	94.2%	84.9%	Binary F1: 89.3%			
Anom. Trans. (reproduced)	84.95%	89.02%	86.79%	37.42%	89.2%	54.34%
Anom. Trans. [3]	93.0%	94.2%	Binary F1: 83.6%			
Gaussian Naive Bayes	5.64%	95.45%	55.09%	8.91%	84.0%	57.92%
Autoencoder	95.16%	93.06%	**97.02%**	60.12%	63.72%	80.73%
Var. Autoencoder	78.48%	**97.79%**	93.46%	32.51%	95.90%	73.75%

Table 1: Comparison of the models developed in this work with baselines on the 15-second sized chunk dataset published by La Morgia et al.

Forest method achieved strong results while being a lightweight method.

5.2 In-Depth Discussion of our Models

Figure 6 shows the reconstruction errors and the latent spaces our trained autoencoder and variational autoencoder models created on the La Morgia dataset. Taking a look at Figures 6a and 6b, the reconstruction errors of the testing set are plotted for the autoencoder model (left) and variational autoencoder (right). The y-axis shows the total reconstruction error for each sample. For better visualization, we randomly generated scatter on the x-axis between -0.1 and 0.1. For the same reason, the y-axis limit for the reconstruction errors was also set manually by us. The maximum reconstruction error measured for the autoencoder was 3.44 and 11.33 for the variational autoencoder. The yellow line shows the threshold that we have calculated for each model to classify points as either normal or anomaly. It can be seen that both models learned how to reconstruct the normal data points quite well, resulting in low reconstruction errors for normal data points.

The variational autoencoder's reconstruction errors for the data points are centered around a minimum of 0.2 for normal data points. We do not have an explanation for why this is the case. For both models, it can be seen that some normal data points do have high reconstruction errors which results in them getting misclassified. Those are usually points in the dataset of La Morgia that are labeled as normal but from the values look like anomalies. An explanation that La Morgia et al. gave is that those could be pump and dumps that were carried out by groups they did not observe and therefore were not aware of when building the dataset. Overall, the threshold separates the normal and anomaly data points well. As desired, by applying an autoencoder approach the normal data points are represented with low reconstruction errors, and anomalies can not be reconstructed well resulting in the high numbers that are reported in Table 1.

Figure 6c shows the latent space of the autoencoder and Figure 6d the latent space of the variational autoencoder. Since our best variational autoencoder model had a latent space of five neurons, principal component analysis was used to reduce the dimensions to two and make the latent space visualizable. This usage of PCA is important to mention here because the normal and anomaly data points do not look like they are separated in Figure 6d. However, they could very well be separated in a higher dimensional space. When comparing the two latent spaces it is visible that the variational autoencoder is far more regularized and converges towards a Gaussian distribution. So the KL loss that was introduced into the model loss worked well. In contrast, the latent space of the autoencoder looks like a straight line. The encoder part of the model learned to represent all normal data points very close to each other and to maximize the anomalies away from the normal data points in one x-y-direction. Also visible in the center of Figure 6c are some normal data points mixed up between the anomaly points. Those normal data points are the same ones that later have high reconstruction errors.

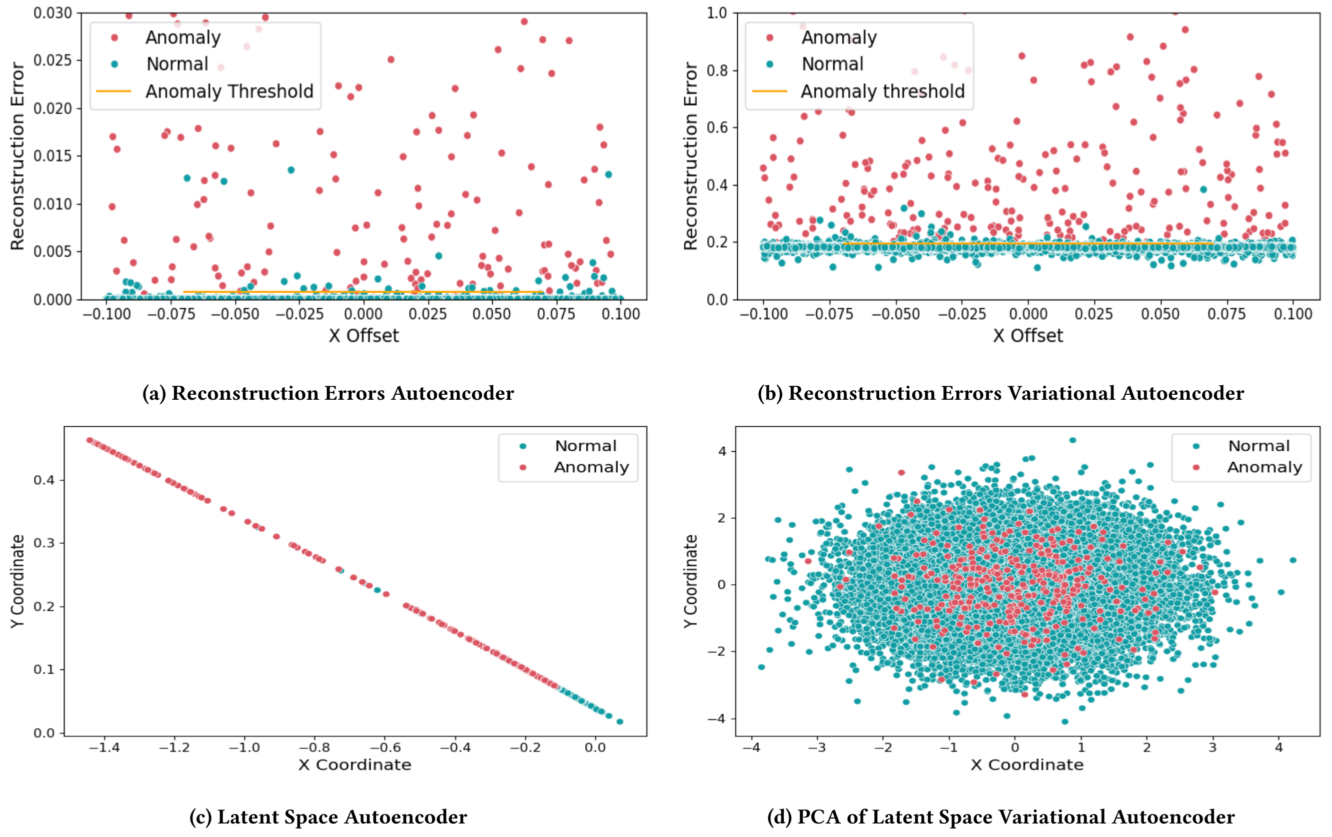
5.3 Influence of Different Features

Figure 7 shows the mean reconstruction errors we calculated as explained in Section 4.3.3. An advantage of our method is that the reconstruction errors are ratio-scaled. Therefore, we can compare the influence of each feature on the final reconstruction directly. The figure shows that the StdVolume feature is by far the most influential when it comes to reconstructing the inputs. Also, StdTrades, AvgPrice, AvgPriceMax, and StdRushOrder have a mediocre influence on the final reconstruction error for a data point. StdPrice, AvgRushOrder, and AvgVolume in general can be reconstructed well over all data points which is why they have a very low influence on the final reconstruction.

It is noteworthy that La Morgia et al. [10] also studied the feature importance of her model. They order the features from most important to least in the following order: StdRushOrder, StdTrades, AvgRushOrder, AvgVolume, StdVolume, StdPrice, AvgPrice, AvgPriceMax. This order greatly differs from the one calculated in our work. Only the StdTrades feature was at second place in our, as well as in their work. StdVolume, AvgPrice, and AvgPriceMax all were four places up in our ranking compared to La Morgias. On the other hand, StdRushOrder, AvgRushOrder, and AvgVolume were down four places in our importance ranking compared to theirs.

5.4 Reflection and Limitations

We showed that with our lightweight autoencoder and variational autoencoder approaches, we were able to hold up against complex



(a) Reconstruction Errors Autoencoder **(b) Reconstruction Errors Variational Autoencoder**

(c) Latent Space Autoencoder **(d) PCA of Latent Space Variational Autoencoder**

Figure 6: Visualizations of reconstruction errors and latent spaces our models created

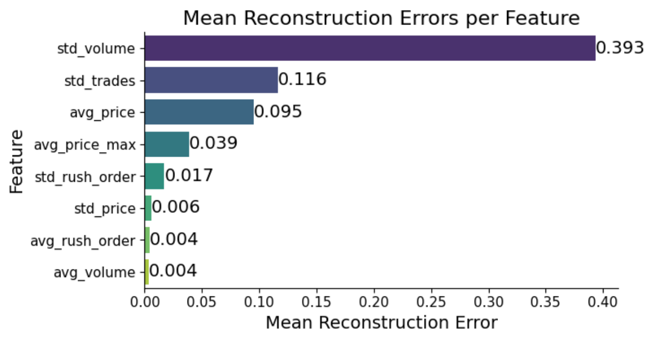


Figure 7: Mean Reconstruction Errors per Feature

deep-learning models for the task of detecting pump and dumps. Also, all the models had a performance loss when including our NatSD data in the La Morgia dataset. This could be because the NatSD inherits a bias but by using the same code La Morgia et al. used to calculate their features, the probability of a bias is minimized. Further, it might be possible that the effect size is stronger or weaker than reflected by the numbers. For example, the ~580,000 data points from the La Morgia dataset could suppress the effect of only 3,040 NatSD data points. We attempted to control this effect by equally distributing the NatSD data points into the train and test sets. Thus, we have shown that the La Morgia dataset depicts the pump and dump detection in a way that makes the problem easier than it is in the real world. Therefore our NatSD data is crucial for testing how good models and features are really for detecting pump and dumps.

Another possible threat to the validity of this research is the trustworthiness of the APIs used to download the price or order book data (Binance APIs), as well as the transaction history (Alchemy API). Both Binance and Alchemy are well-trusted companies that operate globally and have their APIs as a major part of their businesses. Their APIs are used by many other companies and providing reliable data is important for their success.

In terms of generalization, it needs to be discussed whether our results can be generalized onto other blockchain implementations or also onto pump and dump events not carried out on the Binance trading platform. Concerning the performance analysis of the models we conducted, the data stems from Binance order books. Therefore, it is not dependent on the blockchain on which the respective cryptocurrency is implemented on, but on the trading platform the event is carried out. As long as the order books between platforms have a similar structure this causes no threat. The primary limitation of using order book data in detecting cryptocurrency pump and dump schemes, lies in its heavy reliance on trading platforms or exchanges for high qualitative data. This dependency creates a significant constraint, as among the examined exchanges, only Binance offers accessible order book data. This scenario limits the scope and diversity of the data and limits real-world usage of pump and dump detection drastically.

6 CONCLUSION

Our experiments show that autoencoders can effectively detect pump and dumps based on order book data. The results prove that lightweight models can perform equally well or even better than complex models while having the benefit that they can be re-trained extremely fast. Further, we showed that all of the models exhibit drastic performance losses when confronted with our NatSD dataset which reflects a more realistic market scenario. This result is especially grave when discussing how pump and dump detection model could be used for real-world operation.

To improve model robustness against the NatSD data, further research work needs to be done. Also, this work is mainly focused on how pump and dumps on centralized exchanges behave and can be detected. It would be interesting to further investigate how pump and dumps carried out on decentralized exchanges behave and are depicted in the data. A dataset for pump and dump events carried out on decentralized exchanges is (as of today) missing and needs to be created. With that, the search for effective features would be enabled. Such an analysis could help decouple the current state of pump and dump detection from the dependency on the exchange platform's order book data which would be a big step forward when using pump and dump detection models for real-world use cases.

ACKNOWLEDGMENTS

The authors acknowledge support by the state of Baden-Württemberg through bwHPC.

Further, Marcel Hoffmann and Nicolas Lell are acknowledged for their helpful feedback and tips for hyperparameter optimization.

REFERENCES

[1] Jinwon An and Sungzoon Cho. 2015. Variational autoencoder based anomaly detection using reconstruction probability. *Special lecture on IE* 2, 1 (2015), 1–18.

[2] Dor Bank, Noam Koenigstein, and Raja Giryes. 2023. Autoencoders. *Machine Learning for Data Science Handbook: Data Mining and Knowledge Discovery Handbook* (2023), 353–374.

[3] Viswanath Chadalapaka, Kyle Chang, Gireesh Mahajan, and Anuj Vasil. 2022. Crypto Pump and Dump Detection via Deep Learning Techniques. arXiv:2205.04646 [cs.LG]

[4] European Parliament. 2014. *Regulation (EU) No 596/2014 of the European Parliament and of the Council of 16 April 2014 on market abuse (Market Abuse Regulation) (MAR).* https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:32014R0596

[5] European Parliament. 2023. *Regulation (EU) 2023/1114 of the European Parliament and of the Council of 26 June 2023 on a framework for the recovery and resolution of central counterparties and amending Regulations (EU) No 1095/2010, (EU) No 648/2012, and (EU) No 600/2014.* https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:32023R1114

[6] Manon Patrizia Grimm and Shajan Kreuter. 2023. Kryptowerte und Marktmissbrauch. *AG* (2023), 177–189.

[7] Sihao Hu, Zhen Zhang, Shengliang Lu, Bingsheng He, and Zhao Li. 2023. Sequence-Based Target Coin Prediction for Cryptocurrency Pump-and-Dump. *Proc. ACM Manag. Data* 1, 1, Article 6 (may 2023), 19 pages. https://doi.org/10.1145/3588686

[8] Josh Kamps and Bennett Kleinberg. 2018. To the moon: defining and detecting cryptocurrency pump-and-dumps. *Crime Science* 7, 1 (2018), 1–18.

[9] Diederik P Kingma and Max Welling. 2022. Auto-Encoding Variational Bayes. arXiv:1312.6114 [stat.ML]

[10] M. La Morgia, A. Mei, F. Sassi, and J. Stefa. 2020. Pump and Dumps in the Bitcoin Era: Real Time Detection of Cryptocurrency Market Manipulations. In *2020 29th International Conference on Computer Communications and Networks (ICCCN).* 1–9. https://doi.org/10.1109/ICCCN49398.2020.9209660

[11] Marion Laboure and Reid Jim. 2020. The future of payments-part iii. digital currencies: The ultimate hard power tool. *Deutsche Bank Research* (2020).

[12] Tao Li, Donghwa Shin, and Baolian Wang. 2018. Cryptocurrency Pump-and-Dump Schemes. *SSRN Electronic Journal* (01 2018). https://doi.org/10.2139/ssrn.3267041

[13] Joana Lorenz, Maria Inês Silva, David Aparício, João Tiago Ascensão, and Pedro Bizarro. 2021. Machine Learning Methods to Detect Money Laundering in the Bitcoin Blockchain in the Presence of Label Scarcity. In *Proceedings of the First ACM International Conference on AI in Finance* (New York, New York) *(ICAIF '20)*. Association for Computing Machinery, New York, NY, USA, Article 23, 8 pages. https://doi.org/10.1145/3383455.3422549

[14] Satoshi Nakamoto. 2008. Bitcoin: A Peer-to-Peer Electronic Cash System. (2008).

[15] Huy Nghiem, Goran Muric, Fred Morstatter, and Emilio Ferrara. 2021. Detecting cryptocurrency pump-and-dump frauds using market and social signals. *Expert Systems with Applications* 182 (2021), 115284. https://doi.org/10.1016/j.eswa.2021.115284

[16] Huy Nghiem, Goran Muric, Fred Morstatter, and Emilio Ferrara. 2021. Detecting cryptocurrency pump-and-dump frauds using market and social signals. *Expert Systems with Applications* 182 (2021), 115284. https://doi.org/10.1016/j.eswa.2021.115284

[17] Trishie Sharma, Rachit Agarwal, and Sandeep Shukla. 2023. Understanding Rug Pulls: An In-Depth Behavioral Analysis of Fraudulent NFT Creators.

[18] Friedhelm Victor and Tanja Hagemann. 2019. Cryptocurrency Pump and Dump Schemes: Quantification and Detection. In *2019 International Conference on Data Mining Workshops (ICDMW).* 244–251. https://doi.org/10.1109/ICDMW.2019.00045

[19] Haowen Xu, Wenxiao Chen, Nengwen Zhao, Zeyan Li, Jiahao Bu, Zhihan Li, Ying Liu, Youjian Zhao, Dan Pei, Yang Feng, Jie Chen, Zhaogang Wang, and Honglin Qiao. 2018. Unsupervised Anomaly Detection via Variational Auto-Encoder for Seasonal KPIs in Web Applications. In *Proceedings of the 2018 World Wide Web Conference* (Lyon, France) *(WWW '18)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, 187–196. https://doi.org/10.1145/3178876.3185996

[20] Jiahua Xu and Benjamin Livshits. 2019. The Anatomy of a Cryptocurrency Pump-and-Dump Scheme. In *Proceedings of the 28th USENIX Conference on Security Symposium* (Santa Clara, CA, USA) *(SEC'19)*. USENIX Association, USA, 1609–1625.

[21] Jiehui Xu, Haixu Wu, Jianmin Wang, and Mingsheng Long. 2022. Anomaly Transformer: Time Series Anomaly Detection with Association Discrepancy. arXiv:2110.02642 [cs.LG]

[22] Bo Zong, Qi Song, Martin Renqiang Min, Wei Cheng, Cristian Lumezanu, Daeki Cho, and Haifeng Chen. 2018. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In *International conference on learning representations.*

A HYPERPARAMETER OPTIMIZATION

The parameter space for the optimization of our autoencoder and variational autoencoder was defined as follows:

- learning rate: 0.1, 0.01, 0.001, 0.0001
- latent dimension: 1, 2, 3, 4, 5
- epochs: 20, 50, 100, 200
- batch size: random samples from uniform distribution between 500 - 2000
- activation functions: linear activation, relu, softmax
- architecture: we compared different possible architectures varying between number of layers and number of neurons per layer
- kullback–leibler divergence loss weighting (only variational autoencoder): random samples from uniform distribution between 0.5 - 1.5

For the Anomaly Transformer and the C-LSTM we optimized the leraning rate (0.01, 0.001, 0.001 and also the number of epochs. The best hyperparameter combination for our autoencoder was:

- learning rate: 0.001
- latent dimension: 2
- epochs: 20
- batch size: 789
- activation functions: softmax
- architecture: 8 input neurons, 4 neurons first hidden layer, 2 neurons latent dimension, 4 neurons second hidden layer, 8 output neurons

The best hyperparameter combination for our variational autoencoder was:

- learning rate: 0.01
- latent dimension: 4
- epochs: 50
- batch size: 1712
- activation functions: softmax
- architecture: 8 input neurons, 7 neurons first hidden layer, 5 neurons second hidden layer, 4 neurons latent dimension, 5 neurons third hidden layer, 7 neurons fourth hidden layer, 8 output neurons

B QUALITATIVE ANALYSIS OF TRANSACTIONS DURING PUMP AND DUMPS (RQ3)

In literature, the used data for pump and dump detection often derives from order book data [18] which the respective asset exchange platform made public. However, this creates dependence on the exchange platform. The goal of this section is to discuss how on-chain data (e.g. asset transfer history), and platform-independent data (e.g. price data) could be used for detection.

RQ3 *How can the transaction data realized on the blockchain be utilized to detect pump and dumps?*

B.1 Data

The approach we followed was to design features that could separate pump and dump data patterns from regular data by using two dimensions. The first dimension consists of features that are characteristic for spikes, for example, the change of return rate (or the price change) or the total number of trades. Although price and return are not stored in transactions on the Ethereum blockchain, those are still features that are not dependent on a particular exchange platform. The second dimension includes features that can separate pump and dumps from natural spikes. Features identified for this are the number of wallets that have participated in a previously confirmed pump and dump event or the mean and standard deviation of the wallet ages currently participating in trades. We propose that a combination of those features can be a good indicator for pump and dump detection.

To test this idea, the confirmed pump and dump events from the dataset of La Morgia et al. could be taken. The necessary data to calculate the proposed features could be downloaded using the Binance historical prices API, as well as the Alchemy full-node API[4], which provides ease of use for on-chain data.

[4]https://www.alchemy.com/

B.2 Method

As discussed above, it is important to research how well data from public open sources, like e.g. on-chain data or price data represents pump and dump events. To answer this question, we conduct a qualitative analysis of the transactions written onto the blockchain during pump and dump events.

The idea is to analyze how many transactions were written onto the Ethereum blockchain for a specific cryptocurrency which was the target of the pump and dump event. Hypothetically, the bare number of transactions will rise and fall in a manner that follows the behavior of the price of the targeted asset during a pump and dump.

B.3 Results & Discussion

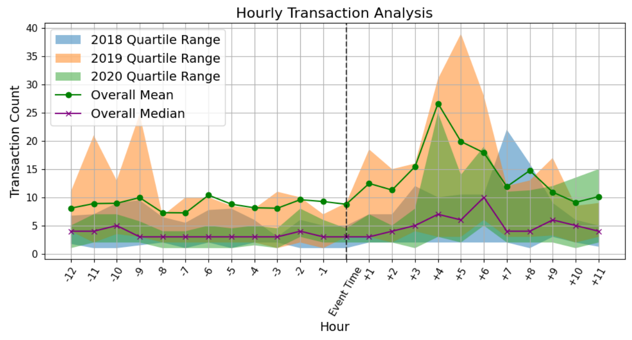


Figure 8: Counted transactions in a one-day window around the pump and dump events

In Figure 8, the transactions visible on the blockchain for a window of twelve hours before and after an event are visualized. The transactions were downloaded for the 317 pump and dumps carried out on the Binance platform (the same events the La Morgia dataset consists of) using the Alchemy API. The number of transactions per hour is calculated and the interquartile range per year as well as the overall mean and median are visualized. Interestingly, there are very few transactions written on the blockchain, with the median moving between only 3 to 10 transactions per hour throughout the measured time. This contradicts the initial presumption which was that transaction numbers would also rise and fall during a pump and dump event. The explanation for this is that centralized trading platforms (like Binance is one) do not reflect every trade performed on them onto the blockchain. Instead, they only move traded assets between the accounts a user has on their platform. Only when a user moves his assets from the platform to his private wallet, or vice versa, the transaction is visible on the blockchain itself.

Another interesting observation from Figure 8 is that there is a peak in transactions visible a few hours after the pump and dump events. We double-checked that this is not caused by the API data being in a different timezone than the events and therefore the whole plot being shifted by the timezone difference. A possible explanation for this peak might be that the participants of the pump and dump move coins they bought but not sold again into their private wallets. The transactions would go from the trading platforms into private wallets and therefore become visible after the pump. Since Ethereum can hold transactions with low priority in its transaction pools for a while, this could explain the time difference to the event. Yet, this is only a hypothesis and further research has to be done to prove or refute it.

Also unexpected is that there are no changes in the transaction numbers over the years. 2019 has a slightly broader IQR than the other years, but 2020 and 2018 are looking almost equal. Further analysis is required to explain this.

B.4 Conclusion and Limitations

In section 3, we motivated the problem of the La Morgia dataset consisting of features partly calculated from order book data and therefore being dependent on trading platforms open-sourcing their data. To tackle this problem, we have proposed that the transaction history on blockchains is inherently open-sourced and may be a good alternative to decouple pump and dump detection from the trading platforms. However, the qualitative analysis of the transaction data has shown that the transactions do not reflect the typical pump and dump behavior. There is a big difference in how pump and dumps are reflected in order book data compared to the transactions carried out to the blockchain. It may still be possible to calculate features based on the transactions realized on the blockchain. An example could be by analyzing the wallets that transactions come from or go to. Further research is necessary to conclude this.

Further, there may be a difference in how transactions are logged on different blockchains which may lead to different results when it comes to analyzing the transactions. We have proven that centralized trading platforms do not reflect trades made on them onto the blockchain. Therefore, pump and dump events are not depicted in the transaction history independent of the blockchain underneath. Still, the results may differ when analyzing pump and dumps carried out on decentralized exchanges.