Noname manuscript No.
(will be inserted by the editor)

Temporal mixture ensemble models for intraday volume forecasting in cryptocurrency exchange markets

Nino Antulov-Fantulin* · Tian Guo* · Fabrizio Lillo

Received: date / Accepted: date

Abstract We study the problem of the intraday short-term volume forecasting in cryptocurrency exchange markets. The predictions are built by using transaction and order book data from different markets where the exchange takes place. Methodologically, we propose a temporal mixture ensemble model, capable of adaptively exploiting, for the forecasting, different sources of data and providing a volume point estimate, as well as its uncertainty. We provide evidence of the outperformance of our model by comparing its outcomes with those obtained with different time series and machine learning methods. Finally, we discuss the difficulty of volume forecasting when large quantities are abruptly traded.

1 Introduction

Cryptocurrencies recently attracted massive attention from researchers in several disciplines such as finance, economics, and computer science. It originated from a decentralized peer-to-peer payment platform through the Internet. When new transactions are announced on this network, they have to be verified by network nodes and recorded in a public distributed ledger called the blockchain [1]. Cryptocurrencies are created as a reward in the verification

*Shared first authorship.
N. Antulov-Fantulin
ETH Zurich
Aisot GmbH, Zurich, Switzerland

T. Guo
RAM Active Investments, Switzerland
Work done when at ETH Zurich

F. Lillo
Department of Mathematics
University of Bologna, Italy
E-mail: fabrizio.lillo@unibo.it

competition in which users offer their computing power to verify and record transactions into the blockchain. Bitcoin is one of the most prominent decentralized digital cryptocurrencies and it is the focus of this paper, although the model developed below can be adapted to other cryptocurrencies with ease, as well as to other "ordinary" assets (equities, futures, FX rates, etc.).

The exchange of Bitcoins with other fiat or cryptocurrencies takes place on exchange markets, which share some similarities with the foreign exchange markets [2]. These markets typically work through a continuous double auction, which is implemented with a limit order book mechanism, where no designated dealer or market maker is present and limit and market orders to buy and sell arrive continuously. Moreover, as observed for traditional assets, the market is *fragmented*, i.e. there are several exchanges where the trading of the same asset, in our case the exchange of a cryptocurrency with a fiat currency, can simultaneously take place.

The automation of the (cryptocurrency) exchanges lead to the increase of the use of automated trading [3, 4] via different trading algorithms. An important input for these algos is the prediction of future trading volume. This is important for several reasons. First, trading volume is a proxy for liquidity which in turn is important to quantify transaction costs. Trading algorithms aim at minimizing these costs by splitting orders in order to find a better execution price [5, 6] and the crucial part is the decision of when to execute the orders in such a way to minimize market impact or to achieve certain trading benchmarks (e.g. VWAP) [7–9]. Second, when different market venues are available, the algorithm must decide where to post the order and the choice is likely the market where more volume is predicted to be available. Third, volume is also used to model the time-varying price volatility process, whose relation is also known as Mixture of Distribution Hypothesis" [10].

In this paper, we study the problem of intraday short-term volume prediction on cryptocurrency markets, intending to obtain not only point estimate but also an interval of uncertainty [11–14]. Moreover, conventional volume predictions focuses on using data or features from the same market. Since cryptocurrency markets are traded on several markets simultaneously, it is reasonable to use cross-market data not only to enhance the predictive power, but also to help understanding the interaction between markets. In particular, we investigate the exchange rate of Bitcoin (BTC) with a fiat currency (USD) on two liquid markets: Bitfinex and Bitstamp. The first market is more liquid than the second, since its traded volume in the investigated period is 2.5 times larger[1]. Thus one expects an asymmetric role of the past volume (or other market variables) of one market on the prediction of volume in the other market. We propose a class of models, termed temporal mixture ensemble models, to build predictions of volume and we compare out-of-sample forecasts with those obtained with some traditional time-series approaches and with a machine learning benchmark (gradient boosting).

[1] Recently, there have been few reports that are showing fake reported volume for certain Bitcoin exchange markets. In this study, we are working with Bitcoin exchange markets that have been independently verified to report true values [15].

Specifically, the contribution of this paper can be summarized as follows:

- We formulate the cross-market volume prediction as a supervised multi-source learning problem. We use multi-source data, i.e. transactions and limit order books from different markets, to predict the volume of the target market.
- We propose the temporal mixture ensemble model, which models individual source's relation to the target and adaptively adjusts the contribution of the individual source to the target prediction.
- By equipping with modern ensemble techniques, the proposed model can further quantify the predictive uncertainty consisting of the epistemic and aleatoric component, for volume and source contributions.
- As main benchmarks for volume dynamics, we use different time-series and machine learning models (clearly with the same regressors/features used in our model). We observe that our dynamic mixture ensemble is often having superior out-of-sample performance on conventional prediction error metrics e.g. root mean square error (RMSE) and mean absolute error (MAE). More importantly, it presents much better calibrated results, evaluated by metrics taking into account predictive uncertainty, i.e. normalized negative log-likelihood (NNLL), uncertainty interval coverage (IC) and width (IW).
- We show that the main difficulty in predicting volume (for all models) is related to very large and unexpected volumes. Outside these situations, our model strongly outperforms the other benchmarks.

The paper is organized as follows: in Section 2 we present the investigated markets, the data, and the variables used in the modeling. In Section 3 we present our benchmark models. In Section 4 we present our empirical investigations on the cryptocurrency markets for the prediction of intraday market volume. Finally, Section 5 presents some conclusions and outlook for future work.

2 Multiple market cryptocurrency data

Our empirical analyses are performed on a sample of data from two exchanges, Bitfinex[2] and Bitstamp[3], where Bitcoins can be exchanged with US dollars. These markets work through a limit order book, as many conventional exchanges. The investigated period is June-November 2018. The main analyses are performed on five-minute intervals, thus the length of the investigated time series is $34,346$, since the markets are open $24/7$. In the Appendix we also show some analyses performed at one minute resolution, raising the number of data points to 171k.

For each of the two markets we consider two types of data: transaction data and limit order book data.

[2] https://www.bitfinex.com
[3] https://www.bitstamp.net

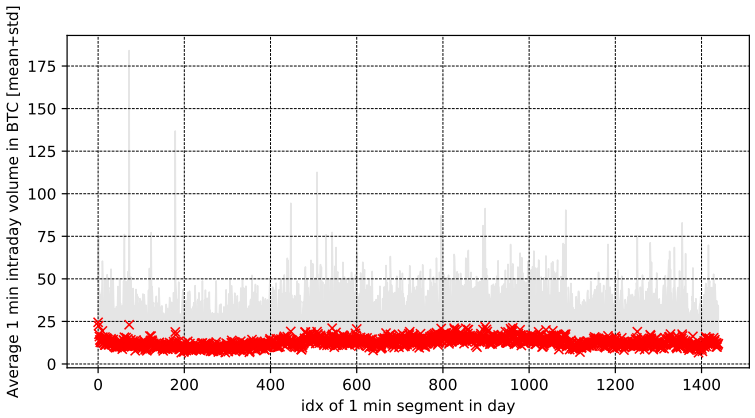

Fig. 1: The intraday average 1-min transaction volume plus 1 std of BTC/USD rate in Bitfinex market over the period from June 2018 to November 2018.

From **transaction data** we extract the following features for each 5-min interval:

- *Buy volume* - in BTC units of executed transactions
- *Sell volume* - in BTC units of executed transactions
- *Volume imbalance* - absolute difference between buy and sell volume
- *Buy transactions* - number of executed transactions on buy side
- *Sell transactions* - number of executed transactions on sell side
- *Transaction imbalance* - absolute difference between buy and sell number of transactions

From **limit orderbook data** we extract the following features [16, 17], obtained by averaging the one minute variables in each 5-min interval:

- *Spread* is the difference between the highest price that a buyer is willing to pay for a BTC (bid) and the lowest price that a seller is willing to accept (ask).
- *Ask volume* is the number of BTCs on the ask side of order book.
- *Bid volume* is the number of BTCs on the bid side of order book.
- *Imbalance* is the absolute difference between ask and bid volume.
- *Ask/bid Slope* is estimated as the volume until δ price offset from the best ask/bid price. δ is estimated by the bid price at the order that has at least 1%, 5% and 10 % of orders with the highest bid price.
- *Slope imbalance* is the absolute difference between ask and bid slope at different values of price associated to δ. δ is estimated by the bid price at the order that has at least 1%, 5% and 10 % of orders with the highest bid price.

The target variable y_t that we aim at forecasting is the sum of the two first features of transaction data of the target market, i.e. the sum of buy and sell volume.

In the proposed modeling approaches (described in Section 3) we consider different sources that at each time can affect the probability distribution of trading volume in the next time interval in a given market. Given the setting presented above, in our analysis, there are $S = 4$ sources, namely one for transaction data and one for limit order book data for the two markets. We indicate with $\mathbf{x}_{s,t} \in \mathbb{R}^{d_s}$ the data from source s at time step t, and d_s the dimensionality of source data s. Given the list of variables presented above, we have $d_s = 6$ when the source is transaction data in any market, while $d_s = 13$ for orderbook data.

Figure 1 shows the average 1-min transaction volume as a function of the time of the day. We observe the lack of a strong intra-daily "U-shape" component, which is instead observed in other asset classes [18] and used in intraday volume modeling.

3 Models

Econometric modeling of intra-daily trading volume relies on a set of empirical regularities [7–9] of volume dynamics. These include fat tails, strong persistence and an intra-daily clustering around the "U"-shaped periodic component. Brownlees et al. [7] proposed Component Multiplicative Error Model (CMEM), which is the extension of Multiplicative Error Model (MEM) [19]. The CMEM volume model has a connection to the component-GARCH [20] and the periodic P-GARCH [21]. Satish et al. [8], proposed four-component volume forecast model composed of: (i) rolling historical volume average, (ii) daily ARMA for serial correlation across daily volumes, (iii) deseasonalized intra-day ARMA volume model and (iv) a dynamic weighted combination of previous models. Chen et al. [9], simplify the multiplicative volume model [7] into an additive one by modeling the logarithm of intraday volume with the Kalman filter.

Given the lack of intraday periodicity (see Fig. 1) in the investigated cryptocurrency markets and the need of adding external multi-source regressors, our econometric benchmarks are AR-GARCH type models, described in the next subsection. The machine learning benchmark is the gradient boosting method, while our main model is the temporal mixture ensemble model, which naturally allows using multi-source data. When multi-source temporal data are from different sources, $\mathbf{x}_{s,t}$ indicates the data from source s at time step t and $\mathbf{x}_{s,t}$ could be multi-dimensional, i.e. $\mathbf{x}_{s,t} \in \mathbb{R}^{d_s}$, where d_s is the data dimensionality of source s. With these multi-source data, we aim at predicting the future value of the target variable y_t.

In this paper, the target variable is the 5 min trading volume in one of the two cryptocurrency markets using data from both markets. Regarding the multi-source data, on one hand, it includes the feature time series from

the target market. This data is believed to be directly correlated with the target variable. On the other hand, there is an alternative market, which could interact with the target market. Together with the target market, the feature time series of this alternative market constitute the multi-source data. For each market, we use features from both transaction and order book data. The experiment section 4 provides more details about markets, transactions, order book data, and features.

3.1 Econometric benchmarks

As mentioned, our benchmarks belong to the AR-GARCH class with external regressors. More specifically, the volume process y_t is modelled with the following autoregressive process (AR(p)) with external regressors:

$$y_t = \mu + \sum_{i=1}^{p} \phi_i y_{t-i} + \sum_{s=1}^{S} \sum_{j=1}^{d_s} \psi_{s,j} \mathbf{x}_{s,t-1}(j) + \epsilon_t, \tag{1}$$

where $\mathbf{x}_{s,t-1}(j)$ denotes the j-th feature from external feature vector $\mathbf{x}_{s,t-1}$ at time[4] $t-1$ from source s. The total number of sources $S = 4$, which includes transactions and limit order book data of the two markets. Since volume exhibits time clustering, we assume that the residuals ϵ_t are modelled by a GARCH process [7–9]:

$$\epsilon_t = \sigma_t e_t \qquad e_t \sim \mathcal{N}(0,1) \tag{2}$$

$$\sigma_t^{\lambda} = \omega + \alpha \epsilon_{t-1}^{\lambda} + \gamma |\epsilon_{t-1}|^{\lambda} \mathbb{1}_{[\epsilon_{t-1}<0]} + \beta \sigma_{t-1}^{\lambda} \tag{3}$$

For $\lambda = 2$, $\gamma = 0$, we get standard GARCH(1,1) model [22]. In case of $\lambda = 2$, $\gamma = 1$, we get GJR-GARCH model [23], that captures asymmetry in positive and negative shocks[5].

3.2 Machine learning benchmark

We take the gradient boosting machine [25] as a machine learning baseline. Gradient boosting approximates the volume $\hat{y}_t = F(\mathbf{x}_t)$ with a function that has the following additive expansion (similar to other functional approximation methods like radial basis functions, neural networks, wavelets, etc.):

$$\hat{y}_t = F(\mathbf{x}_t) = \sum_{m=0}^{M} \beta_m h(\mathbf{x}_t; \mathbf{a}_m), \tag{4}$$

[4] Note, that we have also evaluated all ARX-GARCH models by using autoregressive external features $\{\mathbf{x}_{s,t-i}(j)\}_{i=1}^{p}$ terms, but results were not better and training time and convergence become problematic.

[5] We also tested the case of $\lambda = 1$, $\gamma = 1$ corresponding to the Threshold heteroskedastic models [24], but the model displays sometimes convergence problems, thus we decided not to present it. Anyhow the forecasting ability of this model is comparable to that of the other GARCH models.

where $\mathbf{x}_t$ denotes the feature vector, that is constructed as a concatenation from different sources[6] $\mathbf{x}_t = (\mathbf{x}_{s=1,t}, \mathbf{x}_{s=2,t}, \mathbf{x}_{s=3,t}, \mathbf{x}_{s=4,t})$.

The functions $h(\mathbf{x}_t; \mathbf{a}_m)$ are also called "base learners" and in our case they are regression trees with parameters $\mathbf{a}_m$ and β_m is a simple scalar.

Each base learner $h(\mathbf{x}_t; \mathbf{a}_m)$ partitions the feature space $\mathbf{x}_t \in \mathbf{X}$ into L-disjoint regions $\{R_{l,m}\}_1^L$ and predicts a separate constant value in each:

$$h(\mathbf{x}_t; \{R_{l,m}\}_1^L) = \sum_{l=1}^{L} \bar{y}_{l,m} \mathbf{1}(\mathbf{x}_t \in R_{l,m}), \tag{5}$$

where $\bar{y}_{l,m}$ is inferred during the learning phase along with the expansion coefficients $\{\beta_m\}$ and the parameters of regression trees $\mathbf{a}_m$. Learning procedure starts by defining the loss function $\Psi(y_t, F(\mathbf{x}_t))$ e.g. squared loss $\sum_t (y_t - F(\mathbf{x}_t))^2$ and initial regression tree $F_0(\mathbf{x}_t)$. Then, for each $m = 1, ..., M$, we solve the optimization problem:

$$(\beta_m, \mathbf{a}_m) = \arg\min_{\beta, \mathbf{a}} \sum_{t=1}^{T} \Psi(y_t, F_{m-1}(\mathbf{x}_t) + \beta h(\mathbf{x}_t; \mathbf{a})) \tag{6}$$

and

$$F_m(\mathbf{x}_t) = F_{m-1}(\mathbf{x}_t) + \beta_m h(\mathbf{x}_t; \mathbf{a}_m). \tag{7}$$

See Appendix for more details. Furthermore, note that different variants of tree boosting have been empirically proven to be state-of-the-art methods in predictive tasks across different machine learning challenges [26, 27] and more recently in finance [28, 29].

3.3 Temporal mixture ensemble

In this paper, we construct an intra-daily dynamic mixture ensemble model, belonging to the class of of mixture models [30–34], that takes previous transactions and limit order book data [16,17] from multiple markets simultaneously into account. Though mixture models have been widely used in machine learning and deep learning [35–37], they have been hardly explored for prediction tasks in cryptocurrency markets. Moreover, our proposed temporal mixture ensemble can provide predictive uncertainty of the target volume by the use of Stochastic Gradient Descent (SGD) based ensemble techniques [38–40]. Predictive uncertainty reflects the confidence of the model over the prediction. It is valuable extra information for model interpretability and reliability.

In principle, the temporal mixture ensemble exploits latent variables to capture the contributions of different sources of data to the future evolution of the target variable. The source contributing at a certain time depends on the history of all the sources.

[6] Note, that we have omitted the transpose operators in the next line, as the concatenation is simple operation and to avoid confusion with index of time.

More quantitatively, the generative process of the time series of the target variable conditional on multi-source data $\{\mathbf{x}_{1,t}, \cdots, \mathbf{x}_{S,t}\}_{t=0}^{T}$ is formulated as the following probabilistic mixture process:

$$\begin{aligned} & p(y_1, \cdots, y_T | \{\mathbf{x}_{1,t}, \cdots, \mathbf{x}_{S,t}\}_{t=0}^{T}) \\ = & \sum_{z_1} \cdots \sum_{z_T} p(y_1, \cdots, y_T, z_1, \cdots, z_T \mid \{\mathbf{x}_{1,t}, \cdots, \mathbf{x}_{S,t}\}_{t=0}^{T}) \\ = & \prod_{t} \sum_{z_t=1}^{S} p_\theta(y_t | z_t = s, \mathbf{x}_{s,<t}) \cdot \mathbb{P}_\omega(z_t = s \,|\mathbf{x}_{1,<t}, \cdots, \mathbf{x}_{S,<t}). \end{aligned} \tag{8}$$

The latent variable z_t is a discrete random variable defined on the set of values $\{1, \cdots, S\}$, each of which represents the corresponding data source. The quantity $p_\theta(y_t|z_t = s, \mathbf{x}_{s,<t})$ models the predictive probabilistic density of the target based on the historical data $\mathbf{x}_{s,<t}$ from a certain source s. The quantity[7] $\mathbb{P}_\omega(z_t = s \,|\mathbf{x}_{1,<t}, \cdots, \mathbf{x}_{S,<t})$ is time-varying dependent on multi-source data and adaptively adjusts the contribution of the data source specific density $p_\theta(y_t|z_t = s, \mathbf{x}_{s,<t})$ at each time step. Clearly, it holds $\sum_{s=1}^{S} \mathbb{P}_\omega(z_t = s \,|\mathbf{x}_{1,<t}, \cdots, \mathbf{x}_{S,<t}) = 1$. Finally, θ and ω in Eq. 8 represent the parameters of the probabilistic functions, which are learned in the training phase discussed below.

In the following, we will first present the inference procedure by assuming the posterior distribution of model parameters given training data. This inference process gives rise to various predictions on mean and uncertainties of the target variable. Then we will describe the learning algorithms to obtain the posterior.

3.4 Inference

In this part, we present Bayesian style inference assuming given samples of model parameters. In Section 3.5, we will describe approximate samples of model parameters by ensemble methods. Bayesian inference gives rise to a set of realizations of the models by the posterior distribution of model parameters. Through a probabilistic ensemble of these model realizations, we harvest accurate predictions as well as additional insights into the data and model. This will also be demonstrated in the experiment section.

Specifically, we denote by $\Theta = \{\theta, \omega\}$ the overall set of parameters in the mixture model. The training data is referred to as $\mathcal{D} = \{y_t, \mathbf{x}_{1,t}, \cdots, \mathbf{x}_{S,t}\}_{t=1}^{T}$. The posterior distribution of Θ given $\mathcal{D}$ is defined as:

$$p(\Theta|\mathcal{D}) \propto p_\Theta(y_1, \cdots, y_T | \{\mathbf{x}_{1,t}, \cdots, \mathbf{x}_{S,t}\}_{t=0}^{T}) \cdot p(\Theta), \tag{9}$$

where $p_\Theta(y_1, \cdots, y_T | \{\mathbf{x}_{1,t}, \cdots, \mathbf{x}_{S,t}\}_{t=0}^{T})$ is defined in Eq. 8 and $p(\Theta)$ is the prior.

[7] We indicate with p_θ probability densities and $\mathbb{P}_\omega$ probability mass functions.

Given $p(\Theta|\mathcal{D})$, the predictive density of y_τ is derived by integrating over Θ as:

$$\begin{aligned} & p(y_\tau|\{\mathbf{x}_{1,<\tau},\cdots,\mathbf{x}_{S,<\tau}\},\mathcal{D}) \\ =&\int_\Theta p_\Theta(y_\tau|\{\mathbf{x}_{1,<\tau},\cdots,\mathbf{x}_{S,<\tau}\})\cdot p(\Theta|\mathcal{D})d\Theta \\ =&\frac{1}{M}\sum_{m=1}^{M} p_{\Theta_m}(y_\tau|\{\mathbf{x}_{1,<\tau},\cdots,\mathbf{x}_{S,<\tau}\}), \end{aligned} \tag{10}$$

where Θ_m is a sample from the posterior $p(\Theta|\mathcal{D})$, i.e. $\Theta_m = \{\theta_m, \omega_m\} \sim p(\Theta|\mathcal{D})$.

Predictive mean. The typical prediction on the target y_τ is the expected value, i.e. the conditional mean. In our temporal mixture model, it is derived as:

$$\begin{aligned} & \mathbb{E}[y_\tau|\{\mathbf{x}_{1,<\tau},\cdots,\mathbf{x}_{S,<\tau}\},\mathcal{D}] \\ =&\int_y y\cdot p(y|\{\mathbf{x}_{1,<\tau},\cdots,\mathbf{x}_{S,<\tau}\},\mathcal{D})dy \\ =&\int_y\int_\Theta y\cdot p_\Theta(\{\mathbf{x}_{1,<\tau},\cdots,\mathbf{x}_{S,<\tau}\})\cdot p(\Theta|\mathcal{D})dyd\Theta \\ =&\frac{1}{M}\sum_{m=1}^{M}\mathbb{E}[y_\tau|\mathbf{x}_{1,<\tau},\cdots,\mathbf{x}_{S,<\tau},\Theta_m], \\ & \text{where } \Theta_m \sim p(\Theta|\mathcal{D}) \end{aligned} \tag{11}$$

In Eq. 11, we use Monte Carlo methods to obtain unbiased estimates of the integral on model parameters. $\mathbb{E}[y_\tau|\mathbf{x}_{1,<\tau},\cdots,\mathbf{x}_{S,<\tau},\Theta_m]$ is the conditional mean given one realization Θ_m of the model parameters. In the context of temporal mixture models, it is derived as:

$$\begin{aligned} & \mathbb{E}[y_\tau|\mathbf{x}_{1,<\tau},\cdots,\mathbf{x}_{S,<\tau},\Theta_m] \\ =&\sum_{s=1}^{S}\mathbb{P}_{\omega_m}(z_\tau=s|\mathbf{x}_{1,<\tau},\cdots,\mathbf{x}_{S,<\tau})\cdot\mathbb{E}[y_\tau|\mathbf{x}_{s,<\tau},\theta_m] \end{aligned} \tag{12}$$

Eq. 12 shows that the mixture mean is the weighted sum of means derived from individual data sources.

Predictive aleatoric and epistemic uncertainty. Apart from the mean, the predictive uncertainty of the target is of great interest as well, since it allows to compute confidence intervals on the predictions and facilitates the decision making based on volume predictions. Meanwhile, by jointly modeling predictive mean and uncertainty, our mixture ensemble provides well-calibrated prediction, which will be demonstrated in the experiment section.

In a Bayesian setting, there are two main types of uncertainty one can model.

– Aleatoric uncertainty captures underlying noise inherent in the observations. For instance, in financial markets, a widely used aleatoric uncertainty is the volatility of stock return, which reflects the price fluctuation over time. It can be estimated either by empirical variance or GARCH family models.
– Epistemic uncertainty is the uncertainty in the model, which captures what our model does not know due to lack of training data. It can be explained away with increased training data.

In the following, by deriving the conditional variance of the target in the Bayesian mixture manner, we demonstrate that the total variance is decomposed into aleatoric and epistemic uncertainties, which reflect different aspects of the variance of the target y_τ.

$$
\begin{aligned}
&\mathrm{Var}(y_\tau|\{\mathbf{x}_{1,<\tau},\cdots,\mathbf{x}_{S,<\tau}\},\mathcal{D}) \\
&= \int_y y^2 p(y|\{\mathbf{x}_{1,<\tau},\cdots,\mathbf{x}_{S,<\tau}\},\mathcal{D})dy - \mathbb{E}^2[y_\tau|\{\mathbf{x}_{1,<\tau},\cdots,\mathbf{x}_{S,<\tau}\},\mathcal{D}] \\
&= \underbrace{\frac{1}{M}\sum_{m=1}^{M}\sum_{s=1}^{S}\mathbb{P}_{\omega_m}(z_t=s|\cdot)\mathrm{Var}(y|z_t=s,\mathbf{x}_{s,<\tau},\Theta_m)+}_{\text{Aleatoric Uncertainty}} \\
&\underbrace{\frac{1}{M}\sum_{m=1}^{M}\sum_{s=1}^{S}\mathbb{P}_{\omega_m}(z_t=s|\cdot)\mathbb{E}^2[y|z_t=s,\mathbf{x}_{s,\tau},\Theta_m)-\mathbb{E}^2[y|\{\mathbf{x}_{1,\tau},\cdots,\mathbf{x}_{S,\tau}\}_1^\tau,\mathcal{D}],}_{\text{Epistemic Uncertainty}}
\end{aligned}
\tag{13}
$$

where ω_m is from sample Θ_m. Eq. 13 bridges the aleatoric and epistemic uncertainty by the derivation of total variance of y_τ in the Bayesian mixture setting. The decomposition in Eq. 13 also theoretically demonstrates the relation between total variance and the aleatoric and epistemic uncertainty in Bayesian modeling, namely the total variance is composed of inherent noise and model uncertainty on the target.

The aleatoric part in Eq. 13 stems from variance induced from multi-source data. It captures the noise inherent to the target which could depend on $\mathbf{x}_{s,<\tau}$. As a comparison, a classical aleatoric uncertainty (or volatility) estimation model, the GARCH, is typically used to estimate the volatility solely with the target time series. It has no mechanism to capture the evolving relevance of multi-source data to the aleatoric uncertainty of the target.

The epistemic uncertainty on mean in Eq. 13 accounts for uncertainty in the model parameters i.e. uncertainty which captures our ignorance about which model generated our collected data. This uncertainty can be reduced when enough data are available, and is often referred to as model uncertainty.

Model specification. We now specify in detail the mathematical formulation of each component in the temporal mixture model. The inference process we presented so far does not relies on any specific formulation of the model

and thus it is flexible to different specifications. Without loss of generality, we present the following model specification for cryptocurrency data of this paper's interest.

To specify the model, we need to define the predictive density function of individual sources, i.e. $p_\theta(y_t|z_t = s, \mathbf{x}_{s,<t})$ and the probability function of latent variable, i.e. $\mathbb{P}_\omega(z_t = s\,|\mathbf{x}_{1,<t}, \cdots, \mathbf{x}_{S,<t})$. We make a general assumption for both these functions that data from different sources are taken within the same time window w.r.t. the target time step. We denote by h the window length, i.e. the number of past time steps which enter in the conditional probabilities. We assume that this value is the same for each source. Eq. 8 is thus simplified as:

$$\prod_t \sum_{z_t=1}^{S} p_\theta(y_t|z_t = s, \mathbf{x}_{s,(-h,t)}) \cdot \mathbb{P}_\omega(z_t = s|\mathbf{x}_{1,(-h,t)}, \cdots, \mathbf{x}_{S,(-h,t)}), \tag{14}$$

where $\mathbf{x}_{s,(-h,t)}$ represents the data from source s within the time window from $t-h$ to $t-1$ and $\mathbf{x}_{s,(-h,t)} \in \mathbb{R}^{d_s \times h}$.

First, for $p_\theta(y_t|z_t = s, \mathbf{x}_{s,(-h,t)})$, we choose the Gaussian distribution. Since $\mathbf{x}_{s,(-h,t)} \in \mathbb{R}^{d_s \times h}$ is a matrix, we choose bi-linear regression to parameterize the mean and variance of the Gaussian distribution as follows:

$$y_t|z_t = s, \mathbf{x}_{s,(-h,t)}, \qquad \theta \sim \mathcal{N}\big(\mu_s(\mathbf{x}_{s,(-h,t)}), \sigma_s^2(\mathbf{x}_{s,(-h,t)})\big) \tag{15}$$

$$\mu_s(\mathbf{x}_{s,(-h,t)}) = L_{\mu,s}^\top \cdot \mathbf{x}_{s,(-h,t)} \cdot R_{\mu,s} + b_{\mu,s} \tag{16}$$

$$\sigma_s^2(\mathbf{x}_{s,(-h,t)}) = (L_{\sigma,s}^\top \cdot \mathbf{x}_{s,(-h,t)} \cdot R_{\sigma,s} + b_{\sigma,s})^2, \tag{17}$$

where $L_{\mu,s}$, $L_{\sigma,s} \in \mathbb{R}^{d_s}$ and $R_{\mu,s}$, $R_{\sigma,s} \in \mathbb{R}^h$. $b_{\mu,s}$, while $b_{\sigma,s} \in \mathbb{R}$ are bias terms. Note that the above parameters are data source specific and then the parameter set θ can be denoted by $\theta = \{L_{\mu,s}, L_{\sigma,s}, R_{\mu,s}, R_{\sigma,s}, b_{\mu,s}, b_{\sigma,s}\}_{s=1}^S$.

As a results, mean and variance on individual data source are defined as: $\mathbb{E}[y_\tau|\mathbf{x}_{s,<\tau}, \Theta_m] = \mu_s(\mathbf{x}_{s,(-h,t)})$ and $\mathrm{Var}(y_\tau|z_t = s, \mathbf{x}_{s,<\tau}, \Theta_m) = \sigma_s^2(\mathbf{x}_{s,(-h,t)})$. Consequently, the mean and uncertainties in the inference phase can be explicitly calculated.

Second, we choose for $\mathbb{P}_\omega(z_t = s|\mathbf{x}_{1,(-h,t)})$ a multinomial logistic function:

$$\mathbb{P}_\omega(z_t = s|\mathbf{x}_{1,(-h,t)}, \cdots, \mathbf{x}_{S,(-h,t)}) = \frac{\exp(\, f(\mathbf{x}_{s,(-h,t)})\,)}{\exp(\,\sum_{k=1}^S f(\mathbf{x}_{k,(-h,t)})\,)}, \tag{18}$$

$$f(\mathbf{x}) = L^\top \cdot \mathbf{x} \cdot R + b \tag{19}$$

where $L, R \in \mathbb{R}^{d_s}$ and $b \in \mathbb{R}$ is a bias term

3.5 Learning

In this part, we present how to generate samples of $p(\Theta|\mathcal{D})$ in an empirical ensemble manner.

Our approach is based on the ensemble of Maximum a-posteriori (MAP) optimization, which maximizes the (log) posterior of model parameters, i.e. $\log p(\Theta|\mathcal{D})$. Since $\log p(\Theta|\mathcal{D}) \propto \log p_{\Theta}(y_1, \cdots, y_T|\{\mathbf{x}_{1,t}, \cdots, \mathbf{x}_{S,t}\}_{t=0}^{T}) + \log p(\Theta)$, the MAP of $\log p(\Theta|\mathcal{D})$ can be expressed as the following minimization:

$$\min_{\Theta=\{\theta,\omega\}} \mathcal{L}(\Theta;\mathcal{D}) =$$
$$\min_{\Theta=\{\theta,\omega\}} -\sum_{t=1}^{T} \log \sum_{z_t=1}^{S} p_\theta(y_t|z_t = s, \mathbf{x}_{s,(-h,t)}) \cdot \mathbb{P}_\omega(z_t = s|\{\mathbf{x}_{s,(-h,t)}\}_1^S) - \log p(\Theta), \tag{20}$$

where the prior $p(\Theta)$ is viewed as a regularizer in optimization and typically L2 regularization is used.

Nowadays, stochastic gradient descent (SGD) is popular and widely used for this type of large scale optimization. Starting from a random initialized model parameters, in each iteration SGD samples a batch of training instances to update the model parameters as follows:

$$\Theta_i = \Theta_{i-1} - \eta \nabla \mathcal{L}(\Theta_{i-1}; \mathcal{D}_i), \tag{21}$$

where η is the learning rate, a tunable hyperparameter to control the magnitude of gradient update. $\nabla \mathcal{L}(\Theta_{i-1}; \mathcal{D}_i)$ is the gradient of the loss function w.r.t. model parameters given data batch $\mathcal{D}_i$ at iteration i. A consecutive set of batches passing all training instances once is defined as one epoch of training.

Meanwhile, as the availability of highly optimized matrix optimizations common to state-of-the-art machine learning libraries [41, 42], a variety of SGD based optimization methods are developed to improve the stability and convergence rate, for example, widely used Adam, Adagrad and so on [43, 44]. For more details about the SGD sampling procedure, see Appendix C.

4 Experiments

4.1 Data and metrics

In this section, we present the set of different metrics used in our experiments. In all the experiments, data instances are time ordered and we use the first 70% of points for training, the next 10% for validation, and the last 20% of points for out-of-sample testing. All the metrics are evaluated out of sample.

We use two groups of metrics to study the performance.

Error metric: three metrics are used to evaluate the errors between ground-truth volume y_i and predictive mean of volume $\hat{y}_i$ as follows.

The root mean square error RMSE $= \sqrt{\frac{1}{M}\sum_{i=1}^{M}(y_i - \hat{y}_i)^2}$ and mean absolute error MAE $= \frac{1}{M}\sum_{i=1}^{M}|y_i - \hat{y}_i|$ on the out-of-sample testing period ($M = 6,800$). The Pearson correlation coefficient between the predicted volume and true volume is:

$$\text{CORR} = \frac{\sum_{i=1}^{M}(y_i - E[y_i])(\hat{y}_i - E[\hat{y}_i])}{\sqrt{\sum_{i=1}^{M}(y_t - E[y_i])^2}\sqrt{\sum_{i=1}^{M}(\hat{y}_i - E[\hat{y}_i])^2}} \tag{22}$$

and clearly larger values indicate better models.

Calibration metric: this type of metric evaluates how well the distribution characterized by predictive mean $\hat{y}_i$ and variance $\hat{\sigma}_i^2$ fits the ground-truth target values.

For GARCH-family the predictive Normalized Negative Log-Likelihood score is calculated as [45]

$$\text{NNLL} = -\frac{1}{M}\sum_{i=1}^{M}\ln\mathcal{N}(y_i|\hat{y}_i, \hat{\sigma}_i), \tag{23}$$

where $\mathcal{N}(.|\hat{y}_i, \hat{\sigma}_i^2)$ denotes the Gaussian density function parameterized by predictive mean $\hat{y}_i$ and variance $\hat{\sigma}_i^2$. The normalization is done with the total number of out-of-sample points (M).

For the temporal mixture model, the NNLL score is calculated by using the likelihood function Eq. 10, normalized by the total number of out-of-sample points. Lower values indicate better performance.

The $2\hat{\sigma}$ coverage (IC) counts the fraction of true values y_i that fall within $2\hat{\sigma}_i$ range, where $\hat{\sigma}_i$ is the predictive standard deviation:

$$\text{IC} = \frac{1}{M}\sum_{i=1}^{M}\mathbf{1}_{[\hat{y}_i - 2\hat{\sigma}_i \le y_i \le \hat{y}_i + 2\hat{\sigma}_i]}. \tag{24}$$

Under the Gaussian distribution, the $2\hat{\sigma}_i$ interval around the mean value corresponds to $\text{IC}^* = 0.9545$, which means the percent of ground-truth values falling within $2\hat{\sigma}_i$ is ideally to be 95.45% for well-calibrated models [46].

The closer the empirical value to IC^*, the better calibrated the model is. Therefore, we define a simpler measure, the $2\hat{\sigma}$ coverage error (ICE), which measures the absolute difference to the ground value IC^* as:

$$\text{ICE} = |\text{IC} - 0.9545|. \tag{25}$$

Finally, for a prediction interval $[y_i^-, y_i^+]$, we calculate the mean prediction interval width (IW) as:

$$\text{IW} = \frac{1}{C}\sum_{i=1}^{M}(y_i^+ - y_i^-) = \frac{1}{C}\sum_{i=1}^{M}4\hat{\sigma}_i, \tag{26}$$

where $y_i^+ = \hat{y}_i + 2\hat{\sigma}_i$ and $y_i^- = \hat{y}_i - 2\hat{\sigma}_i$ and C is the number of true target values falling within the prediction interval $C = \sum_{i=1}^{M}\mathbf{1}_{[\hat{y}_i - 2\hat{\sigma}_i \le y_i \le \hat{y}_i + 2\hat{\sigma}_i]}$. The

IW measure should be minimized and it tells that the high-quality prediction intervals should be as narrow as possible, while capturing a specified portion of data, without assumptions on the distribution [46].

4.2 Results

In the first set of experiments we concentrate on 5-min ahead predictions for both markets. For all the processes in the AR-GARCH family the autoregressive order was fixed to $p = 10$, which is the maximum lag for which the partial autocorrelation function is still significant.

Table 1 and Table 2 show the out of sample prediction metrics for the two markets. Notice that the 5 min mean volume on test set is 49.67 for Bitfinex and 21.47 for Bitstamp. By comparing these numbers with the MAE, we observe that the level of noise is quite high, since the two values are comparable. This can be also seen by computing the average relative forecasting error. For AR-GARCH model this value is $\mathbb{E}[(y_t - \hat{y}_t)/y_t] = 3.356$, $\mathbb{E}[(y_t - \hat{y}_t)/y_t] = 3.9857$ for Bitfinex and Bitstamp, respectively. However the Pearson correlation between AR-GARCH predictions and true volume for the two markets are $\rho_1(y_t, \hat{y}_t) = 0.5268$, $\rho_2(y_t, \hat{y}_t) = 0.4799$, which are pretty high values. Furthermore, we observe that external features are not helping the ARX-GARCH family to get better scores w.r.t to a simpler AR-GARCH family.

When comparing different models, we observe that the temporal mixture ensemble model has significantly lower MAE, even with respect to the machine learning benchmark (gradient boosting), while the RMSE is comparable or slightly lower. We provide an explanation for this result later when we condition the forecast on the volume quartile. Finally, we note that the mixture ensemble has drastically better volume uncertainty predictions (as measured by NNLL, IC, and IW) than the other models. Remind that gradient boosting does not provide estimates on the uncertainty of the forecast.

Table 1: 5-min ahead prediction metrics for Bitfinex markets. Out of sample performance of 5-min ahead predictions for the period of June 2018 - November 2018 (70% train, 10% validation and 20% test). The arrow symbols in the first line indicate the direction of the metrics for better models.

MODEL	RMSE ↓	MAE ↓	NNLL ↓	CORR ↑	ICE ↓	IW ↓
AR-GARCH	63.952	38.229	5.4866	0.5268	0.012	284.598
AR-GJR-GARCH	63.78	38.838	5.4799	0.5306	0.012	282.843
ARX-GARCH	63.677	38.086	5.482	0.531	0.011	282.934
ARX-GJR-GARCH	63.597	37.208	5.4788	0.4225	0.010	285.323
Gradient boosting	62.529	37.768	NA	0.562	NA	NA
Mixture ensemble	63.68	33.72	4.82	0.55	0.002	184.54

Table 2: 5-min ahead prediction metrics for Bitstamp markets. Out of sample performance of 5-min ahead predictions for the period of June 2018 - November 2018 (70% train, 10% validation and 20% test). The arrow symbols in the first line indicate the direction of the metrics for better models.

MODEL	RMSE ↓	MAE ↓	NNLL ↓	CORR ↑	ICE ↓	IW ↓
AR-GARCH	38.118	17.089	4.9026	0.4799	0.022	158.496
AR-GJR-GARCH	38.051	17.328	4.8952	0.4819	0.022	162.632
ARX-GARCH	39.13	20.356	4.931	0.4534	0.023	157.959
ARX-GJR-GARCH	39.131	20.324	4.9251	0.453	0.023	158.16
Gradient boosting	37.764	15.966	NA	0.5177	NA	NA
Mixture ensemble	38.90	15.54	3.89	0.51	0.005	91.69

We now focus on the forecast of the temporal mixture ensemble In Fig 2 and Fig 3 we show the volume and uncertainty prediction on a 5 min level for the two markets in a random interval of two hundred 5 min intervals. In both markets the 95% confidence interval covers quite well the actual values. We notice in both plots the presence of large spikes, corresponding to 5 min intervals where, unexpectedly, a large volume is traded and clearly the model is unable to forecast them. We believe that these volume bursts are responsible of the large difference between MAE and RMSE and of the fact that all models have comparable RMSE (but different MAE). Below we provide more evidence of this.

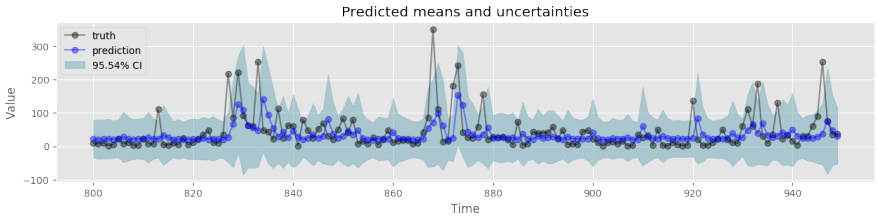

Fig. 2: Sample time series of 5 min trading volumes in Bitfinex (black line). The blue line is the 5 min ahead prediction with the temporal mixture ensemble and the light blue area represent its 95% confidence interval.

The temporal mixture ensemble is able to quantify at each time step the contribution of each source to the target forecasting. In Fig 4 we show the dynamical contributions of the $S = 4$ sources for a random sample of 5-min ahead predictions for Bitfinex market. We notice that the relative contributions varies with time and we observe that the external order book source from the less liquid market does not contribute much to predictions. On the contrary, in Fig. 5, where the data for Bitstamp are shown, external order book and external transaction features from the more liquid market (Bitfinex) play a more dominant role.

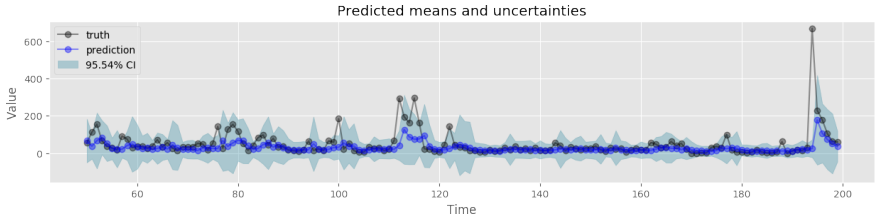

Fig. 3: Sample time series of 5 min trading volumes in Bitstamp (black line). The blue line is the 5 min ahead prediction with the temporal mixture ensemble and the light blue area represent its 95% confidence interval.

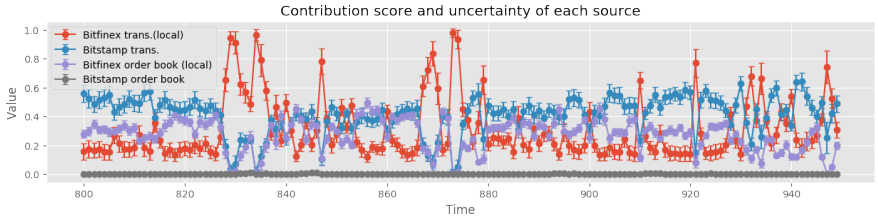

Fig. 4: Data source contribution for a time series sample of 5 min trading volume in Bitfinex. The contributions are obtained with the temporal mixture ensemble.

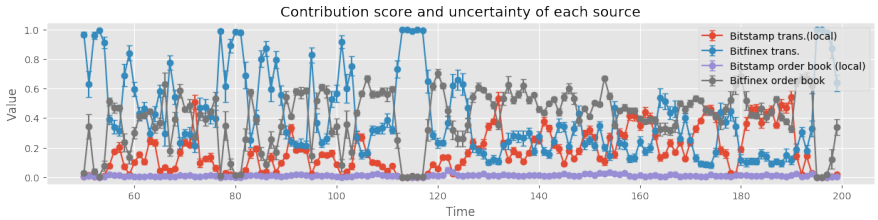

Fig. 5: Data source contribution for a time series sample of 5 min trading volume in Bitstamp. The contributions are obtained with the temporal mixture ensemble.

In order to understand in a more quantitative way the role of large volumes in the forecasting ability of the different models, we compute the RMSE and MAE conditional to the quartile of the true value of the volume of the target market. Table 3 reports the results. First of all, we notice that, for all the methods, both error measures change by almost an order of magnitude when moving from the lowest to the largest quartile. This is a strong indication that the main problems in forecasting derive from large and unexpected volume bursts. We finally notice that the temporal mixture ensemble outperforms the other models, both considering RMSE and MAE, when Bitfinex volume is in

Q1-Q3. For Bitstamp market the results are less clear, but in general machine learning methods work better than the benchmark econometric models.

Table 3: 5-min ahead prediction metrics for both markets conditional to the quartile of the target volume in the period of June 2018 - November 2018.

BITFINEX MARKET	RMSE Q1	RMSE Q2	RMSE Q3	RMSE Q4
AR-GARCH	25.4712	32.5289	41.3908	116.2046
ARX-GARCH	29.0931	31.2223	34.6301	114.9813
Gradient Boosting	30.9203	29.4742	28.4582	112.6362
Mixture ensemble	22.82	21.06	25.74	120.93
	MAE Q1	MAE Q2	MAE Q3	MAE Q4
AR-GARCH	18.7562	21.1582	27.2724	80.4797
ARX-GARCH	24.9695	23.1059	22.5028	77.9657
Gradient Boosting	28.0851	24.4661	20.2829	74.4320
Mixture ensemble	20.90	15.14	16.94	84.49
BITSTAMP MARKET	RMSE Q1	RMSE Q2	RMSE Q3	RMSE Q4
AR-GARCH	11.5399	14.5152	17.962	73.2483
ARX-GARCH	18.9325	19.0446	18.1409	71.2284
Gradient Boosting	11.4270	11.0809	10.5455	72.6233
Mixture ensemble	11.73	11.34	11.10	74.87
	MAE Q1	MAE Q2	MAE Q3	MAE Q4
AR-GARCH	7.6147	8.7595	10.5267	40.2782
ARX-GARCH	17.3215	15.8453	12.4302	35.6983
Gradient Boosting	10.0851	8.7940	7.1503	37.4343
Mixture ensemble	11.15	8.55	5.77	39.13

We have also repeated these experiments for the data with 1 min resolution. The results are collected in the figures and tables in the Appendix. Since the volume distribution at small time scale is more leptokurtic than the one at 5 min, large volume bursts are more frequent and tend to deteriorate significantly the forecasting performance of all the models. This can be understood by considering that the average relative error of the AR-GARCH model is 447 and 119 for the two markets, to be compared with the values 3.35 and 3.99 observed at 5 min resolution. Looking at Table 6, where the analysis conditional to quartile is presented, it is again clear that machine learning methods outperforms the econometric benchmarks (except in the fourth quartile, as expected). Finally, the temporal mixture ensemble provides confidence intervals which are significantly more accurate than those obtained with the other models.

5 Conclusion and discussion

In this paper, we analyzed the problem of predicting trading volume and its uncertainty in cryptocurrency exchange markets. The main innovations proposed in this paper are (i) the use of transaction and order book data from

different markets and (ii) the use of a class of model able to identify at each time step the set of data locally more useful in predictions.

By investigating data from BTC/USD exchange markets, we found that time series models of the AR-GARCH family do provide fair basic predictions for volume and its uncertainty, but when external data (e.g. from order book and/or from other markets) are added, the prediction performance does not improve significantly. Our analysis suggests that this might be due to the fact that the contribution of this data to the prediction could be not constant over time, but depending on the "market state". The temporal mixture ensemble model is designed precisely to account for such a variability. Indeed we find that this method outperforms time series models both in point and in interval predictions of trading volume. Moreover, especially when compared to other machine learning methods, the temporal mixture approach is significantly more interpretable, allowing the inference of the dynamical contributions from different data sources as a core part of the learning procedure. This has important potential implications for decision making in economics and finance.

One of the critical outcomes of the forecasting exercise is that the predictability significantly depends on the size of the volume to be forecast. We found that our method works better than the benchmarks when volume is not in the top quartile, while in this extreme case all the methods perform poorly. This is likely due to the presence of unexpected bursts of volume which are very challenging to forecast. As a consequence, the prediction is significantly less accurate when the time interval of the series is too short, since in this case extreme fluctuations are more frequent.

Finally, although the method has been proposed and tested for cryptocurrency volume in two specific exchanges, we argue that it can be successfully applied (in future work) to other cryptocurrencies and to more traditional financial assets.

Acknowledgement

This work has been funded by the European Program scheme 'INFRAIA-01-2018-2019: Research and Innovation action', grant agreement #871042 'SoBigData++: European Integrated Infrastructure for Social Mining and Big Data Analytics'.

References

1. S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008. [Online]. Available: http://bitcoin.org/bitcoin.pdf
2. E. Baumhl, "Are cryptocurrencies connected to forex? a quantile cross-spectral approach," *Finance Research Letters*, vol. 29, pp. 363 – 372, 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1544612318303611

3. A. P. Chaboud, B. Chiquoine, H. E., and C. Vega, "Rise of the machines: Algorithmic trading in the foreign exchange market," *The Journal of Finance*, vol. 69, no. 5, pp. 2045–2084, 2014. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/jofi.12186
4. T. Hendershott, C. Jones, and A. Menkveld, "Does algorithmic trading improve liquidity?" *The Journal of Finance*, vol. 66, no. 1, pp. 1–33, 2011. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1540-6261.2010.01624.x
5. C. Frei and N. Westray, "Optimal execution of a vwap order: A stochastic control approach," *Mathematical Finance*, vol. 25, no. 3, pp. 612–639, 2015. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/mafi.12048
6. A. Barzykin and F. Lillo, "Optimal vwap execution under transient price impact," *Available at https://arxiv.org/abs/1901.02327*, 2018.
7. C. T. Brownlees, F. Cipollini, and G. M. Gallo, "Intra-daily volume modeling and prediction for algorithmic trading," *Journal of Financial Econometrics*, vol. 9, no. 3, pp. 489–518, 2010.
8. V. Satish, A. Saxena, and M. Palmer, "Predicting intraday trading volumeand volume percentages," *The journal of trading*, vol. 9, no. 3, pp. 15–25, 2014.
9. R. Chen, Y. Feng, and D. Palomar, "Forecasting intraday trading volume: A kalman filter approach," *Available at SSRN 3101695*, 2016.
10. T. G. Andersen, "Return volatility and trading volume: An information flow interpretation of stochastic volatility," *The Journal of Finance*, vol. 51, no. 1, pp. 169–204, 1996.
11. J. Chu, S. Nadarajah, and S. Chan, "Statistical analysis of the exchange rate of bitcoin," *PLOS ONE*, vol. 10, no. 7, pp. 1–27, 2015.
12. A. Urquhart, "The inefficiency of bitcoin," *Economics Letters*, vol. 148, pp. 80–82, 2016.
13. P. Katsiampa, "Volatility estimation for bitcoin: A comparison of GARCH models," *Economics Letters*, vol. 158, pp. 3–6, 2017.
14. M. Balcilar, E. Bouri, R. Gupta, and D. Roubaud, "Can volume predict bitcoin returns and volatility? a quantiles-based approach," *Economic Modelling*, vol. 64, pp. 74–81, 2017.
15. M. Hougan, H. Kim, M. Lerner, and B. A. Management, "Economic and non-economic trading in bitcoin: Exploring the real spot market for the worlds first digital commodity," *Bitwise Asset Management*, 2019.
16. M. D. Gould, M. A. Porter, S. Williams, M. McDonald, D. J. Fenn, and S. D. Howison, "Limit order books," *Quantitative Finance*, vol. 13, no. 11, pp. 1709–1742, 2013.
17. M. Rambaldi, E. Bacry, and F. Lillo, "The role of volume in order book dynamics: a multivariate hawkes process analysis," *Quantitative Finance*, vol. 17, no. 7, pp. 999–1020, 2016.
18. T. G. Andersen and T. Bollerslev, "Intraday periodicity and volatility persistence in financial markets," *Journal of empirical finance*, vol. 4, no. 2-3, pp. 115–158, 1997.
19. R. Engle, "New frontiers for arch models," *Journal of Applied Econometrics*, vol. 17, no. 5, pp. 425–446, 2002.
20. R. F. Engle and M. E. Sokalska, "Forecasting intraday volatility in the us equity market. multiplicative component garch," *Journal of Financial Econometrics*, vol. 10, no. 1, pp. 54–83, 2012.
21. T. Bollerslev and E. Ghysels, "Periodic autoregressive conditional heteroscedasticity," *Journal of Business & Economic Statistics*, vol. 14, no. 2, pp. 139–151, 1996.
22. T. Bollerslev, "Generalized autoregressive conditional heteroskedasticity," *Journal of econometrics*, vol. 31, no. 3, pp. 307–327, 1986.
23. L. R. Glosten, R. Jagannathan, and D. E. Runkle, "On the relation between the expected value and the volatility of the nominal excess return on stocks," *The journal of finance*, vol. 48, no. 5, pp. 1779–1801, 1993.
24. J.-M. Zakoian, "Threshold heteroskedastic models," *Journal of Economic Dynamics and control*, vol. 18, no. 5, pp. 931–955, 1994.
25. J. H. Friedman, "Greedy function approximation: a gradient boosting machine," *Annals of statistics*, pp. 1189–1232, 2001.
26. D. Nielsen, "Tree boosting with xgboost-why does xgboost win" every" machine learning competition?" Master's thesis, NTNU, 2016.

27. S. B. Taieb and R. J. Hyndman, "A gradient boosting approach to the kaggle load forecasting competition," *International journal of forecasting*, vol. 30, no. 2, pp. 382–394, 2014.
28. N. Zhou, W. Cheng, Y. Qin, and Z. Yin, "Evolution of high-frequency systematic trading: a performance-driven gradient boosting model," *Quantitative Finance*, vol. 15, no. 8, pp. 1387–1403, 2015.
29. X. Sun, M. Liu, and Z. Sima, "A novel cryptocurrency price trend forecasting model based on lightgbm," *Finance Research Letters*, 2018.
30. S. R. Waterhouse, D. MacKay, and A. J. Robinson, "Bayesian methods for mixtures of experts," in *Advances in neural information processing systems*, 1996, pp. 351–357.
31. S. E. Yuksel, J. N. Wilson, and P. D. Gader, "Twenty years of mixture of experts," *IEEE transactions on neural networks and learning systems*, vol. 23, pp. 1177–1193, 2012.
32. X. Wei, J. Sun, and X. Wang, "Dynamic mixture models for multiple time-series." in *IJCAI*, vol. 7, 2007, pp. 2909–2914.
33. L. Bazzani, H. Larochelle, and L. Torresani, "Recurrent mixture density network for spatiotemporal visual attention," *arXiv preprint arXiv:1603.08199*, 2016.
34. T. Guo, T. Lin, and N. Antulov-Fantulin, "Exploring interpretable lstm neural networks over multi-variable data," in *International Conference on Machine Learning*, 2019, pp. 2494–2504.
35. T. Guo, A. Bifet, and N. Antulov-Fantulin, "Bitcoin volatility forecasting with a glimpse into buy and sell orders," in *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2018, pp. 989–994.
36. P. Schwab, D. Miladinovic, and W. Karlen, "Granger-causal attentive mixtures of experts: Learning important features with neural networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 4846–4853.
37. R. Kurle, S. Günnemann, and P. van der Smagt, "Multi-source neural variational inference," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 4114–4121.
38. B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," in *Advances in neural information processing systems*, 2017, pp. 6402–6413.
39. W. J. Maddox, P. Izmailov, T. Garipov, D. P. Vetrov, and A. G. Wilson, "A simple baseline for bayesian uncertainty in deep learning," in *Advances in Neural Information Processing Systems*, 2019, pp. 13 132–13 143.
40. J. Snoek, Y. Ovadia, E. Fertig, B. Lakshminarayanan, S. Nowozin, D. Sculley, J. Dillon, J. Ren, and Z. Nado, "Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift," in *Advances in Neural Information Processing Systems*, 2019, pp. 13 969–13 980.
41. M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, "Tensorflow: A system for large-scale machine learning," in *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, 2016, pp. 265–283.
42. A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," 2017.
43. S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747*, 2016.
44. D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *International Conference on Learning Representations*, 2015.
45. Y. Wu, J. M. Hernández-Lobato, and Z. Ghahramani, "Gaussian process volatility model," in *NIPS*, 2014, pp. 1044–1052.
46. T. Pearce, A. Brintrup, M. Zaki, and A. Neely, "High-quality prediction intervals for deep learning: A distribution-free, ensembled approach," in *International Conference on Machine Learning*, 2018, pp. 4075–4084.
47. F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

48. S. Mandt, M. D. Hoffman, and D. M. Blei, "Stochastic gradient descent as approximate bayesian inference," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 4873–4907, 2017.
49. G. Gur-Ari, D. A. Roberts, and E. Dyer, "Gradient descent happens in a tiny subspace," *arXiv preprint arXiv:1812.04754*, 2018.

6 Appendix A: Results on 1-min intervals

In this Appendix we report, for the sake of completeness, the results obtained with 1 min data. As we have mentioned in the main text, the high burstiness of the volume data on this time scale lead to a significant deterioration of the forecasting ability of all the models.

First of all, we note that for AR-GARCH model (and similarly for the other models) the relative mean absolute error for the two markets are huge being equal to $AVG[(y_t - \hat{y_t})/y_t] = 447.751$, $AVG[(y_t - \hat{y_t})/y_t] = 119.168$. Table 4 and 5 reports the metrics of all the models' predictions for Bitfinex and Bitstamp, respectively. We observe that different variants of AR-GARCH and ARX-GARCH behave similarly w.r.t. RMSE, MAE and NNLL metric. Furthermore, gradient boosting has the lowest RMSE and MAE errors for volume predictions. We observe that the temporal mixture ensemble model is having comparable volume predictions but drastically better volume volatility predictions (NNLL, ICE, IW).

In Fig. 6 and Fig. 7 we show the 1 min-ahead volume prediction along with uncertainty for Bitfinex and Bitstamp, respectively. At the same time, we plot the contributions from different sources (see Fig. 8 and 9), where we observe the dynamic contribution of different sources. In particular, for 1-min ahead prediction in Bitfinex (Fig 8) we see the interplay of local transaction features and external features and the smallest contribution of local order book features. In Fig. 9 for less liquid Bitstamp, we observe that the external order book features are playing a dominant dynamic role together with local transaction features.

Finally, Table 6 shows the RMSE and MAE for the various models conditioning on the quartile of the target variable. Again it is clear that machine learning models works better for volume in Q1-Q3, while for large volumes in Q4 the performances of the models become similar.

Table 4: 1-min ahead prediction metrics for Bitfinex markets. Out of sample performance of 1-min ahead predictions for the period of June 2018 - November 2018 (70% train, 10% validation and 20% test). The arrow symbols in the first line indicate the direction of the metrics for better models.

MODEL	RMSE ↓	MAE ↓	NNLL ↓	CORR ↑	ICE ↓	IW ↓
AR-GARCH	20.113	9.936	4.2542	0.447	0.016	86.523
AR-GJR-GARCH	20.086	9.976	4.2522	0.4471	0.016	86.366
ARX-GARCH	20.132	9.703	4.2445	0.4451	0.015	85.423
ARX-GJR-GARCH	20.103	9.867	4.2494	0.4453	0.016	86.603
Gradient boosting	19.985	8.87	NA	0.483	NA	NA
Mixture ensemble	20.00	9.42	2.90	0.46	0.001	61.89

Table 5: 1-min ahead prediction metrics for Bitstamp markets. Out of sample performance of 1-min ahead predictions for the period of June 2018 - November 2018 (70% train, 10% validation and 20% test). The arrow symbols in the first line indicate the direction of the metrics for better models.

MODEL	RMSE ↓	MAE ↓	NNLL ↓	CORR ↑	ICE ↓	IW ↓
AR-GARCH	11.196	4.248	3.5866	0.4774	0.023	40.115
AR-GJR-GARCH	11.193	4.243	3.5841	0.4762	0.023	42.34
ARX-GARCH	11.252	4.504	3.5895	0.4701	0.023	39.794
ARX-GJR-GARCH	11.21	4.188	3.5862	0.4739	0.023	41.598
Gradient boosting	11.182	3.73	NA	0.5317	NA	NA
Mixture ensemble	11.38	4.058	2.03	0.49	0.004	27.67

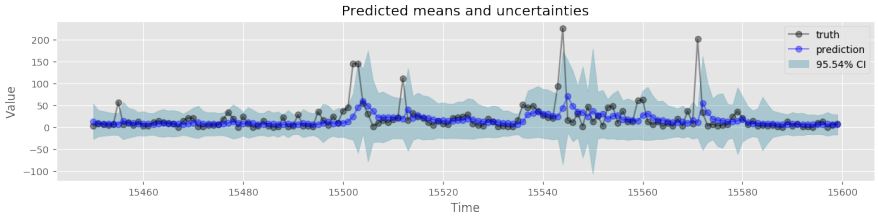

Fig. 6: Sample time series of 1 min trading volumes in Bitfinex (black line). The blue line is the 1 min ahead prediction with the temporal mixture ensemble model and the light blue area represents its 95% confidence interval.

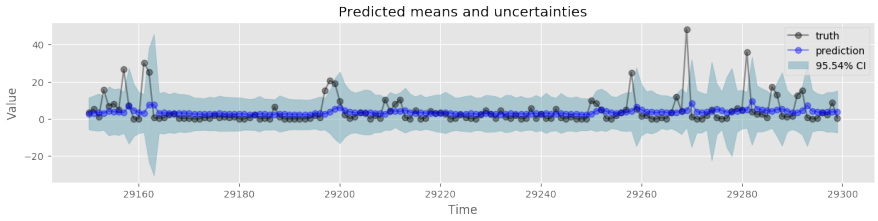

Fig. 7: Sample time series of 1 min trading volumes in Bitstamp (black line). The blue line is the 1 min ahead prediction with the temporal mixture ensemble model and the light blue area represents its 95% confidence interval.

7 Appendix B: Gradient Boosting Machine Learning

For a given training sample $\{y_t, \mathbf{x}_t\}_{t=1}^{T}$, our goal is to find a function $F^*(\mathbf{x})$ such that the expected value of loss function $\Psi(y, F(\mathbf{x}))$ is minimized over the joint distribution of $\{y, \mathbf{x}\}$

$$F^*(\mathbf{x}) = \underset{F(x)}{\arg\min}\, \mathbb{E}_{y,\mathbf{x}} \Psi(y, F(\mathbf{x})). \tag{27}$$

Under the additive expansion $F(\mathbf{x}) = \sum_{m=0}^{M} \beta_m h(\mathbf{x}; \mathbf{a}_m)$ with parameterized functions $h(\mathbf{x}; \mathbf{a}_m)$, we proceed with the minimization of data estimate of ex-

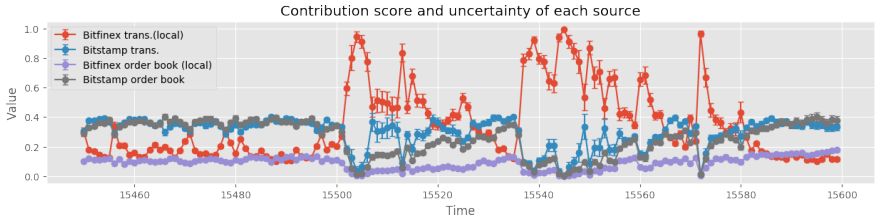

Fig. 8: Data source contribution for a time series sample of 1 min trading volume in Bitfinex. The contributions are obtained with the temporal mixture ensemble model.

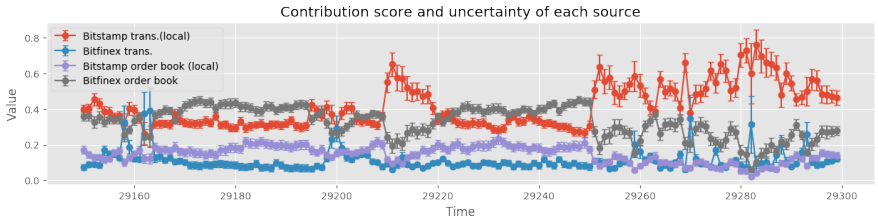

Fig. 9: Data source contribution for a time series sample of 5 min trading volume in Bitstamp. The contributions are obtained with the temporal mixture ensemble model.

Table 6: 1-min ahead prediction metrics for both markets conditional to the quartile of the target volume in the period of June 2018 - November 2018.

BITFINEX MARKET	RMSE Q1	RMSE Q2	RMSE Q3	RMSE Q4
AR-GARCH	7.3131	9.163	10.3165	37.0707
ARX-GARCH	7.0584	9.0724	10.1451	37.2288
Gradient Boosting	5.2867	5.9321	6.0074	38.7101
Mixture ensemble	6.59	7.56	8.086	37.86
	MAE Q1	MAE Q2	MAE Q3	MAE Q4
AR-GARCH	5.8234	6.3874	5.9997	21.5342
ARX-GARCH	5.2314	5.9161	5.8702	21.7944
Gradient Boosting	4.6802	4.78	3.8955	22.1228
Mixture ensemble	6.07	6.16	4.84	21.70
BITSTAMP MARKET	RMSE Q1	RMSE Q2	RMSE Q3	RMSE Q4
AR-GARCH	3.3807	3.9941	4.376	21.3262
ARX-GARCH	3.7292	4.3312	4.6574	21.2603
Gradient Boosting	2.0817	2.2466	2.1172	22.0518
Mixture ensemble	2.71	2.89	2.64	22.19
	MAE Q1	MAE Q2	MAE Q3	MAE Q4
AR-GARCH	2.4622	2.5202	2.2675	9.7414
ARX-GARCH	2.8769	2.9354	2.6175	9.5851
Gradient Boosting	1.7587	1.7614	1.3628	10.0345
Mixture ensemble	2.46	2.40	1.55	9.90

pected loss [25]:

$$\{\beta_m, \mathbf{a}_m\}_1^M = \arg\min_{\beta'_m, \mathbf{a}'_m} \sum_{t=1}^{T} \Psi(y_t, \sum_{m=0}^{M} \beta'_m h(\mathbf{x}_t; \mathbf{a}'_m)). \tag{28}$$

However, for practical purposes first we make the initial guess $F_0(\mathbf{x}) = \arg\min_c \sum_{t=1}^{T} \Psi(y_t, c)$ and then parameters are jointly fit in a forward incremental way $m = 1, ..., M$:

$$(\beta_m, \mathbf{a}_m) = \arg\min_{\beta, \mathbf{a}} \sum_{t=1}^{T} \Psi(y_t, F_{m-1}(\mathbf{x}_t) + \beta h(\mathbf{x}_t; \mathbf{a})) \tag{29}$$

and

$$F_m(\mathbf{x}_t) = F_{m-1}(\mathbf{x}_t) + \beta_m h(\mathbf{x}_t; \mathbf{a}_m). \tag{30}$$

First, the function $h(\mathbf{x}_t; \mathbf{a})$ is fit by least-squares to the pseudo-residuals $\widetilde{y}_{t,m}$

$$\mathbf{a}_m = \arg\min_{\mathbf{a}, \rho} \sum_{t=1}^{T} [\widetilde{y}_{t,m} - \rho h(\mathbf{x}_t; \mathbf{a})]^2, \tag{31}$$

which for squared loss $\Psi(y_t, F(\mathbf{x}_t)) = \frac{1}{2}(y_t - F(x))^2$ at stage m is a residual $\widetilde{y}_{t,m} = (y_t - F_{m-1}(\mathbf{x}_t))$. For general loss Ψ, we have

$$\widetilde{y}_{t,m} = -\left[\frac{\partial \Psi(y_t, F(\mathbf{x}_t))}{\partial F(\mathbf{x}_t)}\right]_{F(\mathbf{x})=F_{m-1}(\mathbf{x})}. \tag{32}$$

Now, we just find the coefficient β_m for the expansion as

$$\beta_m = \arg\min_{\beta} \sum_{t=1}^{T} \psi(y_t, F_{m-1} + \beta h(\mathbf{x}_t; \mathbf{a}_m)). \tag{33}$$

Each base learner $h(\mathbf{x}_t; \mathbf{a}_m)$ partitions the feature space $\mathbf{x}_t \in \mathbf{X}$ into L-disjoint regions $\{R_{l,m}\}_1^L$ and predicts a separate constant value in each:

$$h(\mathbf{x}_t; \{R_{l,m}\}_1^L) = \sum_{l=1}^{L} \bar{y}_{l,m} \mathbf{1}(\mathbf{x}_t \in R_{l,m}), \tag{34}$$

where $\bar{y}_{l,m}$ is the mean value of pseudo-residual (eq. 32) in each region $R_{l,m}$

$$\bar{y}_{l,m} = \frac{\sum_{t=1}^{T} \widetilde{y}_{t,m} \mathbf{1}[\mathbf{x}_t \in R_{l,m}]}{\sum_{t=1}^{T} \mathbf{1}[\mathbf{x}_t \in R_{l,m}]}. \tag{35}$$

We have used the GBM implementation from Scikit-learn library [47] for all our experiments [8].

[8] Within this library, for hyper-parameters optimization, we take the following regression tree hyper-parameters into the account: "n_estimators", "max_features", "min_samples_leaf", "max_depth" and the following learning hyper-parameters: "learning_rate" and "loss".

8 Appendix C: SGD-based sampling

In stochastic gradient descent (SGD) based optimization, stochasticity comes from two places:

- SGD trajectory. The iterates $\{\Theta_0, \cdots, \Theta_i\}$ forms a exploratory trajectory of posterior space $\log p(\Theta|\mathcal{D})$, as Θ_i is updated by randomly data sample $\mathcal{D}_i$. Recent works [48, 49] studied the connection of trajectory iterates to an approximate Markov chain Monte Carlo sampler by analyzing the dynamics of SGD.
- Model initialization. Different initialization of model parameters, i.e. Θ_0, leads to distinct trajectories. It has been shown that ensembles of independently initialized and trained models empirically often provide comparable performance in prediction and uncertainty quantification w.r.t. sampling and variational inference based methods, even though it does not apply conventional Bayesian grounding [38, 40].

In this paper, we make a hybrid approach, that uses both sources of stochasticity to obtain approximate samples $\{\Theta_m\} \sim p(\Theta|\mathcal{D})$ as follows:

$$\{\Theta_m\} \approx \bigcup_j \{\Theta_i^j, \cdots, \Theta_I^j\} \tag{36}$$

Eq. 36 indicates that from each independently trained SGD trajectory (indexed by j), we skip the beginning few epochs as a "burn-in" step (common in Monte Carlo methods). We choose the remaining as samples from this trajectory. Then, we further take the union of samples from independent trajectories as the samples used by the inference in Sec. 3.4.

In our experiments, we use Adam optimization, a variant of SGD, which has been widely used in machine learning [44]. We found that 5 to 10 independent training processes can give rise to decently accurate and calibrated forecasting. Moreover, by parallel computing on GPU, we perform each training process in parallel without loss of efficiency.