Temporal Mixture Ensemble for Probabilistic Forecasting of Intraday Volume in Cryptocurrency Exchange Markets[*]

Benson Lee **Aliaksandr Samushchyk** **Peter Furjesz**
ETH Zurich
{benlee, asamushchyk, pfuerjesz}@student.ethz.ch

Abstract

We replicate and extend a Temporal Mixture Ensemble (TME) model for probabilistic forecasting of intraday trading volume in cryptocurrency markets. The model captures temporal dynamics and learns the importance of different data sources while preserving feature interpretability. We evaluate it using high-frequency transaction and order book data from the Bitstamp exchange.

1 Introduction

In recent years, cryptocurrencies have emerged as a focal point of interest across various academic disciplines, including finance, economics, computer science, and complex systems. At the core of this innovation is a decentralized peer-to-peer transaction network secured by cryptographic protocols such as elliptic curve cryptography and SHA-256 hashing. Transactions are verified by network participants and recorded on a publicly accessible distributed ledger, known as the blockchain. The process of verification, referred to as mining, provides cryptocurrency rewards to users who contribute computational resources through mechanisms like Proof of Work. Bitcoin, the most prominent cryptocurrency, serves as the primary focus of our study, though the methodology is readily applicable to other digital or even traditional financial assets.

Trading of cryptocurrencies typically occurs on fragmented exchange platforms, which function similarly to foreign exchange markets via continuous double auction systems with limit order books. These decentralized platforms facilitate simultaneous trading of the same asset across multiple venues, presenting both opportunities and challenges for market participants. The growing prevalence of algorithmic trading in these markets has underscored the importance of reliable short-term volume prediction. Accurate volume forecasts support optimal order execution, venue selection, and volatility modeling—key components in minimizing transaction costs and improving algorithmic performance.

In this work, we replicate and expand upon a recent study that tackles the problem of intraday volume prediction for Bitcoin across multiple exchanges (Antulov-Fantulin et al. [2020]). Specifically, we examine BTC/USD trading activity on the Bitstamp exchange, leveraging various limit order book and trading features to improve prediction accuracy. Unlike traditional approaches that consider each market in isolation, the original study introduces a Temporal Mixture Ensemble (TME) model designed to dynamically integrate multiple data sources from 2 exchanges (Bitstamp and Bitfinex) and quantify predictive uncertainty. Our project reproduces the key results of that study and explores several variations, including alternative model architectures, training windows, and data preprocessing techniques, to assess the robustness and adaptability of the proposed methodology.

[*] Course: Machine Learning for Finance and Complex Systems. Project: High-Frequency Cryptocurrency Volume Prediction.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

2 Dataset

Unlike the original work by Antulov-Fantulin et al. [2020], in our reproduction study we had access to data from only a single exchange, namely **Bitstamp**. Specifically, we obtained both transaction-level data and limit order book (LOB) BTC/USD data covering the period from *May 31, 2018 at 21:55 UTC* to *September 30, 2018 at 21:59 UTC*.

In order to replicate the multi-source modeling approach proposed in the original paper—which utilized data from both Bitfinex and Bitstamp—we treated the transaction data and LOB data from Bitstamp as two independent data sources. This results in a two-source setup, compared to the four-source setup in the original study (transaction and LOB data from both Bitstamp and Bitfinex).

A noteworthy limitation of our dataset is the presence of missing LOB data. In particular, data for the first three days of each calendar month were missing. Consequently, we excluded these timestamps from our analysis to maintain data integrity.

Following the methodology in Antulov-Fantulin et al. [2020], we extracted a set of features aggregated over 1-minute intervals. See Section A.3 for a complete overview of input features. Our prediction target is the **total traded volume**, i.e., the sum of buy and sell volumes over the target interval. We consider three prediction horizons: 1-minute, 5-minute, and 10-minute intervals.

Before model training, we conducted an exploratory analysis of the target variable. Figure 1a presents the distribution of the 1-minute log-transformed deseasonalized total traded volume. The distribution appears approximately Gaussian, although slightly negatively skewed. The distribution of 1-minute log-transformed volume and the corresponding intraday quantile patterns are shown in Figures 1a and 1b, respectively. The opening times of four major global stock exchanges are marked with vertical lines. While no abrupt volume changes are observed, we identify a mild but persistent intraday pattern in trading activity.

To remove these intraday seasonal effects, we normalize the volume series using the following transformation. Let v_t denote the raw volume at time t, and let $I(t)$ return the index of the intraday time slot to which t belongs. Define $a_{I(t)}$ as the average volume at time slot $I(t)$ across all days in the training set. The deseasonalized target variable is then given by: $y_t := \frac{v_t}{a_{I(t)}}$. To prevent information leakage, $a_{I(t)}$ is computed solely from the training dataset and is then used consistently for training, validation, and test splits.

3 Method

3.1 Baseline Model: ARMA-GARCH

To model the short-term dynamics and volatility of the log-transformed trading volume, we employ the ARMA-GARCH framework, a standard approach in financial time series analysis that captures both linear dependencies and conditional heteroskedasticity. Specifically, we use an ARMA(p, q)-GARCH(1,1) model defined as follows:

$$\log y_t = \mu + \sum_{i=1}^{p} \phi_i \log y_{t-i} + \sum_{j=1}^{q} \theta_j e_{t-j} + e_t \tag{1}$$

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

$$\sigma_t^2 = \omega + \alpha e_{t-1}^2 + \beta \sigma_{t-1}^2 \tag{3}$$

Here, y_t represents the de-seasonalized volume at time t, μ is the intercept, ϕ_i and θ_j are the autoregressive and moving average coefficients, respectively. The residual term e_t is modeled as a product of time-varying volatility σ_t and a standard normal innovation ϵ_t. The volatility dynamics follow a GARCH(1,1) process with parameters ω, α, and β.

The ARMA-GARCH model was trained using 80% of the available data, while the remaining 20% was reserved for testing. The optimal ARMA orders (p, q) were selected based on the Akaike Information Criterion (AIC). This model serves as one of our key econometric baselines, allowing us to benchmark its predictive performance against more complex machine learning approaches.

We compare our implementation with the results reported in the original paper across three aggregation levels: 1-minute, 5-minute, and 10-minute prediction horizons. Table 2 presents the optimal (p, q) values chosen by AIC for each frequency, along with the RMSE and MAE on the test set.

Our results indicate that the ARMA-GARCH model performs competitively, with superior predictive accuracy on 1-minute and 10-minute horizons compared to the baseline reported in the original study. However, for the 5-minute interval, our model slightly underperforms in terms of both RMSE and MAE. These differences could be attributed to the fact that we did not discard the zero-volume observations, as it was done in the original paper. Instead, we substituted zero-volume data points with the 10% of the minimal observed volume. This allowed the log transformation to be defined on those points.

3.2 Baseline Model - eXtreme Gradient Boosting (XGBoost)

Gradient Boosting Machine (GBM)Friedman [2000] was chosen as a baseline in the original work by Antulov-Fantulin et al. [2020] because it has been shown empirically to be highly effective in various machine learning challenges Gulin et al. [2011], Ben Taieb and Hyndman [2013] and has been applied successfully to tackle problems in finance Zhou et al. [2015], Sun et al. [2018]. Instead of the GBM, we used eXtreme Gradient Boosting (XGBoost) as a baseline because it is an extension of the GBM with better performance Chen and Guestrin [2016]. XGBoost, like GBM, has seen excellent results in many machine learning challenges.

XGBoost takes as input the features from both the transactions and order book. The features at time $t-1$ are used to predict the log of the deseasonalised volume, $\log y_t$, at time t. The data is ordered by time. The first 70% of the dataset is used for training and the last 20% for testing, with the remaining 10% used for validation. The hyperparameters were selected via 500 trials of random search on a grid: the number of estimators are in the range [100,1000], the fraction of subsampling in [0,1], minimum child weight in [2,9], maximum tree depth in [4,9], learning rate in [0.005,0.05].

3.3 TME Ensemble Model

Similarly to Antulov-Fantulin et al. [2020], we adopt a temporal mixture modeling framework to forecast the parameters of the volume-generating distribution. In the original paper the trading volume was modeled as log-normally distributed, implying that the logarithm of deseasonalized volume follows a Gaussian distribution. This choice reflects the empirical asymmetry and heavy-tailed behavior observed in raw volume data and further encodes the fact that the volume is never negative. Alternative distributional choices can be a direction for future work.

Let $\mathbf{x}_{s,(-h,t)}$ denote the input features from source $s \in \{\text{TRX}, \text{LOB}\}$ over a historical window of length h. We assume the following probabilistic model:

$$\begin{gathered}
\ln(y_t) \mid z_t = s, \mathbf{x}_{s,(-h,t)} \sim \mathcal{N}\left(\mu_{t,s}, \sigma^2_{t,s}\right) \\
\mu_{t,s} := L_{\mu,s}^{\top} \cdot \mathbf{x}_{s,(-h,t)} \cdot R_{\mu,s} + b_{\mu,s} \\
\sigma^2_{t,s} := \exp\left(L_{\sigma,s}^{\top} \cdot \mathbf{x}_{s,(-h,t)} \cdot R_{\sigma,s} + b_{\sigma,s}\right)
\end{gathered}$$

Two bilinear regression models are used to estimate the distribution parameters μ and σ^2 based on transaction-level (TRX) and order book (LOB) features, respectively. Each model processes a fixed-length historical window of size h.

In addition, a third bilinear regression model is trained to act as a gating mechanism. This gate produces a context-dependent weighting over the expert predictions, conditioned on the full input feature set over the same h-length window. The final forecast is obtained as a mixture of the two expert distributions, with weights learned jointly via minimization of the negative log-likelihood of the observed log-volume.

$$\begin{gathered}
\mathbb{P}_{\omega}\left(z_t = s \mid \left\{\mathbf{x}_{s,(-h,t)}\right\}_1^S\right) = \frac{\exp(f_s(\mathbf{x}_{s,(-h,t)}))}{\sum_{k=1}^{S} \exp(f_k(\mathbf{x}_{k,(-h,t)}))} \\
f_s(\mathbf{x}_{s,(-h,t)}) = L_{z,s}^{\top} \cdot \mathbf{x}_{s,(-h,t)} \cdot R_{z,s} + b_{z,s}
\end{gathered}$$

In the ensemble model, multiple instances of the networks above were implemented and trained simultaneously with different random seeds.

3.3.1 Prediction

Point predictions were generated by averaging the prediction of models with different random seeds. Given the assumption that the deseasonalised volume y_t is log-normally distributed, the prediction of y_t is given by its expectation:

$$\mathbb{E}[y_t|z_t = s, \mathbf{x}_{s,(-h,t)}, \theta_s] = e^{\mu_{t,s}+\frac{1}{2}\sigma^2_{t,s}}.$$

The predicted variation was calculated as follows. During inference, each model $m = 1, \dots, M$ produces a point prediction through the mixture mean $\mu^{(m)} = \sum_s w_s^{(m)} \mu_s^{(m)}$, while the predictive uncertainty is captured by the mixture variance $\sigma^{2(m)} = \sum_s w_s^{(m)} \sigma_s^{2(m)}$, which accounts for both within-expert variability and disagreement among experts with

$$\sigma_s^{2(m)} = \mathbb{V}[y_t|z_t = s, \mathbf{x}_{s,(-h,t)}, \theta_s] = e^{\sigma^2_{t,s}-1}e^{2\mu_{t,s}+\sigma^2_{t,s}}.$$

Using the law of total variance, the ensemble decomposes the total uncertainty into aleatoric (average predictive variance) and epistemic (disagreement between models) components:

$$\mathrm{Var}\left[y_t\right] = \underbrace{\frac{1}{M}\sum_{m=1}^{M} \sigma^{2(m)}}_{\text{Aleatoric}} + \underbrace{\frac{1}{M}\sum_{m=1}^{M} \left(\mu^{(m)} - \bar{\mu}\right)^2}_{\text{Epistemic}}$$

This separation distinguishes inherent noise from model uncertainty in the final predictions.

3.3.2 Training and Hyper Parameter Tuning

The model exhibited significant instability, particularly when using 1-minute data frequency. We implemented several approaches to enhance performance and stability. First, as previously mentioned, we removed the intraday trend (learned from the training set) and normalized the feature values.

To further improve stability, we trained multiple TME models (TME ensemble) with different random seeds, and defined the prediction of the ensemble as the average of the individual predictions (means). We employed the Negative Log Likelihood function to evaluate how well the predicted distributions fit the validation set data and incorporated L^2 regularization to mitigate overfitting and prevent parameter value explosion:

$$-\sum_{t=1}^{T} \ln \sum_{z_t=1}^{S} \frac{1}{y_t \sigma_{t,s}\sqrt{2\pi}} \exp\left(-\frac{\left(\ln y_t - \mu_{t,s}\right)^2}{2\sigma^2_{t,s}}\right) \cdot \mathbb{P}_\omega\left(z_t = s \mid \left\{\mathbf{x}_{s,(-h,t)}\right\}_1^S\right) - \lambda\|\Theta\|_2^2$$

We conducted hyperparameter tuning via random search to optimize the TME ensemble model. From the parameter space of 9,600 possible combinations (see Table 3 for the hyperparameter grid) , we randomly sampled 50 configurations for evaluation. For each parameter combination, we performed 5-fold cross-validation where we trained an ensemble of n_{models} individual TME models on each training subset and validated on the corresponding validation subset. The training employed early stopping based on validation loss, retaining the model state from the epoch with minimum validation loss. The final performance metric for each parameter configuration was computed as the average validation loss across all folds. The optimal parameter set was selected based on this cross-validated performance.

The primary goal of hyperparameter tuning is to identify the configuration that yields the lowest average validation loss across the cross-validation folds. In addition, we aimed to maintain and improve model stability by analyzing the variance in validation loss under different hyperparameter settings. The maximum number of epochs, batch size, and lookback horizon (beyond the minimal value of 50) had only a minor effect on model performance. In contrast, the $L2$ penalty, learning rate, and ensemble size had an extensive and significant impact. $L2$ and learning rate matter most because volume data is noisy, so regularization prevents overfitting to spurious fluctuations, while the right learning rate ensures stable training without overshooting. Ensemble size helps significantly because averaging multiple models reduces variance, which is critical for noisy volume predictions. Epochs, batch size, and lookback matter less because volume trends are often short-term and repetitive, so extra training or sequence context adds little new information.

Figure 10 summarizes how each hyperparameter independently affects the validation loss. To better understand interactions between variables, we also analyzed their effects when conditioned on specific parameter values. Figure 11 shows the role of batch size across different horizon settings. Figure 12 explores how increasing training epochs impacts models with varying prediction horizons. Figure 13 presents ensemble size behavior as a function of prediction horizon. Figure 14 analyzes the interaction between lookback window length and ensemble size. Figure 15 visualizes how learning rate and L2 regularization strength affect training stability (measured via the standard deviation of validation loss).

An increase in the learning rate led to a higher standard deviation of the validation loss, as expected, since the model tends to overshoot during optimization. In contrast, stronger $L2$ regularization not only improved performance (as previously discussed) but also reduced variance, which aligns with expectations, as regularization generally encourages smaller weights and thereby reduces sensitivity to input fluctuations.

The optimal configuration ($L2 = 0.1$, $lr = 0.1$, $n_{models} = 10$) confirms our expectations: strong regularization and careful learning rate tuning were essential for this noisy volume prediction task, while the minimal horizon (50) and moderate epochs (15) sufficed. The large batch size (4096) had negligible impact, but the ensemble's 10-model variance reduction proved critical.

After determining the optimal hyperparameters, we implemented two distinct training approaches to enhance model performance and generalizability. First, we trained a single model using the optimal hyperparameters on the full training set (we split the data into 70% training, 10% validation, and 20% test sets), selecting the number of epochs based on validation performance to prevent overfitting. Subsequently, to further improve robustness, we employed 5-fold cross-validation, training five separate models on different training folds while using their respective validation sets for epoch selection. For final predictions, we created an ensemble by averaging predictions from all five models ($n_{\text{models}} \times$ number of folds). This ensemble approach served two key purposes: (1) mitigating the impact of random initialization effects through model averaging, and (2) reducing overfitting to any particular data partition by leveraging multiple training-validation splits.

4 Improvements and Modifications

4.1 Stochastic Weight Averaging Gaussian (SWAG)

A modification to the TME was tested by introducing SWAG. In TME, the predictions are the average of multiple networks with the variation in network weights, and hence predictions due to the random initialization of the network weights. In SWAG, the variation in the network weights is the result of sampling network weights from their posterior distribution via stochastic gradient descent Maddox et al. [2019]. The training procedure is faster and more efficient than the training for the standard ensemble because only one network needs to be pretrained before it can be used to sample network weights from the posterior as opposed to the need to train as many networks as required in the standard ensemble method.

4.2 Batch Normalisation

Instead of using standardisation in the preprocessing step, a batch normalisation layer can be introduced as the first layer that receives the input features. This would allow the network to be deployed in an online setting because the running estimates of the mean and variance can be used at inference time.

4.3 Weighted Averaging of Networks

The TME combines the component networks with equal weight. An obvious extension of this idea is of course to weigh the component networks with time varying weights. For example, in a two network ensemble, the first network may have a weight of 0.9 for odd time steps and a weight of 0.1 for even time steps. The weights with which to combine the component network predictions can be interpreted as probabilities and a network can be trained to learn them. The output of the network is a softmax layer and the two versions of the network were tested: one used as input the predictions of the TME component networks and the other used the input features obtained from the transaction and order book data. The training objective is the squared loss.

5 Experiments

5.1 Evaluation Metrics

The model is trained on the deseasonalized log-volume, but the evaluation metrics are computed on the original (raw) volume data. We evaluate model performance using several metrics. Root Mean Square Error (RMSE) and Mean Absolute Error (MAE) measure the deviation between predicted and true raw volumes, with RMSE defined as $\sqrt{\frac{1}{T}\sum_{t=1}^{T}\left(v_t-\hat{v}_t\right)^2}$ and MAE as $\frac{1}{T}\sum_{t=1}^{T}\left|v_t-\hat{v}_t\right|$. We also report the coefficient of determination $\left(\mathrm{R}^2\right)$ to assess explained variance, and Mean Absolute Percentage Error (MAPE).To jointly capture accuracy and uncertainty calibration, we compute the Normalized Negative Log-Likelihood (NNLL). Uncertainty is further evaluated using the Interval Width (IW), defined as the mean predictive standard deviation $\mathrm{IW}=\frac{1}{T}\sum_{t=1}^{T}\hat{\sigma}_t$ where $\hat{\sigma}_t^2$ denotes the model's predictive variance. Finally, Coverage (C) quantifies the proportion of true values falling within a specified prediction interval (e.g., 95%), indicating the calibration quality of the predictive distribution under a normality assumption.

5.2 TME model

As shown in Figure 7 we can observe the learning curves of the ensemble models, with the average learning curve highlighted in a darker color. The curves of the individual models are nearly indistinguishable from each other, which means the random seed has minimal impact on the models' performance, especially as training progresses through the epochs.

The parameter importance was calculated by summing the time-specific betas for each feature. In terms of transaction parameter importance, six parameters have almost equal importance, though those directly related to total volume show slightly higher beta values. As depicted in Figure 8, some features dominate the 10-minute horizon predictions. For order book features, slope — particularly at short horizons — plays a dominant role in predictions. Spread, volume imbalance also contribute significantly. Interestingly, small ask slopes exhibit higher beta values, suggesting that small incoming buy orders have a stronger impact when the ask side is shallow. On the bid side, however, the 10% bid slope is more influential than smaller bid slopes, implying that deeper liquidity levels on the bid side better explain future outcomes. This asymmetry may reflect behavioral differences: buy orders tend to be smaller and more reactive, while sell orders may arrive in larger sizes, making their effect more visible at aggregated depth. These findings are consistent with the observed asymmetry in the order book volume distribution.

We obtained similar results at both the one-minute frequency (see Section A.6) and the five-minute frequency (see Section A.8).

Figure 9 shows the model's predictions against actual values. Across all data frequencies, we observed significantly lower prediction variability compared to a single-model (non-ensemble), but at the cost of a reduced coverage ratio. Our implementation slightly underperformed the paper's reference model. This discrepancy may stem from several factors: Data source differences, missing value handling, implementation nuances (e.g., hyperparameter tuning or computational constraints).

Table 1 provides a comprehensive summary of all experimental results, comparing our implementations with those reported in the original paper.

While absolute errors (RMSE, MAE) increase with the frequency window, as expected due to the cumulative nature of volume, relative errors (MAPE) and explanatory power $\left(R^2\right)$ deteriorate, indicating a decline in predictive accuracy. Moreover, the drop in coverage at the 10-minute frequency suggests poor uncertainty quantification, despite the interval width (IW) increasing consistently with the forecast horizon. Interestingly, the normalized negative log-likelihood (NNLL) improves at higher aggregation levels, potentially reflecting better calibration under the normality assumption. However, this contrasts with the worsening MAPE and R^2, which suggest that the model fails to capture underlying dynamics effectively at coarser scales. These empirical results challenge the common assumption that aggregation over time leads to more Gaussian-like behavior in the target distribution, as implied by the Central Limit Theorem. In the case of volume prediction, it appears that aggregation may not reduce noise or improve normality in a meaningful way, possibly due to heavy-tailed or autocorrelated behavior in the underlying volume process.

5.3 TME - alternative implementations with training tricks

In addition to reproducing the original Temporal Mixture Ensemble (TME) model, we explored several implementation enhancements aimed at improving its predictive performance and robustness. The key modifications in our alternative setup are as follows for the 1 minute frequency:

- All input features were standardised to have zero mean and unit variance, a common preprocessing step that often facilitates faster and more stable convergence of neural network models.
- Instead of modeling the de-seasonalised log-volume, our variant was trained directly on the log-transformed raw volume. This simplifies the preprocessing pipeline and may help the model learn the full variance structure inherent in the original signal.
- Hyperparameter selection was carried out using a grid search strategy over a predefined range of values, optimising for validation set performance.

These adjustments led to a modest but consistent improvement in predictive accuracy. Table 1 summarises the results between the original different models' implementations and this alternative setup for the 1 minute prediction interval is labeled there as **TME (Alt. Implementation)**.

The results suggest that our version of the TME model outperforms the original one in terms of both point prediction accuracy (RMSE and MAE) and predictive uncertainty calibration (interval width). These findings indicate that careful data preprocessing and hyperparameter tuning can yield meaningful improvements, even within the same model architecture.

Minor alterations were tested on the 10 minute frequency. These are:

- All input and target features were standardised to have zero mean and unit variance, a common preprocessing step that often facilitates faster and more stable training of machine learning models when the features have wildy different ranges of values.
- Full batch gradient descent was used instead of stochastic gradient descent. This should lead to better convergence of the network weights at the cost of longer training times.
- Hyperparameter selection was carried out using a grid search over the window size in range [3,24 × 6] and learning rates in $\{10^{-3}, 10^{-4}, 10^{-5}\}$, with the stopping criteria for training determined by the network with the lowest loss evaluated on the validation set. With full batch gradient descent, the trajectories of the network parameters during training are deterministic, given a particular initialisation, and lower learning rates were used to continue training from the stopping point of larger learning rates.

Since the target variable, $\log y_t$, was also standardised, the formula for the prediction of y_t becomes slightly different. Defining $\log \hat{y}_t = \frac{\log y_t - \hat{\mu}}{\hat{\sigma}}$, where $\hat{\mu}, \hat{\sigma}$ are the empirical mean and standard deviation of $\log y_t$ respectively, the network learns the mean and variance of $\log \hat{y}_t \sim \mathcal{N}(\mu, \sigma^2)$. Some algebra leads to

$$\log y_t = \log \hat{y}_t \times \hat{\sigma} + \hat{\mu} \sim \mathcal{N}(\mu \times \hat{\sigma} + \mu, \sigma^2 \times \hat{\sigma}^2),$$

so the expectation of y_t is

$$\mathbb{E}[y_t | z_t = s, \mathbf{x}_{s,(-h,t)}, \theta_s] = e^{\hat{\mu} + (\mu_{t,s} + \frac{\hat{\sigma}}{2}\sigma^2_{t,s})\hat{\sigma}}.$$

5.4 Stochastic Weight Averaging Gaussian (SWAG)

SWAG was implemented on top of the alternative implementation of TME for the 10 minute frequency, with standardisation of features and targets. The hyperparameters of the network are the same as the ones used in the alternative implementation. Only very minor hyperparameter tuning for the SWAG was performed. The pretrained network was trained for a further 20 epochs with stochastic gradient descent to obtain 20 sets of network weights. The learning rate was 10^{-4}, batch size 256, and rank of the deviation matrix was 20.

Very minor improvements can be observed in all 4 metrics, but the training time is decreased significantly.

5.5 Batch Normalisation

Batch Normalisation was implemented on top of the alternative implementation of TME for the 10 minute frequency, with standardisation of features and targets replaced with the batch normalisation layer. The hyperparameters of the network are the same as the ones used in the alternative implementation.

The results are extremely poor due to the existence of a few networks in the ensemble that converged to bad local minima during training. The bad predictions heavily distort the overall predictions of the TME. The existence of divergent predictions leads to the interval width being undefined.

5.6 Weighted Averaging of Networks

Weighted averaging of the networks in the ensemble was implemented on top of the alternative implementation of TME for the 10 minute frequency, with standardisation of features and targets. The hyperparameters of the network are the same as the ones used in the alternative implementation. Two versions of the weighting network was tested. One which took as input the predictions of the ensemble component networks, and the other took as input the features from the transactions and order book data.

The training of the weighting network which takes as input the predictions of the component networks converges very fast which means the network weights are very close to the initial weights. Upon further inspection of the weighting network, only 1 network is assigned a high probability in the softmax for all time steps. Since no hyperparameter tuning has been done, the results can possibly be improved with proper hyperparameter selection. Training of the weighting network with transaction and order book features as input is much longer and does not reduce to one network dominating the predictions. However, the metrics in this case are worse than those from the weighting network with input predictions, so further investigation needs to be done.

6 Conclusion

We presented a Temporal Mixture Ensemble model for intraday volume forecasting in cryptocurrency markets. By leveraging time-dependent importance attribution and probabilistic supervision, the TME outperforms the baseline models.

After re-implementing the base strategies and successfully replicating the performance reported in the paper, we re-implemented the main models as well. We took several steps to improve model stability and introduced rigorous hyperparameter tuning. During parameter testing, we applied an optimized ensemble method to eliminate the effect of random seeds, and we used cross-fold validation to neutralize sample randomness and prevent overfitting. Once again, we achieved results comparable to those in the original paper, but with a much deeper understanding of the training process and the influence of each parameter.

The introduction of SWAG to the TME was shown to achieve comparable results with the added benefit of much shorter training time. In our specific implementation, one component network in the TME was pretrained and the 20 component networks were obtained by simply training for 20 additional epochs to collect the SGD iterates and sample network parameters from the posterior distribution.

The terrible performance of the TME with a batch normalisation layer is due to there being networks which converged to local minima with high loss in the TME. This highlights the possibility in general of network ensembling methods leading to worse results because of poorly trained networks. Removing these networks from the ensemble should enable batch normalisation to be used instead of standardisation of training features.

All the modifications and improvements to the basic TME method can be combined together. Future work can be done to investigate whether this would yield better and more robust volume predictions.

Other potential directions that the TME discussed here can be further improved include modeling the deseasonlised volume y_t with other distributions that reflect the data better than the log normal distribution assumed here. For higher frequency data, the log normality assumption becomes less valid. An attention mechanism can be included to weigh the contributions from the different sources of data or even the different component networks that make up the ensemble.

Table 1: Performance comparison across models and time frequencies on the **Bitstamp** market. Paper values are taken from the original study; "Ours" are our re-implementations or ablations.

Model	Freq	RMSE	MAE	NNLL	IW
GARCH (Paper)	1-min	14.587	7.688	1.719	97.618
GARCH (Ours)	1-min	12.018	3.638	NA	NA
GBM (Paper)	1-min	11.740	3.515	NA	NA
GBM (Ours)	1-min	11.360	**3.390**	NA	NA
TME (Paper)	1-min	11.378	4.299	1.720	10.295
TME (Ours)	1-min	11.780	4.360	4.950	49.260
TME (Alt. Implementation)	1-min	**11.130**	3.584	NA	**10.098**
GARCH (Paper)	5-min	38.300	17.606	3.732	34.797
GARCH (Ours)	5-min	41.213	18.142	NA	NA
GBM (Paper)	5-min	39.196	14.714	NA	NA
GBM (Ours)	5-min	**34.970**	**13.340**	NA	NA
TME (Paper)	5-min	38.223	17.287	3.765	**29.148**
TME (Ours)	5-min	35.610	18.370	2.810	39.760
GARCH (Paper)	10-min	66.486	31.942	4.452	51.228
GARCH (Ours)	10-min	62.339	27.827	NA	NA
GBM (Paper)	10-min	67.128	27.719	NA	NA
GBM (Ours)	10-min	**55.400**	**24.380**	NA	NA
TME (Paper)	10-min	66.234	31.460	4.507	**49.972**
TME (Ours)	10-min	58.040	35.210	2.380	73.340
TME Alt. (for SWAG)	10-min	60.78	36.640	218.68	77.99
SWAG	10-min	60.160	**36.580**	210.00	77.87
Batch Norm	10-min	∞	4.31×10^{172}	194.48	NA
Weighted Avg (Pred)	10-min	**60.015**	36.588	205.76	**77.62**
Weighted Avg (Feat)	10-min	79.841	37.377	226.50	79.77

We tested multiple alternative distributions to model order size, and found that the t-distribution provided the best fit. This result is intuitive, as order sizes often exhibit heavy-tailed behavior due to the presence of both small routine trades and occasional large institutional orders. It is intuitive because order flow and intraday volume often exhibit heavy tails and occasional bursts of extreme activity, which the t-distribution can capture better than the normal distribution. We recognized that based on Azzalini and Capitanio [2003] the asymmetry of the t-distribution could be better captured using an $Azzalini - type$ skew-t distribution. In Appendix A.10, we present comparative results demonstrating its potential. However, due to the significantly higher computational cost during training, we opted to retain the standard normal distribution for our final model. This decision balances performance with practicality, though the skew-t distribution remains a promising option for future improvements if greater computational resources become available.

References

Nino Antulov-Fantulin, Tian Guo, and Fabrizio Lillo. Temporal mixture ensemble for probabilistic forecasting of intraday volume in cryptocurrency exchange markets. *https://arxiv.org/abs/2005.09356*, 2020. URL `https://arxiv.org/abs/2005.09356`.

Adelchi Azzalini and Antonella Capitanio. Distributions generated by perturbation of symmetry with emphasis on a multivariate skew t-distribution. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 65(2):367–389, 2003. doi: 10.1111/1467-9868.00391.

Souhaib Ben Taieb and Rob Hyndman. A gradient boosting approach to the kaggle load forecasting competition. *International Journal of Forecasting*, 30, 01 2013. doi: 10.1016/j.ijforecast.2013.07.005.

Tianqi Chen and Carlos Guestrin. XGBoost: A Scalable Tree Boosting System. *arXiv e-prints*, art. arXiv:1603.02754, March 2016. doi: 10.48550/arXiv.1603.02754.

Jerome Friedman. Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29, 11 2000. doi: 10.1214/aos/1013203451.

Andrey Gulin, Igor Kuralenok, and Dimitry Pavlov. Winning the transfer learning track of yahoo!'s learning to rank challenge with yetirank. In Olivier Chapelle, Yi Chang, and Tie-Yan Liu, editors, *Proceedings of the Learning to Rank Challenge*, volume 14 of *Proceedings of Machine Learning Research*, pages 63–76, Haifa, Israel, 25 Jun 2011. PMLR. URL `https://proceedings.mlr.press/v14/gulin11a.html`.

Wesley Maddox, Timur Garipov, Pavel Izmailov, Dmitry Vetrov, and Andrew Gordon Wilson. A Simple Baseline for Bayesian Uncertainty in Deep Learning. *arXiv e-prints*, art. arXiv:1902.02476, February 2019. doi: 10.48550/arXiv.1902.02476.

Xiaolei Sun, Mingxi Liu, and Zeqian Sima. A novel cryptocurrency price trend forecasting model based on lightgbm. *Finance Research Letters*, 32, 12 2018. doi: 10.1016/j.frl.2018.12.032.

Nan Zhou, Wen Cheng, Yichen Qin, and Zongcheng Yin. Evolution of high-frequency systematic trading: a performance-driven gradient boosting model. *Quantitative Finance*, 15:1387–1403, 08 2015. doi: 10.1080/14697688.2015.1032541.

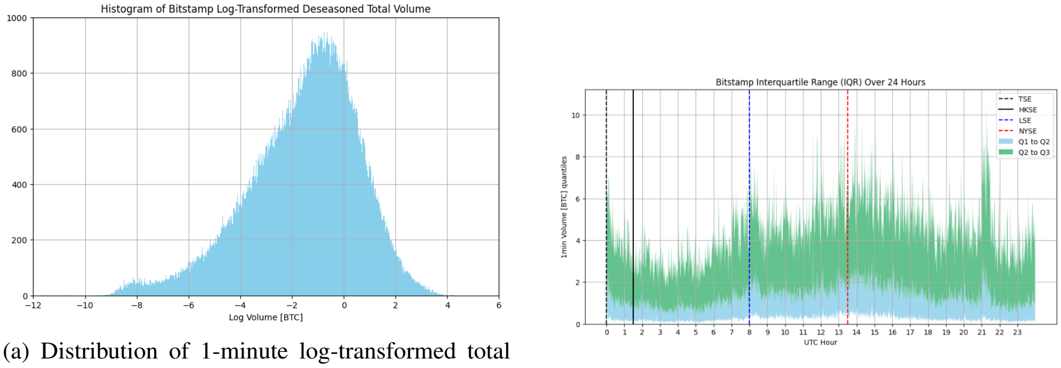
A Appendix A: Additional Plots and Results

A.1 ARMA-GARCH results

Table 2: ARMA-GARCH performance comparison with original paper

Frequency	Optimal (p, q)		RMSE		MAE	
	Paper	Ours	Paper	Ours	Paper	Ours
1-min	(3,3)	(6,7)	14.587	**12.018**	7.688	**3.638**
5-min	(5,4)	(6,1)	**38.300**	41.213	**17.606**	18.142
10-min	(3,2)	(3,7)	66.486	**62.339**	31.942	**27.827**

A.2 Volume Distribution Plot



(a) Distribution of 1-minute log-transformed total traded volume.

(b) Intraday quantiles of BTC/USD 1-minute volume.

Figure 1: Exploratory analysis of BTC/USD volume data.

A.3 List of features

From transaction data:

- **Buy volume** — total BTC volume of buyer-initiated transactions.
- **Sell volume** — total BTC volume of seller-initiated transactions.
- **Volume imbalance** — absolute difference between buy and sell volumes.
- **Buy transactions** — number of executed buy-side transactions.
- **Sell transactions** — number of executed sell-side transactions.
- **Transaction imbalance** — absolute difference between buy and sell transaction counts.

From limit order book (LOB) data:

- **Spread** — the difference between the best ask and best bid prices.
- **Ask volume** — total BTC volume available on the ask side of the order book.
- **Bid volume** — total BTC volume available on the bid side of the order book.
- **Imbalance** — absolute difference between ask and bid volumes.
- **Ask/Bid slope** — volume-weighted slope up to a price offset δ from the best ask/bid price. δ is estimated using the bid price at which cumulative volume reaches 1%, 5%, and 10% of the top bid orders.
- **Slope imbalance** — absolute difference between ask and bid slopes, computed at the same δ thresholds.

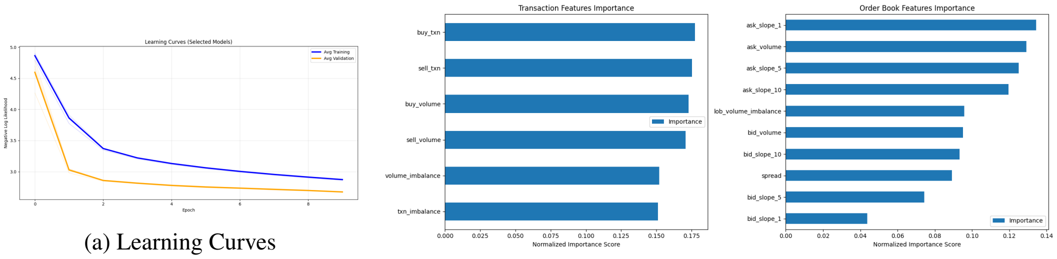
A.4 Hyperparameter grid used for tuning

Hyperparameter	Values
Lookback Horizon (h)	$\{50, 75, 100, 125, 150\}$
Learning Rate (η)	$\{0.1, 0.01, 0.001, 0.0001\}$
L2 Regularization (λ)	$\{0.0, 0.0001, 0.001, 0.01, 0.1\}$
Batch Size (B)	$\{2048, 4096, 8192, 16384\}$
Ensemble Size (M)	$\{5, 10, 15, 20\}$
Training Epochs (T)	$\{5, 10, 15, 20, 25, 30\}$

Table 3: Hyperparameter grid used for tuning.

A.5 Full Results Overview

A.6 1 Minute results



(a) Learning Curves

(b) Variable Importance

Figure 2: Model diagnostics showing (a) training convergence and (b) feature importance scores

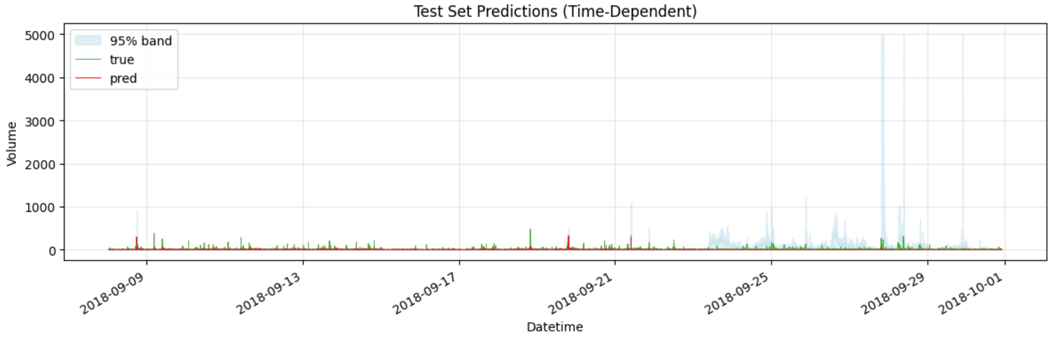


Figure 3: Prediction results on test data

A.7 5 Minute results

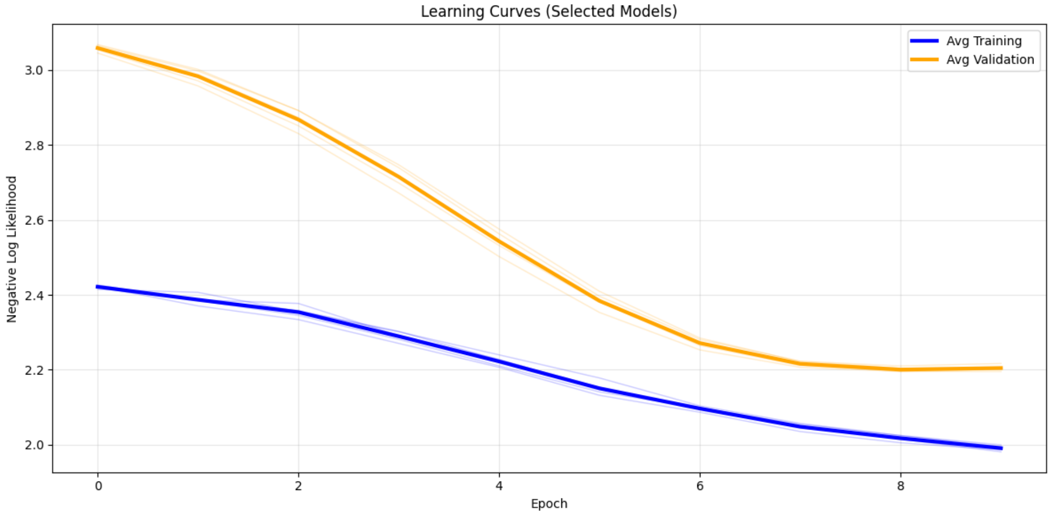


Figure 4: Learning curves of the model on 5-minute frequency data.

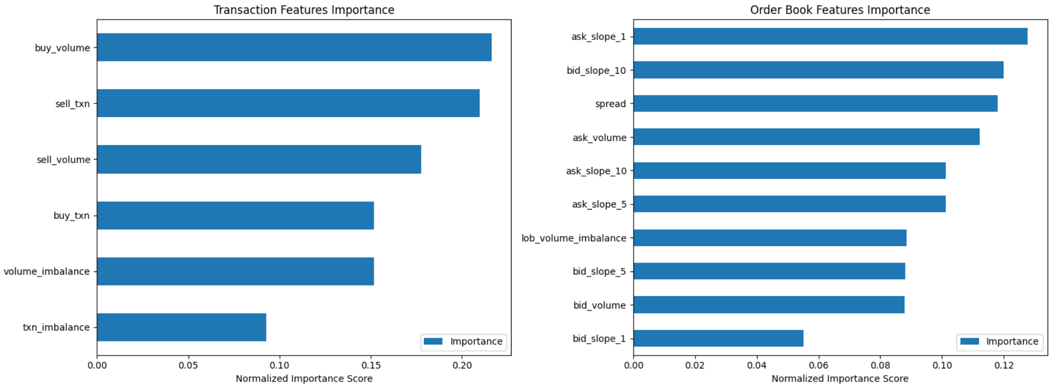


Figure 5: Feature importance scores for the 5-minute prediction horizon.

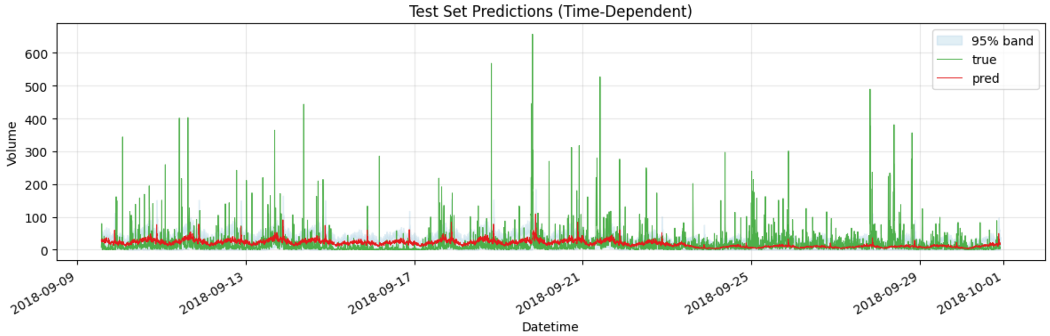


Figure 6: Prediction results on test data

A.8 10 Minute results

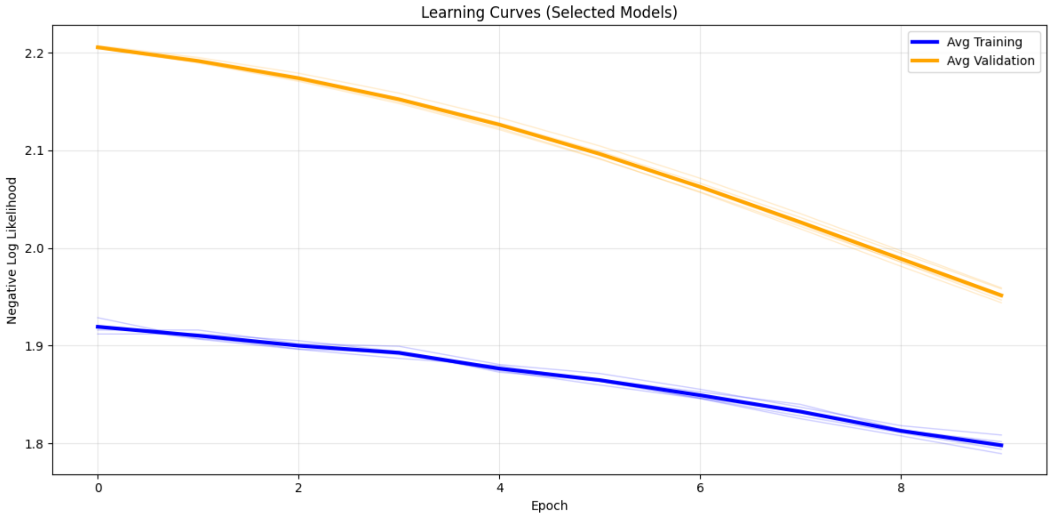


Figure 7: Training convergence (freq = 10 min)

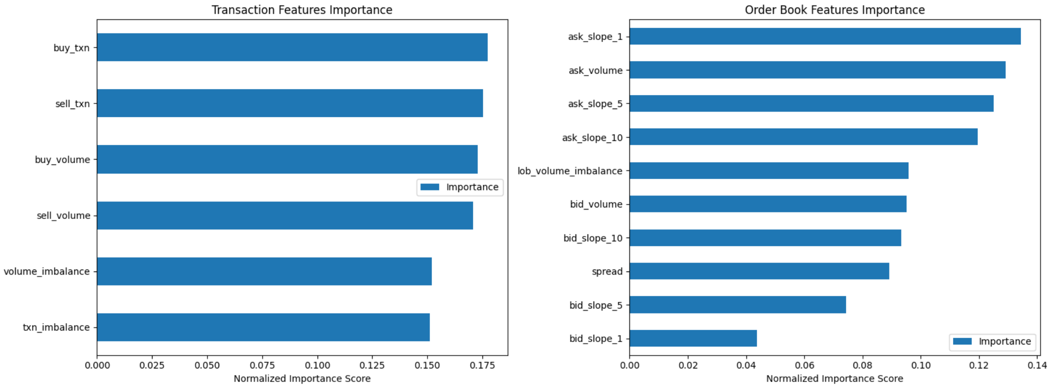


Figure 8: Feature importance scores (freq = 10 min)

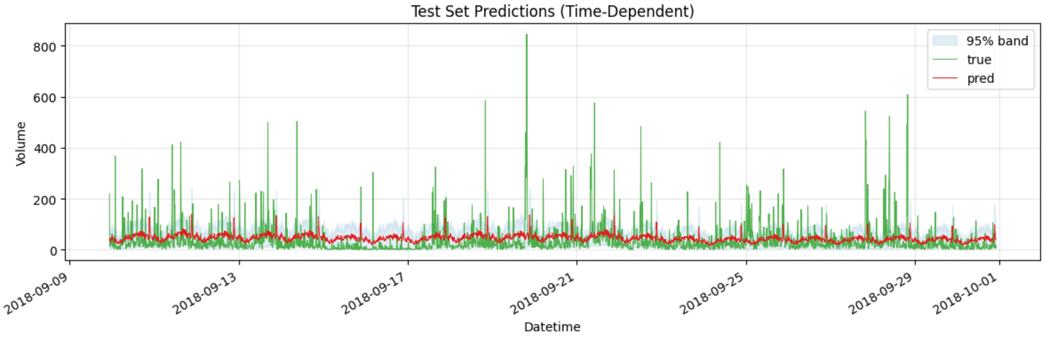


Figure 9: Prediction results on test data (freq = 10 min)

A.9 Hyper Parameter Tuning Results

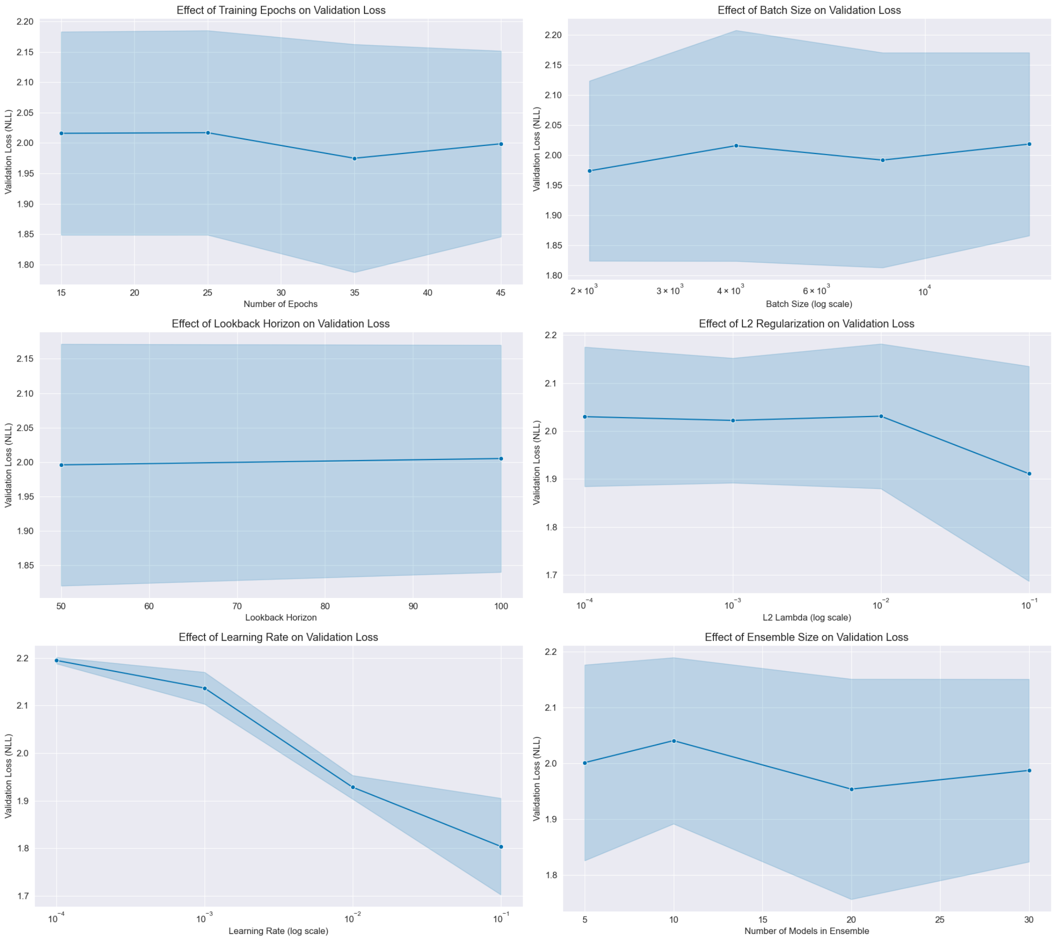


Figure 10: Overview of individual hyperparameter effects on validation loss: learning rate, L2 regularization, batch size, number of epochs, lookback horizon, and ensemble size.

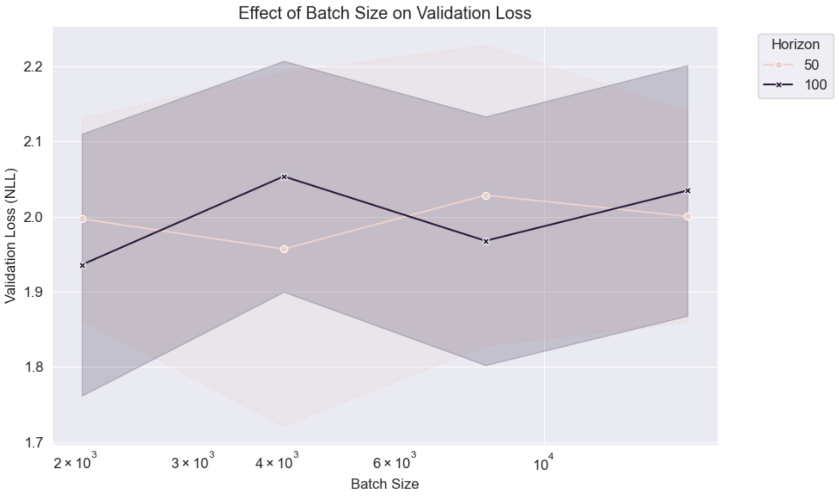


Figure 11: Effect of batch size on validation loss for different prediction horizons.

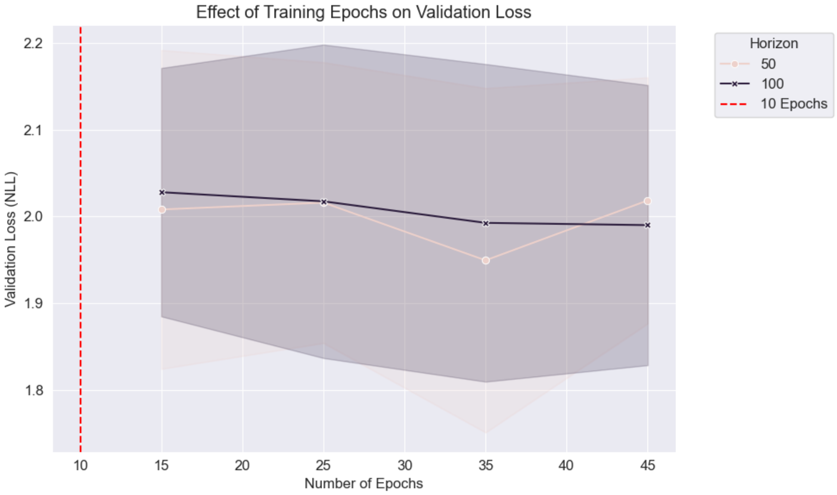


Figure 12: Effect of training epochs on validation loss under different horizon lengths.

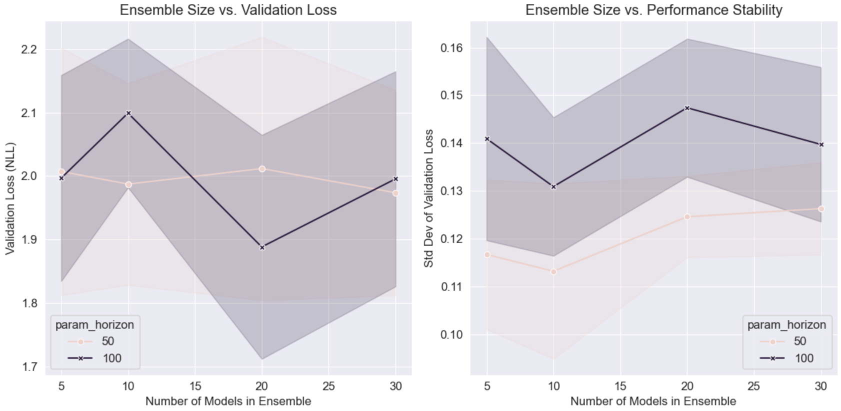


Figure 13: Effect of ensemble size on validation loss across different horizon lengths.

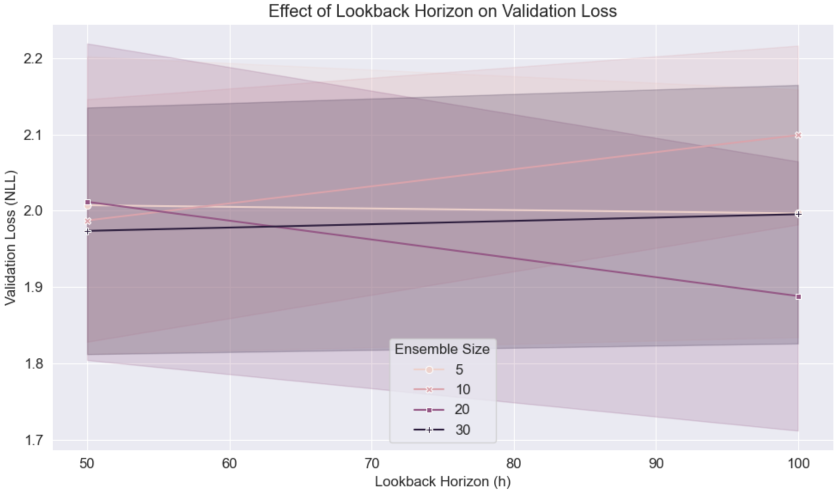


Figure 14: Impact of lookback window size across ensemble sizes on validation performance.

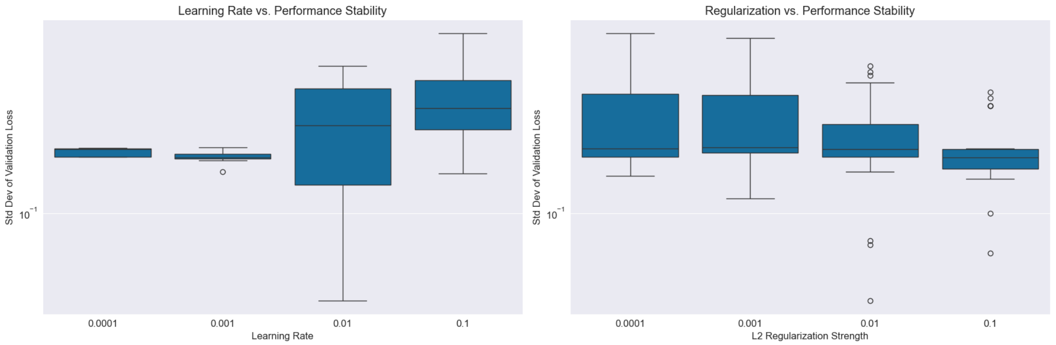


Figure 15: Standard deviation of validation loss under different learning rates and L2 regularization strengths, showing model stability.

A.10 Alternative Distributions for Probabilistic Volume Modeling

Table 4: Kolmogorov-Smirnov Goodness-of-Fit Statistics

Distribution	**KS Statistic**
Student-t	0.0438
Generalized Extreme	0.0489
Normal	0.0571
Laplace	0.0569
Gamma	0.0658
Inverse-Gaussian	0.0861
Log-Normal (LNP)	0.7792
Generalized Pareto	0.7792

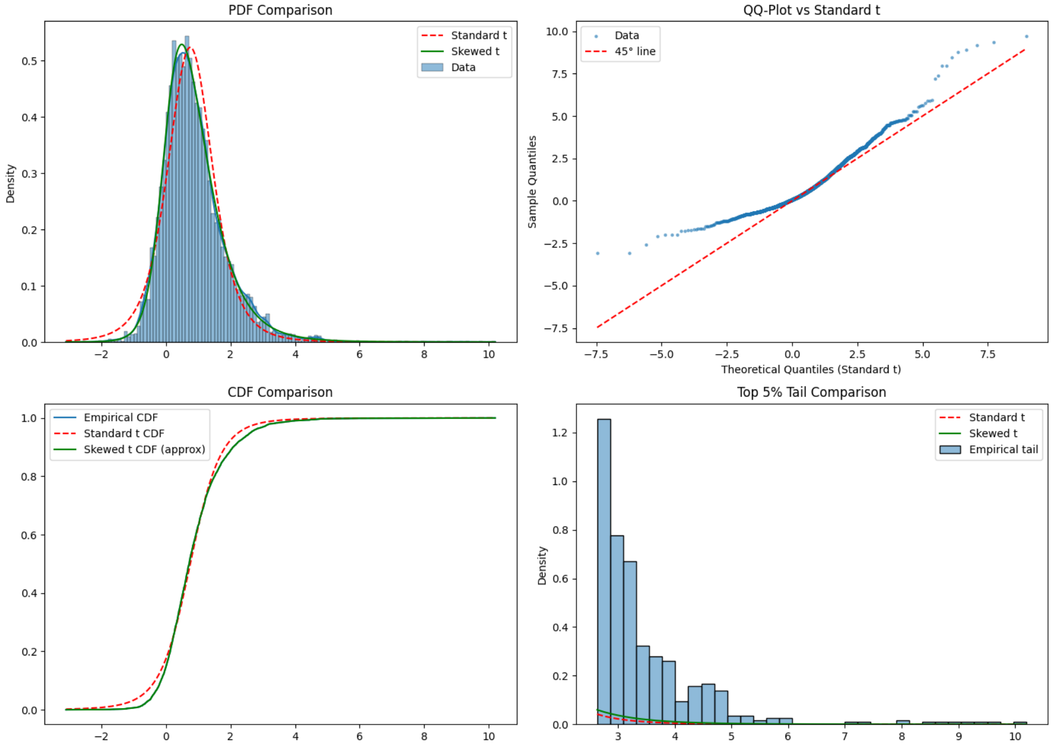
Skewed Student's t-Distribution Probability Density Function:

$$f(x;\xi,\omega,\alpha,\nu) = \frac{2}{\omega} t_\nu\left(\frac{x-\xi}{\omega}\right) T_{\nu+1}\left(\alpha \cdot \frac{x-\xi}{\omega} \cdot \sqrt{\frac{\nu+1}{\nu + \left(\frac{x-\xi}{\omega}\right)^2}}\right)$$

Where:

- t_ν : standard t PDF
- $T_{\nu+1}$: CDF of t-distribution
- ξ : location
- $\omega > 0$: scale
- α : skewness
- $\nu > 0$: degrees of freedom

Goodness-of-Fit Diagnostics for Standard and Skewed Student's t-Distributions Against Empirical Data:


PDF Comparison
Density
Standard t
Skewed t
Data
0.0
0.1
0.2
0.3
0.4
0.5
−2
0
2
4
6
8
10
QQ-Plot vs Standard t
Sample Quantiles
Theoretical Quantiles (Standard t)
Data
45° line
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0
CDF Comparison
Empirical CDF
Standard t CDF
Skewed t CDF (approx)
0.0
0.2
0.4
0.6
0.8
1.0
Top 5% Tail Comparison
Density
Standard t
Skewed t
Empirical tail
0.0
0.2
0.4
0.6
0.8
1.0
1.2
3
4
5
6
7
8
9
10