Erasmus School of Economics

Master Thesis Econometrics and Management Science - Quantitative Finance

The effect of sampling frequency on volatility forecasting performance using an LSTM neural network

June 1, 2023

Name student:

Joost van Tol

Student ID:

434522

Supervisor:

Dr. Marina Khismatullina

Second assessor:

Prof. Dr. P.H.B.F. Franses

Abstract

Accurately forecasting volatility is important in the whole financial sector. Nowadays, high frequency data is available to make more accurate forecasts. This thesis researches the effects of the sampling frequency of the data on the forecasting performance of volatility. Millisecond quote data is used of the SPDR S&P 500 ETF. An LSTM neural network is used to make forecasts using different frequencies of data. The forecasts are evaluated against the realized volatility, a proxy for the actual latent volatility. The results show that increasing sampling frequency does increase forecasting accuracy. It provides support for using a volatility signature plot to determine the optimal frequency for forecasting volatility.

The content of this thesis is the sole responsibility of the author and does not reflect the view of the supervisor, second assessor, Erasmus School of Economics or Erasmus University.

Contents

1 Introduction

In financial time series, volatility is a measure of the fluctuation of the given return time series. With securities, volatility can also be seen as a measure of the riskiness of investing in the security. A more volatile security is considered riskier because of the larger price movements and thus the larger potential loss. On the other hand, more volatile securities also have potentially larger returns. To correctly price and assess riskiness of securities it is of great importance to be able to accurately forecast volatility. In pricing derivatives, volatility is one of the most important variables. In pricing options, for example, volatility is a direct input parameter to calculate the options price (Poon & Granger, 2003). Also in financial risk management and portfolio management it is important to be able to accurately forecast volatility to accurately measure Value at Risk or the riskiness of a portfolio (González-Rivera et al., 2004). Because of the importance, a broad literature regarding measuring and forecasting volatility has been developed in the past decades.

A difficulty with volatility is that it is inherently a latent variable and thus not directly observable. The most used proxy of the actual volatility in financial forecasting articles is the realized volatility (Wilhelmsson, 2006). Realized volatility at time t (σ_t) is calculated as:

$$\sigma_t = \sqrt{\sum_{i=1}^{n} r_{i-1,i}^2}, \tag{1}$$

where $r_{i-1,i}^2$ is the squared return between time $i-1$ and i. It is a non-parametric approach that does not rely on any assumptions on the underlying distribution or restrictions on parameters that need to be estimated like in stochastic volatility models or conditional heteroscedasticity models. This makes it an easy to use proxy of volatility and resulted in an important stream of literature on the properties of this non-parametric approach (Bucci et al., 2017). As shown by Andersen and Bollerslev (1998), realized volatility is a consistent estimator of the actual volatility when the frequency of the returns goes to infinity. In practice this does not work due to the microstructure noise in high frequency data. Microstructure noise can be seen as the noise in the observed returns caused by imperfections in the trading process like discreteness of prices, bid-ask bounce or irregular trading (Bai et al., 2000; Bandi & Russell, 2003). As noted by Zhang et al. (2005) and Bandi and Russell (2003), the realized volatility estimator fails to converge to the actual volatility when sampling frequency increases. Instead, it seams to increase at higher frequencies for liquid assets and decrease at higher frequencies for illiquid assets. Bandi and Russell (2007) state that because of the microstructure noise in high frequency data there is a bias-variance trade-off in estimating actual volatility with realized volatility. Increasing the sampling frequency will increase the noise component in the observed return series causing the realized volatility to be a more biased estimate of the actual volatility but will lower the variability. At sufficiently low frequencies this bias can be negligible but the variability in the estimate of the actual volatility can be substantial (Bandi & Russell, 2007). Therefore, to find an optimal sampling frequency to estimate the actual volatility, Andersen et al. (1999) propose to plot the average of daily realized volatility estimates against different sampling frequencies, a so-called 'volatility signature plot', to visualize at what frequency the volatility estimates start to suffer from higher sampling error. For liquid assets, Andersen et al. (1999) consider the frequency with the lowest volatility estimates as proxy for the actual volatility. A volatility signature plot is also used in this paper to determine the optimal frequency of the returns to construct the estimates of the actual volatility with. These estimates serve as a proxy for the actual latent volatility and are in the rest of this thesis referred to as "true volatility". They are used as the target values for the proposed models to forecast volatility. In this paper the volatility forecasts are constructed using datasets with different sampling frequencies. Then they are compared to the true volatility to evaluate the forecasting performance. The goal is to evaluate which data frequency is optimal to forecast the true volatility and hence the actual latent volatility.

Previous papers like Andersen et al. (1999) suggest using a 5-20 minute sampling frequency to estimate and forecast volatility. When plotting the average daily estimates of realize volatility against sampling frequency, this is the resulting optimal frequency for the volatility estimates in their papers. However, no research has been done to evaluate whether the same frequency is also optimal for forecasting volatility. When return data is available at a 1 second time interval, this would mean the 300-1200 available data points are aggregated into 1 observation. While aggregating the data makes the observations less noisy and reduces the computational costs of training models and making forecasts, from a data science point of view it seems suboptimal to discard so much data when it is readily available and might contain valuable information. Therefore knowing the effect of sampling frequency on forecasting performance can be relevant when working with high frequency data.

Many papers have been written about constructing accurate volatility estimates, but almost none have focused on the effects of sampling frequency on the forecasting performance of volatility. This paper looks to contribute to the existing literature by researching this effect. In several papers the optimal frequency for estimation is used to make the forecasts. However, there is no literature to support that this is also the optimal frequency for forecasting. This thesis looks to fill this gap in the literature by comparing forecasting results of a benchmark model, like the Generalized Autoregressive Conditional Heteroskedastic (GARCH) model as proposed by Bollerslev (1986), with that of a Long Short-Term Memory (LSTM) neural network as developed by Hochreiter and Schmidhuber (1997) for different frequencies. To construct the forecasts many different models can be used. Recently methods like machine learning and deep learning gained popularity in forecasting financial time series. There are many papers finding strong evidence that machine learning models outperform traditional models in forecasting volatility (Rahimikia and Poon (2020), Y. Liu (2019), Christensen et al. (2021)). As noted by Rahimikia and Poon (2020), recurrent neural networks, like the LSTM neural network are among the most frequently used machine learning models in both the financial industry and academia when working with high frequency data. The LSTM neural network is designed to capture long-term dependencies in long sequence data and can achieve good forecasting performance on big raw data (Y. Liu, 2019). Since volatility exhibits long memory properties, the LSTM neural network will be used to construct volatility forecasts.

All of the above brings us to the following research question:

"Does increasing sampling frequency also increase out-of-sample volatility forecasting performance when using an LSTM neural network?"

As benchmark model the Generalized Autoregressive Conditional Heteroskedastic (GARCH) model as proposed by Bollerslev (1986), the GJR-GARCH model by Glosten et al., 1993, the threshhold-GARCH (TGARCH) model by Zakoian, 1994 and the exponential-GARCH (EGARCH) model by Nelson, 1991 are considered. The forecasts of the optimal benchmark model are compared to the forecasts of the LSTM models using different frequencies of data. The LSTM neural network is expected to be able to learn more from the extra information in higher frequency data than lose from the extra noise and therefore also able to have a better forecasting performance when increasing the sampling frequency. Being able to more accurately forecast volatility is important to better assess risk in financial markets and therefore interesting for practical applications like assessing market risk for portfolio managers and trading companies or Value at Risk and expected shortfall for banks. Also regulators can benefit from models that produce more accurate volatility forecasts by using it to assess and create regulation to protect investors and the financial market as a whole.

The results show that increasing sampling frequency does increase forecasting performance until the bias in the returns becomes too large. A volatility signature plot seems to be an appropriate tool to determine the optimal sampling frequency to forecast volatility.

The rest of this paper is structured in the follow way. First a review on the relevant literature is given in section 2. Then, a description of the dataset follows in section 3. Next, an overview of the used methodology is given in section 4. The results are discussed in section 5 and in section 7 the conclusion of this research is given.

2 Literature review

This section discusses the relevant literature on microstruture noise and volatility forecasting. First the preliminary literature on the manifestation of microstructure noise in returns is reviewed. Next, a discussion about the volatility forecasting literature is given.

2.1 Preliminaries microstructure noise

Since the famous papers by Engle (1982) and Bollerslev (1986), a broad literature on modeling estimating and forecasting volatility has been developed. To model the characteristics of volatility many models have been proposed. With the large amounts of high frequency transaction and quote data available of different assets, modeling prices in a continuous time setting seems natural (Engle, 2000). Assume the logarithmic prices of an asset $p(t)$ follow a continuous time diffusion process like:

$$dp(t) = \mu(t)dt + \sigma(t)dW(t), \tag{2}$$

where $\mu(t)$ is the drift, $\sigma(t)$ is the spot volatility and $W(t)$ is a standard Brownian motion. Also $\sigma(t) > 0$ and $\sigma(t)$ and $\mu(t)$ are assumed to be independent of the standard Brownian motion $W(t)$. Allowing the spot volatility to be time varying, serially dependent and random, this model implies returns that exhibit some important stylised facts like a fat-tailed unconditional distribution and volatility clustering making it useful in econometrics and finance (O. E. Barndorff-Nielsen & Shephard, 2002). Barndorff-Nielsen and Shephard (2002) and Andersen et al. (2003) showed that the returns $r_t = p(t) - p(t-1)$ are Gaussian conditional on the information set at time t:

$$r_t|\mu(t), \sigma(t) \sim N\left(\mu(t), \int_0^t \sigma^2(s)ds\right), \tag{3}$$

where $\int_0^t \sigma^2(s)ds$ is called the integrated variance and is a measure of the actual latent volatility (McAleer & Medeiros, 2008). Hence it is the object of interest. Following the theory of quadratic variation, Andersen et al. (2003) showed that the realized variance is a consistent estimator of the integrated variance in the absence of microstructure noise, such that

$$\sigma_t^2 \xrightarrow{p} \int_0^t \sigma^2(s)ds, \tag{4}$$

where σ_t^2 is the square of the realized volatility as defined in 1. Andersen and Bollerslev (1998) noted that, in theory, when frequency of observations goes to infinity, the realized volatility measure converges to the measurement of the actual latent volatility. This is why the realized volatility is a good proxy for the actual latent volatility. In practice, however, market microstructure noise makes the convergence infeasible. Following the notation in Zhang et al., 2005, observing the logarithmic prices with noise gives:

$$p_t = p_t^* + \epsilon_t, \tag{5}$$

where p_t^* is considered the true and not contaminated price process and ϵ_t is the microstructure noise. This results in observed return process

$$r_t = r_t^* + \epsilon_t - \epsilon_{t-1} = r_t^* + \nu_t, \tag{6}$$

where again $r_t^* = p_t^* - p_{t-1}^*$ is the true and not contaminated return (McAleer & Medeiros, 2008). The observed returns are autocorrelated making the realized variance and hence the realized volatility a biased estimator of actual latent volatility. Following notation in McAleer and Medeiros (2008),

$$\sigma_t^2 = \sum_{i=1}^{n_t}(r_i^*)^2 + 2\sum_{i=1}^{n_t} r_i^* \nu_i + \sum_{i=1}^{n_t} \nu_i^2, \tag{7}$$

where n_t is the amount of observations in time period t and it follows that

$$\mathbf{E}(\sigma_t^2|r^*) = \sigma_t^{2^*} + 2n_t\mathbf{E}(\epsilon_t^2), \tag{8}$$

such that the realized variance is a biased estimator of the integrated variance. Bandi and Russell (2003) showed that when assuming the microstructure noise has mean zero and is covariance stationary the realized variance and realized volatility estimates converge to infinity when the sampling frequency goes to infinity. Bai et al. (2000) explain this is caused by phenomenon like bid-ask bounce, irregular trading or discreteness of prices. This problem is also likely to manifest itself when forecasting volatility using high frequency data. Andersen et al. (1999) therefore propose to use a higher sampling frequency of the returns instead of using every tick. This way they try to minimize the bias in the volatility estimates while still making optimal use of the convergence properties of the realized variance measure from equation 4. They create volatility signature plots to find the optimal data frequency to construct realized volatility estimates that are closest to the actual latent volatility. Using this technique they find that 5 or 15 minute observations are optimal. They also make use of this result in their other papers (Andersen et al., 2001 Andersen et al., 2004 Andersen et al., 2007). This result is also used by Bollerslev et al. (2006), Andersen et al. (2005), Bollerslev and Wright (2001), Bandi and Russell (2003) and many more. It is clear there is noise in the observed returns and that this can cause difficulties when estimating the realized volatility. However, the effect of the noise when forecasting realized volatility has not been thoroughly researched yet. This thesis evaluates the effect of the sampling frequency on the volatility forecasts and hence whether the optimal frequencies for estimation are also optimal when forecasting the volatility.

2.2 Volatility forecasting

Since the development of the GARCH model it has been extensively research. This has resulted in many extensions and changes to the original GARCH model to try and improve the forecasting power. Hansen and Lunde (2005) compared 330 different GARCH-like models. They find that there is no evidence that any model significantly beats the standard GARCH(1,1) model in their analysis on exchange rate data. However, when using stock data they find that the GARCH(1,1) model is clearly inferior to models that accommodate an asymmetric component. H.-C. Liu and Hung (2010) also find that models allowing asymmetry have a significantly better forecasting performance. Therefore the standard GARCH model and the GJR-GARCH, TGARCH and EGARCH model will be used to determine the best benchmark model.

The development of machine learning models started a new branch of literature in forecasting volatility. Machine learning models can accurately approximate complex and non-linear functions, making them suitable for forecasting realized volatility. Rahimikia and Poon (2020) use an LSTM neural network to forecast realized volatility. They find strong evidence LSTM neural networks dominate HAR and GARCH type models in forecasting power during normal volatility days. Bucci (2020) found that recurrent neural networks like the LSTM neural network are able to outperform all traditional econometric models in forecasting volatility. Y. Liu (2019) compared an LSTM neural network with a GARCH(1,1) model. They find evidence that the LSTM neural network significantly outperforms the GARCH(1,1) model. Y. Liu (2019) notes that LSTM neural networks can learn from big raw data to achieve good predictions for long sequence data. LSTM neural networks are designed to not suffer from the vanishing gradient problem making it capable of remembering long term dependencies in data (Yu et al., 2019). Because of the long memory properties of volatility, the LSTM neural network is a suitable model to forecast realized volatility and will therefore be used to construct the volatility forecasts.

3 Data

3.1 Data manipulation

In this research data is used from Wharton Research Data Services from the NYSE trades and quotes database. It consists of millisecond quote data of the SPDR S&P 500 ETF (SPY) for the period January 2014 until December 2017. Each row of data contains the date, time, bid price and ask price. Only data is used during market open from 09:30 am until 16:00 pm. This dataset contains a total of 6.571.942.712 rows of millisecond quote data. To be able to use the data for this research, the dataset is first cleaned following the procedure used by O. E. Barndorff-Nielsen et al. (2009). First of all, all data entries outside of the 09:30-16:00 time interval are deleted because this research focuses on volatility during market open. Using the holiday calendar of the New York Stock Exchange, markets closed at 1 pm on the following days: 3 July 2014, 28 Nov 2014, 24 Dec 2014, 27 Nov 2015, 24 Dec 2015, 25 Nov 2016, 3 July 2017, 24 Nov 2017. These days do not have the data for the whole day and are excluded from the dataset. Including them would result in much less observations for those days and this could result in problems when optimizing the LSTM neural network due to the missing values in the data matrix that is put into the neural network or it could bias the optimization when all missing values would be set to zero. Next, all entries with a bid or ask price equal to zero are deleted. This step removes errors in the dataset like misrecordings of prices. Then, all quotes with a negative spread, the bid price is larger than the ask price, are deleted because these quotes can not be correct. And last, all duplicate data entries with exactly the same timestamp, bid price and ask price are deleted. The resulting dataset is sampled by so-called tick time. Tick time sampling means that the dataset contains each tick, or quote, containing a bid price and ask price, during the period of interest. This dataset is irregularly spaced because the quotes do not come in at regular time intervals. Because this research uses different sampling frequencies, the dataset is changed to contain data with calendar time sampling. This sampling scheme does contain regularly spaced data which is useful when aggregating the dataset to different frequencies. First, for each unique second all bid prices and ask prices are aggregated by replacing them by their median value. Then for each second the mid price is calculated as the average between the bid price and ask price. The resulting dataset contains the mid prices per second for the whole sample period. Last, the natural logarithm of the returns is calculated for each second using the following formula:

$$r_t = log(P_t/P_{t-1}), \tag{9}$$

where p_t and p_{t-1} are the mid prices for the given time stamps and r_t is the log return.

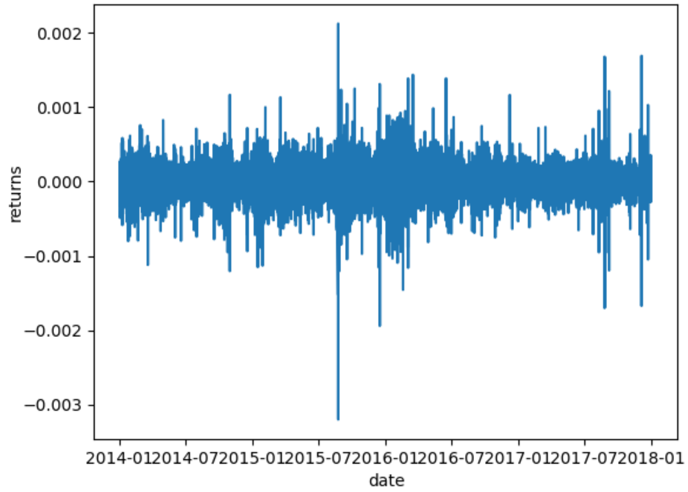

Figure 1: Plot of log returns per second

In Figure 1 the log returns are visualized. One thing that stands out is the large positive and negative observation in August 2015. On the 24th of August in 2015 a flash crash occurred making the S&P500 index lose more than 5% in just minutes. A flash crash is an event where an extreme and rapid drop occurs in the prices of financial assets. Usually the sudden losses are again recovered within the day as is nothing has happened. The 2015 crash also had an impact on the SPY ETF causing a big spike in volatility that day. A crash like this is unpredictable and has a significant impact on the market. It is considered abnormal market activity and distorts the data on small time intervals. The 26 largest absolute returns in the dataset all occurred during this day. Looking at Figure 2 it is clear this crash caused a major volatility spike much bigger than all other days in the sample period. Because the data is distorted this day is excluded from the dataset. The resulting dataset contains 998 different trading days consisting of 23.227.518 second with return data.

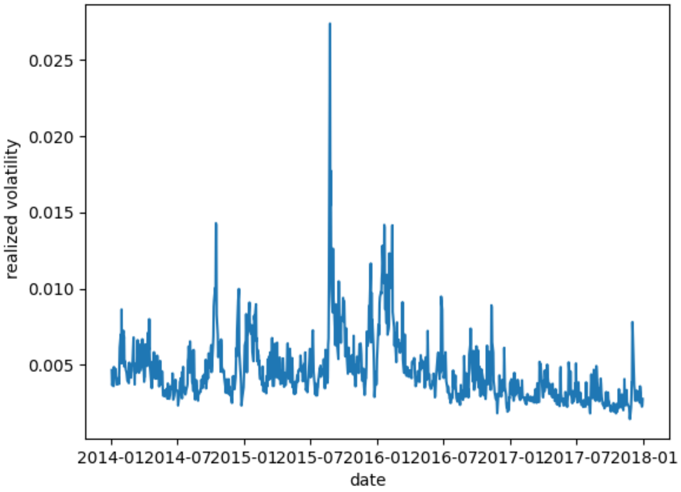

Figure 2: Plot realized volatility per day

3.2 Stylized facts

Asset returns are known to have no significant autocorrelation and are therefore hard to predict. However, the squared returns and the absolute returns usually do have a slow decaying autocorrelation indicating the existence of volatility clustering (Cont, 2001). This means that periods with high volatility alternate with periods with low volatility. This phenomenon is why volatility is more predictable than asset returns. To check for this stylized facts in the data used in this research, an autocorrelation plot is made for the returns, the absolute returns and the squared returns.

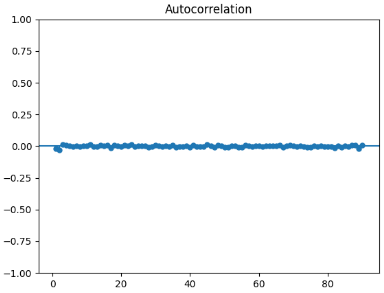

Figure 3: ACF returns

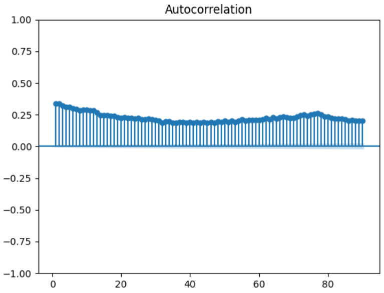

Figure 4: ACF absolute returns

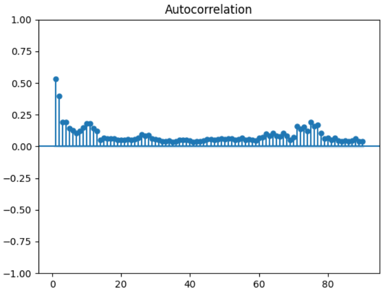

Figure 5: ACF squared returns

Figure 3 shows no significant autocorrelation in the returns while Figure 4 and Figure 5 show that the absolute returns and the squared returns do have a slowly decaying autocorrelation as expected. In the absolute returns the autocorrelation is most prominently visible. This might indicate that the TGARCH model which uses the absolute returns to model the volatility might be the best model to use as a benchmark. Another stylized fact of asset returns is that they are non-normal (Cont, 2001). Returns usually exhibit negative skewness, indicating that large negative returns occur more often than large positive returns. They also have excess kurtosis, meaning that the distribution of returns has a higher peak and fatter tail.

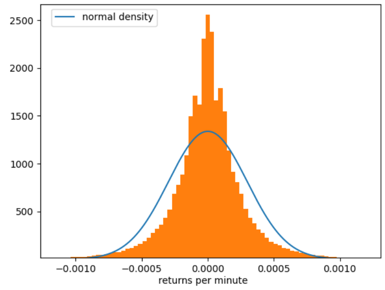

Figure 6: Histogram of returns

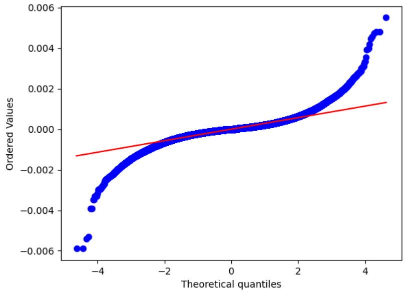

Figure 7: QQ-plot of returns

Figure 6 shows a histogram of the log returns where the blue line indicates the normal distribution. It is clear that the returns do in fact have a higher peaked distribution and fatter tails indicating excess kurtosis. Figure 7 shows a Quantile-Quantile plot where the red line represents a normal distribution. If the returns where normally distributed, the line and data would be aligned, but it is clear the quantiles deviate from the normal quantiles again indication the distribution is not normal. Table 1 show some descriptive statistics of the data. A normal distribution has a skewness of 0 and kurtosis of 3 while the data shows negative skewness and excess kurtosis. The Jarque-Bera test tests whether these conditions hold true for the dataset. The Jarque-Bera test statistic and p-value clearly reject the null hypotheses meaning the returns are not normally distributed.

Table 1: Descriptive statistics

Mean	Std. dev.	Kurtosis	Skewness	Jarque-Bera statistic	p-value
6.770 $(\times 10^{-9})$	3.293 $(\times 10^{-5})$	22.827	-0.051	5.065 $(\times 10^{8})$	0.000

3.3 Volatility signature plot

Using a good volatility proxy to evaluate the performance of the models is important to draw correct conclusions from the results. Following the paper by Andersen et al. (1999) a volatility signature plot is constructed, visualizing the average daily realized volatility for different sampling frequencies. In their paper they consider daily realized volatility as the square of equation 1, σ_t^2. It is important to note that this paper uses formula 1 as the definition for realized volatility and considers the definition used by Andersen et al. (1999) the realized variance. To construct the volatility signature plot, the data is first aggregated to the following frequencies: [1s, 2s, 3s, 4s, 5s, 6s, 10s, 12s, 15s, 20s, 30s, 1m, 2m, 3m, 5m, 6m, 10m, 15m, 20m, 30m]. Next, the average daily realized variance is calculated and plotted against the corresponding sampling frequency.

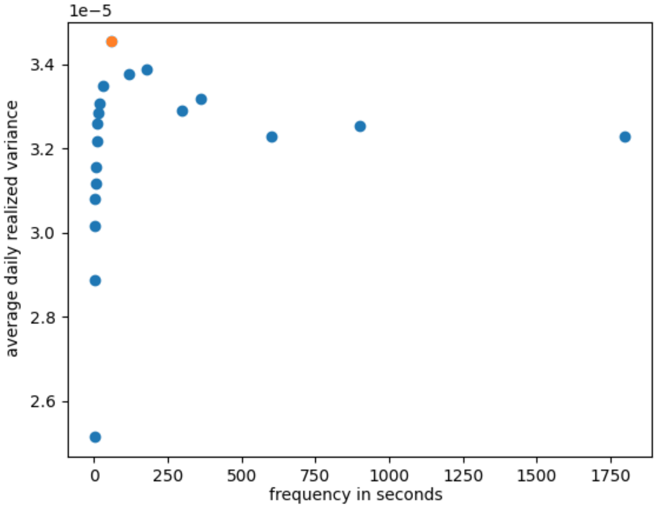

Figure 8: Volatility signature plot

Figure 8 is a visualization of the average daily realized variance and the corresponding sampling frequency in seconds. Andersen et al. (1999) find in their paper that for a liquid asset the largest realized variance estimates occur at the highest sampling frequency. They explain this by negative correlation in the returns that is likely induced by the bid-ask bounce. However, Andersen and Teräsvirta (2009) say that the opposite occurs when using returns constructed from bid-ask quote mid prices because the asymmetric adjustments to the spread make the returns positively correlated. This paper uses returns calculated as the mid price of bid-ask quotes and the pattern observed in Figure 8 is therefore expected. The optimal proxy for the real latent volatility is the frequency where the realized variance starts to suffer from the biases induced by microstructure noise. In this case, at the 60 second frequency indicated by the orange dot in Figure 8, a clear decline in average daily realized variance is visible and this frequency will be used to construct the true volatility, the proxy of the actual latent volatility. The same frequency is found to be optimal by Wilhelmsson

(2006) using high frequency S&P 500 Future data.

4 Methodology

4.1 Volatility models

4.1.1 GARCH

The Generalized Autoregressive Conditional Heteroskedastic (GARCH) process as introduced by Bollerslev (1986) is widely used in the academic literature and by practitioners to model volatility. It assumes that the returns r_t are generated by a model like this:

$$r_t = \mu_t + \sigma_t \epsilon_t, \qquad \Longleftrightarrow \qquad \epsilon_t = (r_t - \mu_t)/\sigma_t, \tag{10}$$

where the ϵ_t's are independently and identically distributed and σ_t is considered the time-varying volatility. The μ_t in this model can be any type of mean process or just a constant. The GARCH(1,1) model reads as follows:

$$\begin{aligned} \sigma_{t+1}^2 &= \omega + \alpha(r_t - \mu)^2 + \beta\sigma_t^2, \\ &= \omega + \alpha\sigma_t^2\epsilon_t^2 + \beta\sigma_t^2, \end{aligned} \tag{11}$$

where the second line follows from 11. The parameters ω, α and β are non-negative to ensure $\sigma_t^2 \geq 0$ for all t with additional constraint $\alpha + \beta < 1$ (Bollerslev, 1986). The model assumes future volatility is predictable conditional on past volatility and past errors. A shortcoming of the GARCH(1,1) model, as discussed by Hansen and Lunde (2005), is the inability to capture the asymmetric properties of return series because η_t and r_t are symmetric time series. To evaluate if models including an asymmetric component better fit the data, three of the most widely used models are considered as a benchmark model.

4.1.2 GJR-GARCH

The GJR-GARCH model is an extension of the traditional GARCH model proposed by Glosten et al. (1993). The mean equation is equal to equation 10, but to the conditional variance equation they added an additional component that is able to capture the asymmetric properties of the time series. The equation is as follows:

$$\sigma_{t+1}^2 = \omega + \alpha(r_t - \mu_t)^2 + \beta\sigma_t^2 + \gamma(r_t - \mu_t)^2 \mathbb{1}_{[r_t - \mu_t < 0]}, \tag{12}$$

where $\mathbb{1}$ is an indicator function that has a value of 1 when the condition $r_t - \mu_t < 0$ holds true and a value of 0 otherwise. If the coefficient γ is zero then this model is again equal to the GARCH model. A positive γ indicates that negative errors have a bigger impact on the conditional volatility than positive errors.

4.1.3 TGARCH

Inspired by the paper by Glosten et al. (1993), Zakoian (1994) developed the threshold-GARCH model (TGARCH). It is very similar to the GJR-GARCH model, again has the same mean equation, but uses the absolute errors instead of the squared errors to model the conditional volatility. The specification is as follows:

$$\sigma_{t+1} = \omega + \alpha|r_t - \mu_t| + \beta\sigma_t + \gamma|r_t - \mu_t| \mathbb{1}_{[r_t - \mu_t < 0]}, \tag{13}$$

where $|r_t - \mu_t|$ is the absolute value of $r_t - \mu_t$ and the asymmetric effect is again captured in the coefficient γ.

4.1.4 EGARCH

The last model used in this paper to capture the asymmetric properties of the return series is the exponential-GARCH model (EGARCH) proposed by Nelson (1991). Again the same mean equation is used but the conditional volatility equation is changed to the following:

$$ln(\sigma_{t+1}^2) = \omega + \alpha(|\epsilon_t| - \sqrt{\frac{2}{\pi}}) + \beta ln(\sigma_t^2) + \gamma\epsilon_t, \tag{14}$$

where ϵ_t is equal to the specification in equation 10 and ln is the natural logarithm. The natural logarithm on the left side of equation 14 makes sure the variance σ_{t+1}^2 can not be negative because the exponential function can only be positive. Therefor there are no restrictions necessary on the parameters of the model. The γ coefficient captures the effect of the asymmetry in the returns.

4.1.5 Long-Short Term Memory neural network

The Long Short-Term Memory (LSTM) neural network as introduced by Hochreiter and Schmidhuber (1997) is a recurrent neural network specifically designed to overcome the vanishing gradient problem. This problem arises in many recurrent neural networks when learning long-term dependencies (Van Houdt et al., 2020). The LSTM neural network consists of one or more memory blocks that are able to retain information over time.

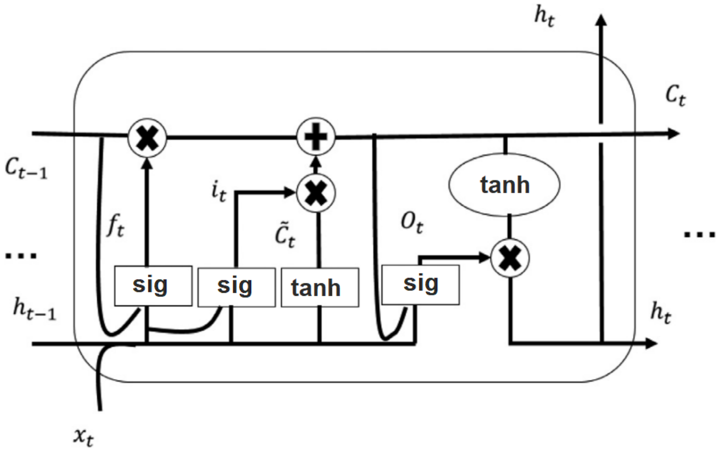

Figure 9: Schematic representation of an LSTM memory block (Y. Liu, 2019)

Figure 9 is a visualization of the LSTM memory block. The main parts of the memory block are the forget gate f_t, the input gate i_t, the output gate O_t and the candidate cell state $\tilde{C}_t$. The sig in Figure 9 is the logistic sigmoid function defined as,

$$sig(x) = \frac{1}{1 + e^{-x}}, \tag{15}$$

and is used as an activation function for the forget gate f_t and the input gate i_t. It returns a value between 0 and 1, The hyperbolic tangent, tanh, is used as activation function to the candidate cell state $\tilde{C}_t$ and the output. It is formulated as follows:

$$tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}, \tag{16}$$

and returns a value between -1 and 1. The logistic sigmoid and hyperbolic tangent are used to enhance the non-linearity of the model making it able to learn complex dependencies in the data.

The candidate cell state $\tilde{C}_t$ is used to update the cell state C_t and uses the current input x_t and the output of previous LSTM unit h_{t-1}. This is done by the following formula:

$$\tilde{C}_t = tanh(W_c x_t + R_c h_{t-1} + b_c), \tag{17}$$

where W_c is the weight vector associated with input x_t and R_c is the weight vector associated with the previous output h_{t-1}. The vector b_c is the bias vector. The input gate i_t is updated using the input x_t, the previous output h_{t-1} and the previous cell state C_{t-1}. It is calculated as follows:

$$i_t = sig(W_i x_t + R_i h_{t-1} + p_i \odot C_{t-1} + b_i), \tag{18}$$

where W_i, R_i and p_i are the weight vectors for x_t, h_{t-1} and C_{t-1} respectively, b_i is the bias vector and $\odot$ is the point-wise multiplication between the two vectors.

The forget gate is used to determine what information should be discarded from the precious cell state C_{t-1}. It uses the input x_t, the previous output h_{t-1} and the previous cell state C_{t-1} in the follow way:

$$f_t = sig(W_f x_t + R_f h_{t-1} + p_f \odot C_{t-1} + b_f), \tag{19}$$

where W_f, R_f and p_f are the weight vectors for x_t, h_{t-1} and C_{t-1} respectively, b_f is the bias vector associated with the forget gate and $\odot$ is the point-wise multiplication between the two vectors.

To compute the cell state C_t, the information computed in the input gate i_t, the forget gate f_t and the candidate cell state $\tilde{C_t}$ as well as the precious cell state C_{t-1} are combined. This is done by the following calculations:

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C_t}. \tag{20}$$

Next the output gate o_t is calculated by combining the current output x_t, previous output y_{t-1} and the previous cell state C_{t-1} in the following way:

$$o_t = sig(W_o x_t + R_o h_{t-1} + p_o \odot C_{t-1} + b_o), \tag{21}$$

where again W_o, R_o and p_o are the weight vectors for x_t, h_{t-1} and C_{t-1} respectively, b_o the bias vector and $\odot$ is the point-wise multiplication. Last, the output h_t is calculated using the current cell state C_t and the output gate o_t as follows:

$$h_t = tanh(C_t) \odot o_t. \tag{22}$$

The neural network is trained using the backpropagation through time algorithm. It feeds the input sequence to the LSTM neural network and computes the output and cell state for each time step. Next, the outputs are compared to the target values and the error is computed as the difference between them. Then, the gradients of the error with respect to the network weights and biases are computed recursively from the last time step back to the first time step. The gradients are used to update the weights and biases in the neural network using an optimization algorithm that minimizes the error. All these steps are then repeated until the network converges or for a predefined number of iterations.

A neural network makes use of hyperparameters that guide the learning process during training. One of them is the amount of layers used in the network. The more layers are used, the better the neural network is able to model complex sequences. However, using more layers makes the model prone to overfitting, resulting in a worse out-of-sample performance. Another hyperparameter is the amount of nodes used in a layer. More nodes can make the model better with the data but can also lead to overfitting. One way to prevent overfitting the model, is to add dropout layers. These layers randomly ignore a preset percentage of the outputs when training. This helps the neural network converge to a more general model which empirically results in better out-of-sample forecasts (Cheng et al., 2017). Also the choice of optimization algorithm, the learning rate and loss function when minimizing the errors of the weights and biases are hyperparamters that can influence the learning of the neural network. A popular algorithm is Adam, a stochastic gradient decent method based on the adaptive estimation of the first and second moments. This algorithm tries to minimize the loss function, usually the mean squared error is used. The Adam optimizer uses by default a learning rate of 0.001 that gets adjusted with each training iteration for faster convergence. The amount of epochs is the hyperparameter that decides on the number of iterations when training the model. In each epoch the whole dataset is used to optimize the network weights and biases. The batch size refers to the number of training observations used in each optimization iteration. Lower batch size can lead to more a stable optimization but also slower convergence. When optimizing over all hyperparameters it is common practice to use k-fold cross-validation to prevent overfitting and support generalization of the model (Berrar, 2019). This method divides the train dataset in k dis-joined subsets of approximately equal size. Then k-1 of these subsets are used to train the model and the remaining subset is used to evaluate its performance using a loss function. This is done until all k subsets have been used to evaluate the model. The average loss of all those models is used to determine the best set of hyperparameters to use for the neural network.

4.2 Evaluation methods

4.2.1 Diebold-Mariano Test

To evaluate which model has a better predictive accuracy a Diebold-Mariano (DM) test is used as proposed by Diebold and Mariano (2002). It compares two forecasts using a loss function and tests whether these two forecasts are significantly different from each other. Patton (2011) shows that from the most commonly used loss functions in volatility forecasting, only the Quasi-Likelihood (QLIKE) and the Mean Squared Prediction Error (MSPE) loss functions are robust to noise in the volatility proxy. Therefor these are used to evaluate the forecasts. The QLIKE loss function is specified similar to Bollerslev et al. (2016):

$$L(\sigma_t^2, \hat{\sigma}_t^2) = \frac{1}{n}\sum_{t=1}^{n}\left(\frac{\sigma_t^2}{\hat{\sigma}_t^2} - log\left(\frac{\sigma_t^2}{\hat{\sigma}_t^2}\right) - 1\right), \tag{23}$$

where σ_t is volatility proxy and $\hat{\sigma_t}$ is the forecasted volatility. The MSPE is specified in the following way:

$$L(\sigma_t^2, \hat{\sigma}_t^2) = \frac{1}{n}\sum_{t=1}^{n}\left(\sigma_t^2 - \hat{\sigma}_t^2\right)^2, \tag{24}$$

where again σ_t is volatility proxy and $\hat{\sigma_t}$ is the forecasted volatility. For two given models, the loss functions are used to calculate the error for each model. Then the loss differential is defined as:

$$d_t = L(\sigma_{1t}^2, \hat{\sigma}_{1t}^2) - L(\sigma_{2t}^2, \hat{\sigma}_{2t}^2), \tag{25}$$

where $L(\sigma_{1t}^2, \hat{\sigma}_{1t}^2)$ and $L(\sigma_{2t}^2, \hat{\sigma}_{2t}^2)$ are the losses for model 1 and 2 respectively. The DM test, tests the null hypothesis of equal predictive performance give by:

$$\mathbb{E}[d_t] = 0. \tag{26}$$

The DM test only requires that the loss differential is covariance stationary. Under this assumption the limiting distribution of the DM test statistic is given by:

$$DM = \frac{\bar{d}}{\hat{\sigma}_{\bar{d}}} \longrightarrow \mathbb{N}(0,1), \tag{27}$$

where $\bar{d}$ is the sample average of the loss differential and $\hat{\sigma}_{\bar{d}}$ a consistent estimate of the standard deviation of the loss differential. Diebold and Mariano (2002) empirically find good results for the DM test but in small samples the test statistic can be oversized. Harvey et al. (1997) therefor propose to adjust the DM test statistic in the following way:

$$DM^* = DM\sqrt{\frac{T + 1 - 2h + T^{-1}h(h-1)}{T}}, \tag{28}$$

where DM is the original statistic from equation 27, T is the amount of forecasts and h represents how many steps ahead the forecasts are made. This statistic is used to evaluate the different models pairwise.

4.2.2 Mincer-Zarnowich regression

The Mincer-Zarnowich regression as proposed by Mincer and Zarnowitz (1969) is used to evaluate the forecasting performance of the individual models. The proxy of the true latent volatility is regressed on a constant and the forecasts:

$$\sigma_t = \beta_0 + \beta_1 \hat{\sigma_t} + \epsilon_t, \tag{29}$$

where $\hat{\sigma_t}$ are the forecasts of the model that is evaluated. The regression is estimated with OLS and uses Newey-West standard errors to correct for autocorrelation and heteroscedasticity. Next a Wald-test is conducted test the joint hypothesis $\beta_0 = 0$ and $\beta_1 = 1$. If the null hypothesis can not be rejected, the forecasts are unbiased.

4.3 Implementation

To be able to test the models on their out-of-sample forecasting performance, the dataset is split in a training dataset, used to train and optimize the models, and a test dataset to evaluate the accuracy of the forecasts out-of-sample. The train dataset consists of the data from the years 2014, 2015 and 2016 while the test dataset consists of the datapoints from 2017. For optimization purposes, all returns are multiplied by a factor 100. This makes it easier for the models to converge.

To determine the most appropriate benchmark model, the train dataset at the daily frequency is used to optimized the models. The GARCH model, TGARCH model, GJR-GARCH model and EGARCH model are all trained on the train data for different settings. For the mean equation, the constant mean and zero mean are considered. For the error distribution, the normal distribution, Student-t distribution, skewed Student-t distribution and Generalized error distribution (GED) are evaluated. Based on the log likelihood of the models, the Akaike information criteria (AIC), the Bayesian information criteria (BIC) and the significance of the parameters of the models a decision is made which model is used as the benchmark model. This model is then used to make the out-of-sample predictions. This is done by training the model using a rolling window and making a one-step ahead forecasts on each iteration. These forecasts are evaluated against the test dataset using the loss functions in equation 23 and 24.

The LSTM neural network is also trained on the training dataset but for different frequencies. The following frequencies are use: [10s, 15s, 20s, 30s, 1m, 2m, 3m, 5m, 6m, 10m, 15m, 30m]. Before training the models the data is transformed using the MinMaxScaler to scale the data between 0 and 1. This can help the model in computing the gradients more efficiently during training which helps converging faster. To optimize the LSTM neural network, many hyperparameters have to be fine tuned. Table 2 shows the different values that are tested for each hyperparameter.

Table 2: LSTM hyperparameters

Hyperparameters	Values
Layers	[1]
Units	[5,10,20,30,50,75,100]
Epochs	[5,10,20,30,50,100,200]
Optimizer	[Adam]
Batch size	[1,2,4,8,16,32,64,128]
Learning rate	[0.00025,0.00050.0,00100]

Only 1 LSTM layer is used in optimizing the LSTM neural networks. For the models using lower frequency data the extra layers didn't perform better than using only a single layer and for the higher frequencies optimizing the model would become extremely computationally expensive. The amount of units, amount of epochs, optimizer and batch sized follow the analysis from Rodikov and Antulov-Fantulin (2022). The LSTM neural networks are trained using the different sets of hyperparameters to determine the optimal model to construct the forecasts. Using 5-fold cross-validation the optimal set of hyperparameters is the set that achieves the lowest average MSPE while training. To prevent overfitting the models to the dataset, a dropout layer is added. This layer randomly drops out 20% of the units in the LSTM layer when training the neural network. The 20% is chosen based on the paper of Zou and Qu, 2020. After optimizing the neural networks for the different data frequencies the forecasts are evaluted against the forecasts of the benchmark model using the Diebold-Mariano test and the Mincer-Zarnowich regression.

5 Results

5.1 Benchmark model

To determine the best benchmark model all models are trained on the training dataset for the different distributions and mean functions. The results for all different models can be found in the Appendix. The first thing to notice is that all models have the worst fit to the data when using a normal distribution compared to the other distributions. The logLikelihood (LogL), AIC and BIC are all much higher for the normal distribution indicating the other distributions better fit the data. This results is for all models the same and does not change for the different mean functions and is in line with the finding from Wilhelmsson (2006). This might be because the returns are also not normally distributed as noted in section 3. The Student's T distribution and the GED have a very similar fit to the data for all models. Again the logL, AIC and BIC are similar when using a zero mean function or a constant mean function. The best fit is achieved when using the skewed Student's T distribution. This is a result of the excess kurtosis and negative skewness in the returns as show in section 3 (Alberg et al., 2008). The results for all different models using the skewed Student's T distribution are shown in table 3. The parameter λ controls the skewness of the distribution. All models report a significant negative skewness at the 1% level. The parameter η controls the tail shape of the distribution similar to the degrees of freedom in the Student's T distribution. An η of 1 would indicate similar tails as the normal distribution. The results show for all models a higher η significant at the 1% level indicating the distribution of the errors has bigger tails. When conducting a Ljung-Box test on the standardized residuals of the models, all models show no significant autocorrelation in the standardized residuals and squared standardized residuals up to at least lag 100. This indicates that the models do capture the autocorrelation in the data.

Table 3: Results for each model using the Skewed Student's T distribution

	GARCH		GJR-GARCH		TGARCH		EGARCH	
	Zero	Constant	Zero	Constant	Zero	Constant	Zero	Constant
μ		0.020		0.012		0.010***		0.011
ω	0.012*	0.012*	0.018**	0.017**	0.035**	0.034**	-0.069*	-0.072*
α	0.164***	0.166***	0.068**	0.069**	0.053**	0.055**	0.272***	0.271***
γ			0.223**	0.215**	0.204***	0.199***	-0.161***	-0.156***
β	0.820***	0.814***	0.787***	0.786***	0.827***	0.827***	0.931***	0.933***
η	4.837***	4.924***	5.245***	5.287***	5.619***	5.666***	5.538***	5.584***
λ	-0.164***	-0.143***	-0.160***	-0.148***	-0.161***	-0.151***	-0.161***	-0.150***
logL	738.993	738.002	732.053	731.708	723.761	**723.530**	725.543	725.253
AIC	1487.99	1488.00	1476.11	1477.42	**1459.52**	1461.06	1463.09	1464.51
BIC	1512.56	1517.49	1505.59	1511.81	**1489.01**	1495.46	1492.57	1498.9

Note: the confidence level is indicated by * for 90% level, ** for 95% level and *** for 99% level.

Looking at table 3 the GARCH model has the highest values for the logL, AIC, and BIC. This indicates that the other models have a better fit to the data. This is also in line with the findings from Hansen and Lunde (2005). Models incorporating an asymmetric component better fit the return data than the standard GARCH(1,1) model. The logL, AIC and BIC do not change much when adjusting the mean function for all models. This result is also reported by Hansen and Lunde (2005). Out of the models that incorporate an asymmetric component the GJR-GARCH model has a lesser fit to the data compared to the TGARCH and EGARCH model based on the logL, AIC, and BIC. The TGARCH reports the lowest values for the logL, AIC and BIC but they do not differ much from the values for the EGARCH model. Therefore the TGARCH and EGARCH with zero mean and constant mean are used to construct forecasts to determine which model has the best out-of-sample forecasting performance and will be used as benchmark model.

Table 4: Out-of-sample results TGARCH and EGARCH models

	TGARCH		EGARCH	
	Zero	Constant	Zero	Constant
MSPE	0.0236	**0.0234**	0.0254	0.0251
QLIKE	0.2711	**0.2695**	0.2791	0.2773
MZ p-value	0.0000	0.0000	0.0000	0.0000

In Table 4 the out-of-sample results for the TGARCH and EGRACH model with zero and constant mean are reported. TGARCH with constant mean has the lowest MSPE and lowest QLIKE value. Although the other models report higher values for the loss functions, they do not differ much. Interestingly the p-values from Minzer-Zarnowitz regression indicate that the forecasts for all models are significantly biased. Figure 10 shows the true volatility in blue and the forecasted volatility by the TGARCH model with constant mean in orange. The forecasts seem to have an upward bias compared to the true volatility and the model did not accurately capture the large spike at the end of the forecasting period. The other three models display similar characteristics in the plots of their forecasts. This explains why the p-value of the Mincer-Zarnowitz regression is 0.0000 for all models.

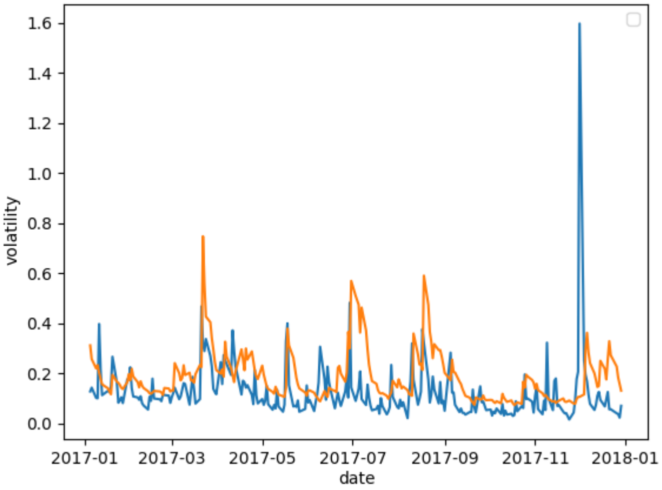

Figure 10: Plot of true volatility (blue) against TGARCH with constant mean forecasts (orange)

To determine which model has the best forecasting performance the Diebold-Mariano test is used. The results for this test are reported in Table 5. The models on the left side of the table are used as model 1 and the models on top are used as model 2 in the test as stated in section 4.2.1. For the MSPE loss function the DM test does indicate a significant difference in forecasting performance between all the models at a significance level of 5%. From the definition of the DM statistic in equation 27 the statistic is negative when the average loss differential is negative and positive when the average loss differential is positive. If the average loss differential is negative, then the average loss for model 1 is smaller than the average loss for model 2. Following the interpretation from Liang et al. (2020), this indicates that a significant and negative DM statistic would indicate that model 1 has a significantly better out-of-sample forecasting performance compared to model 2. Looking at Table 5 the TGARCH model with constant mean has a significant negative DM statistic when comparing with the other models for the MSPE loss function. This means that this model has superior forecasting power over the other models. When looking at the DM statistics for the QLIKE loss function only the TGARCH with constant mean and EGARCH with zero mean have a significant difference in forecasting performance. One reason for this might be the poor performance to predict the big volatility spike at the end of the forecasting period. Since the QLIKE loss function penalizes lower forecasts more heavily than higher forecasts compared to the true value, the bad prediction of the volatility spike can have a big effect on the loss function and the results of the DM test. Because the TGARCH-C model did perform significantly better than the other models for the MSPE, this model is chosen as benchmark.

Table 5: Diebold-Mariano test results

MSPE	TGARCH-Z	TGARCH-C	EGARCH-Z	EGARCH-C
TGARCH-Z		5.933***	-2.383**	-2.100**
TGARCH-C	-5.933***		-2.599***	-2.337**
EGARCH-Z	2.383**	2.599***		5.333***
EGARCH-C	2.100**	2.337**	-5.333***	
QLIKE	**TGARCH-Z**	**TGARCH-C**	**EGARCH-Z**	**EGARCH-C**
TGARCH-Z		1.932*	-1.880*	-1.242
TGARCH-C	-1.932*		-2.589**	-1.792*
EGARCH-Z	1.880*	2.589**		1.962*
EGARCH-C	1.242	1.792*	-1.962*	

Note: the confidence level is indicated by * for 90% level, ** for 95% level and *** for 99% level. The models on the left side are used as model 1 and on the top as model 2 when conduction the DM test from section 4.2.1. The Z and C stand for the zero mean and constant mean models respectively.

5.2 Long-Short Term Memory neural network

For each different frequency of data an LSTM neural network is optimized and trained to make volatility forecasts. They are optimized for the different hyperparameters as noted in Table 2 using 5-fold cross-validation to prevent overfitting the models and allow for more generalized results. The models are optimized with respect to the MSPE loss function from equation 24 where for each data frequency the model with the lowest average loss is considered the best model. This model with the related hyperparemeters is used to make the volatility forecasts and the results for all frequencies are displayed in Table 6.

Table 6: LSTM neural network results

Frequency	Units	Batch	Epochs	learn rate	MSPE	QLIKE	MZ p-value
30m	30	64	100	0.0005	0.0036	0.0774	0.0000
15m	30	32	100	0.0005	0.0026	0.0440	0.0633
10m	30	64	100	0.0005	0.0021	0.0446	0.0000
6m	30	32	100	0.00025	0.0024	0.0232	0.0000
5m	30	32	100	0.00025	0.0023	0.0153	0.1325
3m	50	64	200	0.00025	0.0021	0.0112	0.1848
2m	50	64	200	0.00025	0.0010	0.0175	0.0000
1m	75	64	100	0.00025	0.0005	0.0138	0.0850
30s	75	64	200	0.00025	0.0030	0.0780	0.0000
20s	75	64	200	0.00025	0.0082	0.0716	0.0000

The first thing to note from Table 6 is the increasing amount of units used by the models when the data frequency increases. Higher frequency data has increasingly more data to train the models with. A probable explanation for the increasing amount of nodes is the increasing amount of data used in training the models. Using more units in a model makes it able to get a bitter fit to the data. For the lower frequencies using 30 nodes is enough to reach a good fit to the data while not overfitting the model. Increasing the amount of data makes it harder to get a good fit and learn the complexities and therefor the amount of units used increases. The batch size stays relatively constant only using batches of 32 or 64. The learning rate seems to decrease when data frequency increases. This might be because the higher frequency data contains smaller return values and updating the model parameters using a smaller learning rate can make the model converge more easily and prevent overshooting when updating the gradients. The amount of epochs on the other hand seems to increase when frequency increases. This can be a result of the learning rate decreasing and thus needing more epochs to reach convergence when training the neural network.

The forecasts of the benchmark model had a MSPE of 0.0234. The MSPE values reported in Table 6 are all considerably lower than that. The same holds true for the QLIKE loss. One of the reasons the MSPE for the benchmark model is much higher is because of inaccurately forecasting the volatility spike in the end of the forecasting period. Looking at Figure 11 the LSTM neural network using the 1 minute data is able to make accurate forecasts of the volatility spike. Also

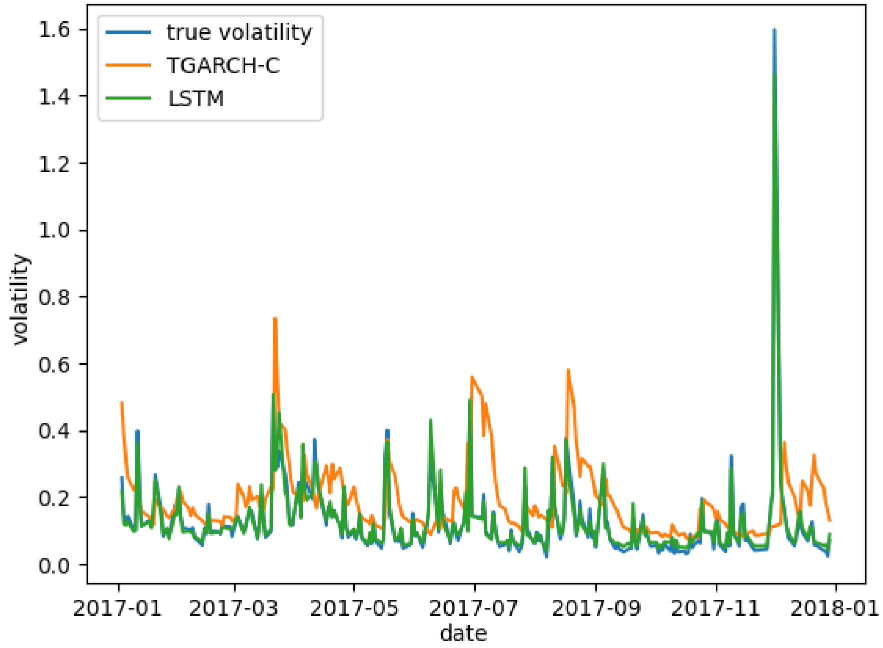

Figure 11: Plot of true volatility (blue) against TGARCH-C (orange) and the LSTM neural network using 1 minute data (green)

during the whole forecasting period it is clearly visible the forecasts of the LSTM model more closely resemble the true volatility compared to the forecasts of the benchmark model. Therefor the MSPE and QLIKE are, as expected, lower for the LSTM neural networks compared to the benchmark model. To test whether the forecasts are unbiased the Mincer-Zarnowitz regression is conducted as explained in section 4.2.2. The p-values in Table 6 indicate that the forecasts of most models are biased. At the 5% significance level only the 15 minute, 5 minute, 3 minute and 1 minute LSTM neural network do have unbiased forecasts. A possible explanation for the biased forecasts is that the LSTM neural network did not manage converge to the true minimum but instead got reached a local minimum. At the 30 second and 20 second frequency the bias could also be induced by the microstructure noise in the data. The MSPE and the QLIKE for these frequencies show a significant increase compared to the 1 minute frequency LSTM model. Overall the MSPE is slowly decreasing when sampling frequency increases indicating that the forecasts are closer to the true value when using an LSTM neural network that is trained with higher frequency data. For the 30 second and 20 second frequency however, the MSPE increases again.

In Figure 12 the MSPE is visualized against the associated data frequency. Interestingly, the

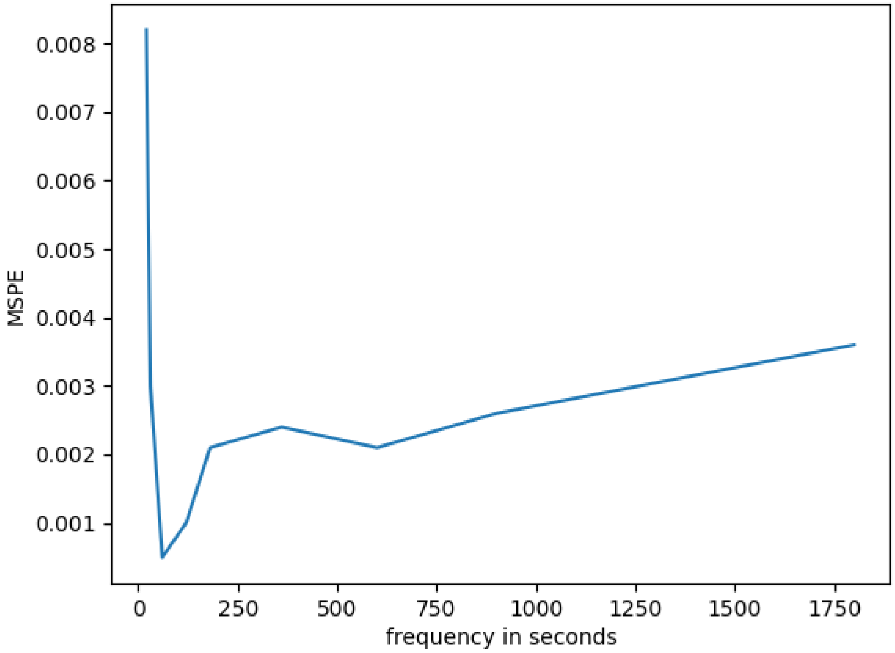

Figure 12: Plot of MSPE values against the different data frequencies

plot is very similar to the volatility signature plot in Figure 8. While the volatility signature plots shows a slight increase in daily average realized variance when sampling frequency increases Figure 12 shows a slight decrease. Both graphs reach their extreme value when using a frequency of 1 minute returns and then the volatility signature plot shows a large decrease while Figure 12 shows a large increase. Andersen et al. (1999) explained that there is a bias variance trade-off in choosing the optimal frequency of returns. Lower frequency returns are less biased but increase the variance of estimating the realized variance while higher frequency returns increasingly suffer from microstructure noise which also causes less accurate estimates of the realized variance. Therefore they consider the lowest frequency before the realized variance estimator clearly starts suffering from the bias in the returns as the optimal frequency. Based on Figure 12 the same result seems to hold when forecasting the realized variance using the LSTM neural network. This supports the use of the volatility signature plot as proposed by Andersen et al. (1999) to choose the optimal frequency of returns when forecasting volatility with an LSTM neural network. Like the MSPE, Table 6 shows a somewhat similar behaviour for the QLIKE measure but it is not as clear. This is probably due to the LSTM neural network being optimized for the MSPE and not the QLIKE.

To formally test whether the LSTM neural networks significantly have superior forecasting power compared to the benchmark model a DM test is conducted. Table 7 shows the DM statistics when comparing the benchmark model against the LSTM model for the different frequencies. At the 5% significance level all LSTM neural networks do significantly outperform the benchmark model when using the MSPE measure aswell as the QLIKE measure. That the LSTM neural networks can outperform GARCH like models is in line with previous research on this topic. The results in Table 7 show that even the models for the 30 second and 20 second frequency have significantly better forecasting power compared to the benchmark model despite the clear manifestation of the microstructure noise bias in the returns data.

Table 7: DM test statistics comparing benchmark vs LSTM for different frequencies

	30m	15m	10m	6m	5m	3m	2m	1m	30s	20s
MSPE	2.390**	2.633***	2.493**	2.773***	2.895***	2.898***	2.545**	2.542**	2.556**	3.953***
QLIKE	4.088**	4.978***	4.998***	5.522***	5.739***	5.830***	5.613***	5.694***	4.259***	4.863***

Note: the confidence level is indicated by * for 90% level, ** for 95% level and *** for 99% level.

To determine what frequency is optimal when forecasting realized variance, the DM test is used to compare the forecasts of all different LSTM models. The DM statistics are displayed in Table 8. Based on the MSPE values in Table 6 the LSTM neural network using the 60 second frequency data was expected to be the best model. However, using the MSPE measure the DM test can not reject the null hypothesis of no difference in forecasting power when comparing it to the results of the LSTM neural network using the 6 minute, 5 minute, 3 minute and 2 minute data. The 2 minute and 60 second frequency do have a significantly better forecasting performance compared to the 30 minute, 15 minute, 10 minute and 30 second frequencies. However, they do not have superior forecasting power compared to the 20 second frequency. Based on the QLIKE from Table 6 the 3 minute frequency seems optimal. This result is confirmed by the DM test results in Table 8. The LSTM neural network using the 3 minute data outperforms all other models at the 99% confidence level except the model using the 60 second data. At the 90% confidence level the 3 minute data also has a significantly better forecasting performance compared to the 60 second data.

Table 8: DM test statistics comparing all LSTM models of different frequencies by MSPE and QLIKE

MSPE	data1800S	data900S	data600S	data360S	data300S	data180S	data120S	data60S	data30S	data20S
data1800S		1.973**	2.972***	1.432	1.256	1.532	3.559***	3.570***	1.102	-1.003
data900S	-1.973**		0.631	0.277	0.398	0.784	1.879*	1.825*	-1.988	-1.338
data600S	-2.972***	-0.631		-0.298	-0.122	0.040	3.124***	3.137**	-1.407	-1.254
data360S	-1.432	-0.277	0.298		0.495	1.468	1.364	1.306	-1.236	-1.492
data300S	-1.256	-0.398	0.122	-0.495		2.085**	1.080	1.024	-1.036	-1.644
data180S	-1.532	-0.784	-0.040	-1.468	-2.085**		0.942	1.003	-1.446	-1.669*
data120S	-3.559***	-1.879*	-3.124***	-1.364	-1.080	-0.942		1.321	-2.392**	-1.460
data60S	-3.570***	-1.825*	-3.137**	-1.306	-1.024	-1.003	-1.321		-2.358**	-1.446
data30S	-1.102	1.988	1.407	1.236	1.036	1.446	2.392**	2.358**		-1.214
data20S	1.003	1.338	1.254	1.492	1.644	1.669*	1.460	1.446	1.214	

QLIKE	data1800S	data900S	data600S	data360S	data300S	data180S	data120S	data60S	data30S	data20S
data1800S		4.188***	3.498***	6.352***	7.16***	7.690***	6.662***	7.190***	-0.059	0.485
data900S	-4.188***		-0.114	4.455***	6.443***	7.529***	5.404***	6.502***	-4.19***	-3.113***
data600S	-3.498***	0.114		7.068***	9.988***	10.812***	11.275***	11.257***	-6.595***	-3.205***
data360S	-6.352***	-4.455***	-7.068***		4.347***	7.001***	2.479**	4.495***	-8.513***	-6.214***
data300S	-7.16***	-6.443***	-9.988***	-4.347***		3.897***	-1.362	1.062	-10.245***	-7.66***
data180S	-7.690***	-7.529***	-10.812***	-7.001***	-3.897***		-3.733***	-1.792*	-10.740***	-8.154***
data120S	-6.662***	-5.404***	-11.275***	-2.479**	1.362	3.733***		3.945***	-12.249***	-6.899***
data60S	-7.190***	-6.5026***	-11.257***	-4.495***	-1.062	1.792*	-3.945***		-11.705***	-7.560***
data30S	0.059	4.194***	6.595***	8.513***	10.245***	10.740***	12.249***	11.705***		0.722
data20S	-0.485	3.113***	3.205***	6.214***	7.664***	8.154***	6.899***	7.560***	-0.722	

Note: the confidence level is indicated by * for 90% level, ** for 95% level and *** for 99% level. The models on the left side are used as model 1 and on the top as model 2 when conduction the DM test from section 4.2.1.

6 Robustness Check

The years 2014-2017 were not very volatile years. To check whether the results for 2014-2017 are robust to more volatile periods, the same analysis is conducted to the year 2018-2021. In Figure 13 in Appendix 8.2 the returns per second are displayed. Comparing this Figure to Figure 1 it is clear that the period 2018-2021 is more volatile. This is also confirmed when comparing the daily realized volatility from Figure 14 in Appendix 8.2 to Figure 2. The data is cleaned following the procedure described in section 3 and the following days have been removed because of early closure of the exchange: 4 July 2018, 22 November 2018, 25 December 2018, 4 July 2019, 28 November 2019, 25 December 2019, 26 November 2020, 25 December 2020, 26 November 2021. Based on the volatility signature plot in Figure 15 in Appendix 8.2 the 2 minute frequency is used as proxy for the actual latent volatility. This is higher then the 1 minute frequency for the less volatile period. This might be because the microstructure noise has a bigger impact on the estimation of volatility in a more volatile period.

All tables and figures with the results of the analysis can be found in Appendix 8.2. The results show that using the skewed Student's T distribution results in the best fit for all benchmark models. The GARCH and GJR-GARCH model have the worst fit to the data and the TGARCH model has the best fit. In the out-of-sample analysis the TGARCH model with constant mean reports the lowest MSPE while the QLIKE is almost the same for all models. Based on the Diebold-Mariano test results from table 19 the TGARCH-C model outperforms the other models based on the MSPE measure. For the QLIKE measure the null hypothesis of equal performance can not be rejected. Based on these results the TGARCH-C model is used as a benchmark. These results are very similar to the analysis of the period 2014-2017.

The results for optimizing the LSTM neural networks for the different frequencies are reported in Table 20. The hyperparameters show strong similarities with the hyperparameters from the period 2014-2017. This is also the case for the MSPE values for the different frequencies. Figure 17 shows a plot of the MSPE values against the sampling frequency. This plot looks like the mirrored volatility signature plot as was also the case in the 2014-2017 period. This means that the lowest MSPE value is again found at the optimal frequency derived from the volatility signature plot. This indicates that the use of the volatility signature plot as proposed by Andersen et al., 1999 is not only appropriate to determine the frequency to estimate the volatility, but it is also appropriate to determine the sampling frequency to use for forecasting volatility in volatile and less volatile periods.

Table 21 shows that the benchmark model is outperformed by the LSTM model at all different frequencies based on the DM test. When comparing the different LSTM models against each other, the DM test results in Table 22 indicates that the 2 minute frequency outperforms all other models for the MSPE measure and for the QLIKE measure while in the less volatile period 2014-2017 the DM test could not reject the null hypothesis of equal performance between the 6 minute, 5 minute, 3 minute, 2 minute and 1 minute frequency. In the period 2018-2021 the 20 second frequency is outperformed by all models except the 30 minute frequency, the 30 second frequency only outperforms the 20 second model and the 1 minute frequency only outperforms the 30 second and 20 second frequency. For this period the 3 minute frequency is only outperformed by the 2 minute frequency and the 5 minute frequency is only outperformed by the 2 minute and 3 minute frequency. Overall, it seems that in the more volatile period 2018-2021 the increased bias at higher frequencies and increased variability at lower frequencies have a bigger effect on the forecasting performance than in the less volatile period 2014-2017.

7 Conclusion

The aim of this thesis is to investigate the effects of sampling frequency on the forecasting performance with an LSTM neural network. The analysis has show that increasing sampling frequency does increase the forecasting performance until the bias induced by microstructure noise becomes too large. The frequency at which the microstructure noise becomes problematic for forecasting is similar to the frequency where the bias in estimating volatility becomes a problem. The volatility signature plot as introduced by Andersen et al., 1999 is an appropriate method to evaluate what sampling frequency should be used to forecast volatility. In the less volatile period 2014-2017 a sampling frequency between 6 minutes and 1 minute resulted in equal forecasting performance. However, the more volatile period 2018-2021 showed that the 2 minute frequency outperforms all other frequencies. Both periods confirm that the optimal frequency resulting from the volatility signature plot is appropriate to use when forecasting volatility.

Previous literature on estimating volatility concluded that frequencies between 20 minutes and 5 minutes are optimal based on the volatility signature plot. However, in this thesis the optimal estimates are the 1 minute frequency for the less volatile period 2014-2017 and the 2 minute frequency for the more volatile period 2018-2021. This difference can be a result of the use of different assets.

It could therefore be interesting to repeat this analysis for different assets. The existing literature is mainly focused on estimation volatility but the effects of sampling frequency on volatility forecasting have not been thoroughly researched. This thesis contributes to the existing literature by showing that increasing sampling frequency can increase the forecasting performance. Because of the importance of volatility in the financial world this can also be interesting for portfolio managers, risk managers, trading firms or regulators. The results support the volatility signature plot as a tool to determine what frequency to use for forecasting volatility, indication another way to use this plot next to finding the optimal estimation frequency.

As mentioned before, to validate the robustness of the results, the research can be extended to different assets or a portfolio of assets. For future research, it is also suggested to repeat the analysis using transaction data instead of quote data. This thesis used the mid prices from quote data to construct the returns. Using transaction data will result in a different volatility signature plot and this might influence the results of the analysis. A limitation of the thesis is the lack of computing power to train the LSTM neural networks. Having more computing power available makes it possible to use even larger datasets in training the models as well as using more layers. This makes is possible to train more complex neural networks that can potentially learn more from the complexities in the high frequency data.

References

Alberg, D., Shalit, H., & Yosef, R. (2008). Estimating stock market volatility using asymmetric garch models. *Applied Financial Economics*, *18*(15), 1201–1208.

Andersen, T. G., & Bollerslev, T. (1998). Answering the skeptics: Yes, standard volatility models do provide accurate forecasts. *International economic review*, 885–905.

Andersen, T. G., Bollerslev, T., & Diebold, F. X. (2007). Roughing it up: Including jump components in the measurement, modeling, and forecasting of return volatility. *The review of economics and statistics*, *89*(4), 701–720.

Andersen, T. G., Bollerslev, T., Diebold, F. X., & Ebens, H. (2001). The distribution of realized stock return volatility. *Journal of financial economics*, *61*(1), 43–76.

Andersen, T. G., Bollerslev, T., Diebold, F. X., & Labys, P. (1999). (understanding, optimizing, using and forecasting) realized volatility and correlation.

Andersen, T. G., Bollerslev, T., Diebold, F. X., & Labys, P. (2003). Modeling and forecasting realized volatility. *Econometrica*, *71*(2), 579–625.

Andersen, T. G., Bollerslev, T., & Meddahi, N. (2004). Analytical evaluation of volatility forecasts. *International Economic Review*, *45*(4), 1079–1110.

Andersen, T. G., Bollerslev, T., & Meddahi, N. (2005). Correcting the errors: Volatility forecast evaluation using high-frequency data and realized volatilities. *Econometrica*, *73*(1), 279–296.

Andersen, T. G., & Teräsvirta, T. (2009). Realized volatility. *Handbook of financial time series* (pp. 555–575). Springer.

Bai, X., Russell, J., & Tiao, G. (2000). Beyond merton's utopia: Effects of non-normality and dependence on the precision of variance estimates using high-frequency financial data. *Unpublished. Graduate school of Business, University of Chicago, Chicago.*

Bandi, F. M., & Russell, J. R. (2003). Microstructure noise, realized volatility, and optimal sampling. *Unpublished paper, Graduate School of Business, University of Chicago.*

Bandi, F. M., & Russell, J. R. (2007). Volatility. *Handbooks in Operations Research and Management Science*, *15*, 183–222.

Barndorff-Nielsen, O. E., Hansen, P. R., Lunde, A., & Shephard, N. (2009). Realized kernels in practice: Trades and quotes.

Barndorff-Nielsen, O. E., & Shephard, N. (2002). Econometric analysis of realized volatility and its use in estimating stochastic volatility models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *64*(2), 253–280.

Barndorff-Nielsen, & Shephard. (2002). Econometric analysis of realized volatility and its use in estimating stochastic volatility models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *64*(3), 253–280.

Berrar, D. (2019). Cross-validation.

Bollerslev, T. (1986). Generalized autoregressive conditional heteroskedasticity. *Journal of econometrics*, *31*(3), 307–327.

Bollerslev, T., Litvinova, J., & Tauchen, G. (2006). Leverage and volatility feedback effects in high-frequency data. *Journal of Financial Econometrics*, *4*(3), 353–384.

Bollerslev, T., Patton, A. J., & Quaedvlieg, R. (2016). Exploiting the errors: A simple approach for improved volatility forecasting. *Journal of Econometrics*, *192*(1), 1–18.

Bollerslev, T., & Wright, J. H. (2001). High-frequency data, frequency domain inference, and volatility forecasting. *Review of Economics and Statistics*, *83*(4), 596–602.

Bucci, A. et al. (2017). Forecasting realized volatility: A review. *Journal of Advanced Studies in Finance (JASF)*, *8*(16), 94–138.

Bucci, A. (2020). Realized volatility forecasting with neural networks. *Journal of Financial Econometrics*, *18*(3), 502–531.

Cheng, G., Peddinti, V., Povey, D., Manohar, V., Khudanpur, S., & Yan, Y. (2017). An exploration of dropout with lstms. *Interspeech*, 1586–1590.

Christensen, K., Siggaard, M., & Veliyev, B. (2021). A machine learning approach to volatility forecasting. *Available at SSRN*.

Cont, R. (2001). Empirical properties of asset returns: Stylized facts and statistical issues. *Quantitative finance*, *1*(2), 223.

Diebold, F. X., & Mariano, R. S. (2002). Comparing predictive accuracy. *Journal of Business & economic statistics*, *20*(1), 134–144.

Engle, R. F. (1982). Autoregressive conditional heteroscedasticity with estimates of the variance of united kingdom inflation. *Econometrica: Journal of the econometric society*, 987–1007.

Engle, R. F. (2000). The econometrics of ultra-high-frequency data. *Econometrica*, *68*(1), 1–22.

Glosten, L. R., Jagannathan, R., & Runkle, D. E. (1993). On the relation between the expected value and the volatility of the nominal excess return on stocks. *The journal of finance*, *48*(5), 1779–1801.

González-Rivera, G., Lee, T.-H., & Mishra, S. (2004). Forecasting volatility: A reality check based on option pricing, utility function, value-at-risk, and predictive likelihood. *International Journal of forecasting*, *20*(4), 629–645.

Hansen, P. R., & Lunde, A. (2005). A forecast comparison of volatility models: Does anything beat a garch (1, 1)? *Journal of applied econometrics*, *20*(7), 873–889.

Harvey, D., Leybourne, S., & Newbold, P. (1997). Testing the equality of prediction mean squared errors. *International Journal of forecasting*, *13*(2), 281–291.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9*(8), 1735–1780.

Liang, C., Wei, Y., & Zhang, Y. (2020). Is implied volatility more informative for forecasting realized volatility: An international perspective. *Journal of Forecasting*, *39*(8), 1253–1276.

Liu, H.-C., & Hung, J.-C. (2010). Forecasting s&p-100 stock index volatility: The role of volatility asymmetry and distributional assumption in garch models. *Expert Systems with Applications*, *37*(7), 4928–4934.

Liu, Y. (2019). Novel volatility forecasting using deep learning–long short term memory recurrent neural networks. *Expert Systems with Applications*, *132*, 99–109.

McAleer, M., & Medeiros, M. C. (2008). Realized volatility: A review. *Econometric reviews*, *27*(1-3), 10–45.

Mincer, J. A., & Zarnowitz, V. (1969). The evaluation of economic forecasts. *Economic forecasts and expectations: Analysis of forecasting behavior and performance* (pp. 3–46). NBER.

Nelson, D. B. (1991). Conditional heteroskedasticity in asset returns: A new approach. *Econometrica: Journal of the econometric society*, 347–370.

Patton, A. J. (2011). Volatility forecast comparison using imperfect volatility proxies. *Journal of Econometrics*, *160*(1), 246–256.

Poon, S.-H., & Granger, C. W. J. (2003). Forecasting volatility in financial markets: A review. *Journal of economic literature*, *41*(2), 478–539.

Rahimikia, E., & Poon, S.-H. (2020). Machine learning for realised volatility forecasting. *Available at SSRN*, *3707796*.

Rodikov, G., & Antulov-Fantulin, N. (2022). Can lstm outperform volatility-econometric models? *arXiv preprint arXiv:2202.11581*.

Van Houdt, G., Mosquera, C., & Nápoles, G. (2020). A review on the long short-term memory model. *Artificial Intelligence Review*, *53*, 5929–5955.

Wilhelmsson, A. (2006). Garch forecasting performance under different distribution assumptions. *Journal of Forecasting*, *25*(8), 561–578.

Yu, Y., Si, X., Hu, C., & Zhang, J. (2019). A review of recurrent neural networks: Lstm cells and network architectures. *Neural computation*, *31*(7), 1235–1270.

Zakoian, J.-M. (1994). Threshold heteroskedastic models. *Journal of Economic Dynamics and control*, *18*(5), 931–955.

Zhang, L., Mykland, P. A., & Aıt-Sahalia, Y. (2005). A tale of two time scales: Determining integrated volatility with noisy high-frequency data. *Journal of the American Statistical Association*, *100*(472), 1394–1411.

Zou, Z., & Qu, Z. (2020). Using lstm in stock prediction and quantitative trading. *CS230: Deep Learning, Winter*, 1–6.

8 Appendix

8.1 Results benchmark models

Table 9: Results GARCH model

Mean	Zero				Constant			
Dist.	Normal	T	skew T	GED	Normal	T	skew T	GED
μ					0.030**	0.040***	0.020	0.034*
ω	0.0174*	0.010	0.012*	0.013*	0.018*	0.012	0.012*	0.014*
α	0.161***	0.148**	0.164***	0.151***	0.165***	0.163**	0.166***	0.160***
β	0.795***	0.838***	0.820***	0.820***	0.791***	0.822***	0.814***	0.809***
ν		4.692***		1.206***		4.547***		1.204***
η			4.837***				4.924***	
λ			-0.164***				-0.143***	
logL	787.172	748.258	738.993	748.438	785.233	743.698	738.002	744.816
AIC	1580.34	1504.52	1487.99	1504.88	1578.47	1497.4	1488	1499.63
BIC	1595.08	1524.17	1512.56	1524.53	1598.12	1521.96	1517.49	1524.2

Note: the confidence level is indicated by * for 90% level, ** for 95% level and *** for 99% level.

Table 10: Results GJR-GARCH model

Mean	Zero				Constant			
Dist.	Normal	T	skew T	GED	Normal	T	skew T	GED
μ					0.016	0.032**	0.012	0.027***
ω	0.022***	0.018**	0.018**	0.019**	0.021***	0.017**	0.017**	0.018**
α	0.023	0.055*	0.068**	0.041	0.022	0.062*	0.069**	0.044
γ	0.273***	0.238**	0.223**	0.250***	0.267***	0.219**	0.215**	0.234***
β	0.783***	0.791***	0.787***	0.787***	0.783***	0.788**	0.786***	0.786***
ν		5.163***		1.259***		4.942***		1.250***
η			5.245***				5.287***	
λ			-0.160***				-0.148***	
logL	769.293	740.678	732.053	739.308	768.733	737.859	731.708	737.128
AIC	1546.59	1491.36	1476.11	1488.62	1547.47	1487.72	1477.42	1486.26
BIC	1566.24	1515.93	1505.59	1513.18	1572.04	1517.2	1511.81	1515.74

Note: the confidence level is indicated by * for 90% level, ** for 95% level and *** for 99% level.

Table 11: Results TGARCH model

Mean	Zero				Constant			
Dist.	Normal	T	skew T	GED	Normal	T	skew T	GED
μ					0.016***	0.026**	0.010***	0.024***
ω	0.039***	0.034**	0.035**	0.035***	0.037***	0.031**	0.034**	0.033***
α	0.007	0.040*	0.053**	0.026	0.010	0.048*	0.055**	0.031
γ	0.245***	0.212***	0.204***	0.225***	0.240***	0.198***	0.199***	0.213***
β	0.836***	0.835***	0.827***	0.836***	0.836***	0.834***	0.827***	0.836***
ν		5.583***		1.298***		5.379***		1.288***
η			5.619***				5.666***	
λ			-0.161***				-0.151***	
logL	757.054	732.087	723.761	731.323	756.379	729.891	723.53	729.435
AIC	1522.11	1474.17	1459.52	1472.65	1522.76	1471.78	1461.06	1470.87
BIC	1541.76	1498.74	1489.01	1497.22	1547.33	1501.26	1495.46	1500.35

Note: the confidence level is indicated by * for 90% level, ** for 95% level and *** for 99% level.

Table 12: Results EGARCH model

Mean	Zero				Constant			
Dist.	Normal	T	skew T	GED	Normal	T	skew T	GED
μ					0.017	0.028**	0.011	0.025*
ω	-0.088***	-0.068*	-0.069*	-0.078**	-0.090***	-0.071*	-0.072*	-0.081**
α	0.216***	0.253***	0.272***	0.236***	0.217***	0.258***	0.271***	0.237***
γ	-0.198***	-0.168***	-0.161***	-0.180***	-0.194***	-0.155***	-0.156***	-0.170***
β	0.923***	0.934***	0.931***	0.930***	0.926***	0.938***	0.933***	0.934***
ν		5.450***		1.288***		5.249***		1.279***
η			5.538***				5.584***	
λ			-0.161***				-0.150***	
logL	760.113	733.861	725.543	733.239	759.385	731.445	725.253	731.217
AIC	1528.23	1477.72	1463.09	1476.48	1528.77	1474.89	1464.51	1474.43
BIC	1547.88	1502.29	1492.57	1501.05	1553.34	1504.37	1498.9	1503.92

Note: the confidence level is indicated by * for 90% level, ** for 95% level and *** for 99% level.

8.2 Tables and figures robustness check

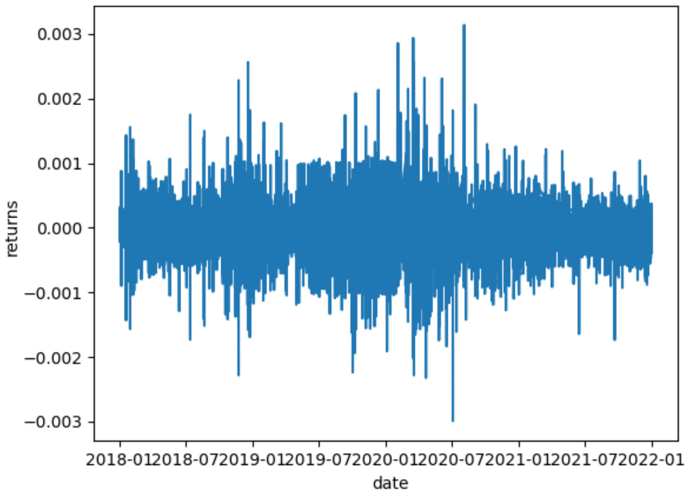

Figure 13: Plot of log returns per second

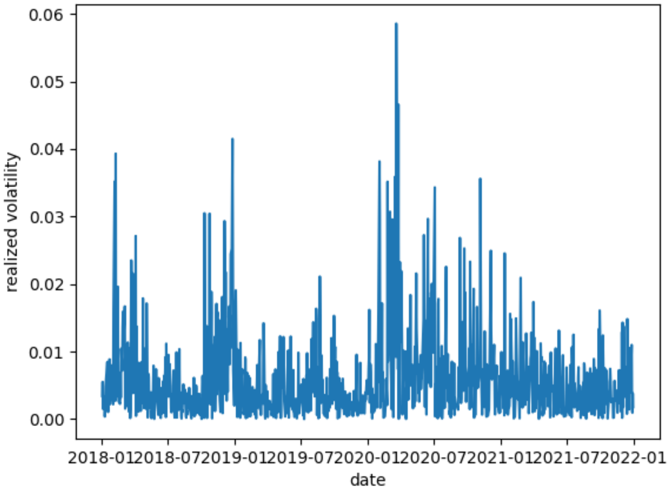

Figure 14: Plot of realized volatility per day

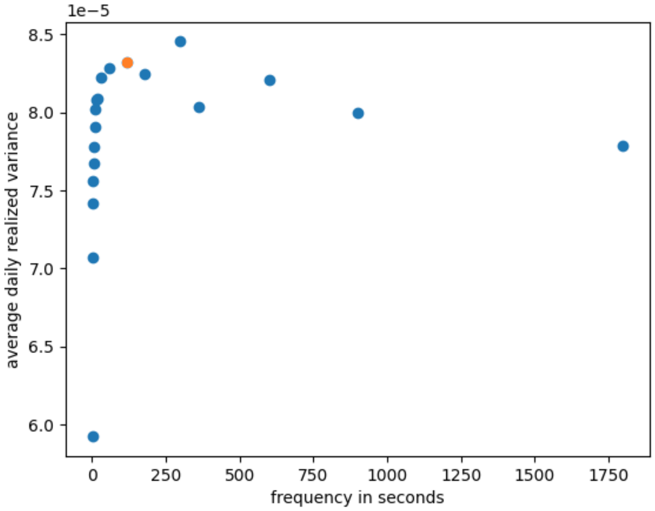

Figure 15: Volatility signature plot

Table 13: Results GARCH model

Mean	Zero				Constant			
	Normal	T	skew T	GED	Normal	T	skew T	GED
μ					0.038*	0.063***	0.031	0.063***
ω	0.025*	0.030**	0.033**	0.026**	0.024*	0.032**	0.031**	0.026**
α	0.166***	0.202***	0.204***	0.175***	0.168***	0.216***	0.2055***	0.181***
β	0.815***	0.789***	0.784***	0.804***	0.815***	0.781***	0.781***	0.799***
ν		4.131***		1.137***		3.899***		1.116***
η			4.108***				4.172***	
λ			-0.173***				-0.148***	
logL	1142.92	1090.69	1079.24	1091.38	1141.24	1084.62	1078.08	1085.83
AIC	2291.84	2189.39	2168.49	2190.76	2290.48	2179.23	2168.15	2181.65
BIC	2306.53	2208.98	2192.97	2210.35	2310.07	2203.72	2197.54	2206.14

Note: the confidence level is indicated by * for 90% level, ** for 95% level and *** for 99% level.

Table 14: Results GJR-GARCH model

Mean	Zero				Constant			
	Normal	T	skew T	GED	Normal	T	skew T	GED
μ					0.021	0.052***	0.018	0.053
ω	0.023**	0.030***	0.034**	0.026**	0.022**	0.029**	0.032**	0.025**
α	0.067**	0.060	0.065	0.059*	0.070**	0.067	0.066	0.063*
γ	0.163***	0.232***	0.236***	0.195***	0.156**	0.216***	0.225***	0.178***
β	0.830***	0.805***	0.797***	0.817***	0.829***	0.803***	0.797***	0.816***
ν		4.237***		1.152***		4.003***		1.129***
η			4.224***				4.257***	
λ			-0.176***				-0.161***	
logL	1134.46	1082.60	1071.12	1084.67	1133.94	1078.41	1070.72	1080.68
AIC	2276.93	2175.20	2154.24	2179.34	2277.88	2168.81	2155.44	2173.35
BIC	2296.52	2199.69	2183.63	2203.83	2302.37	2198.20	2189.72	2202.74

Note: the confidence level is indicated by * for 90% level, ** for 95% level and *** for 99% level.

Table 15: Results TGARCH model

Mean	Zero				Constant			
	Normal	T	skew T	GED	Normal	T	skew T	GED
μ					0.019***	0.047***	0.015	0.048
ω	0.036***	0.040**	0.042***	0.038***	0.035***	0.037***	0.040***	0.036***
α	0.067**	0.061**	0.064**	0.0625**	0.075**	0.073**	0.067**	0.075**
γ	0.173***	0.199***	0.197***	0.185***	0.166***	0.183***	0.191***	0.168***
β	0.844***	0.839***	0.835***	0.841***	0.843***	0.835***	0.835***	0.837***
ν		4.487***		1.177***		4.251***		1.155***
η			4.485***				4.511***	
λ			-0.185***				-0.173***	
logL	1125.56	1077.26	1065.05	1079.76	1125.22	1073.49	1064.75	1076.08
AIC	2259.12	2164.52	2142.10	2169.52	2260.44	2158.98	2143.51	2164.17
BIC	2278.71	2189.01	2171.48	2194	2284.93	2188.37	2177.79	2193.55

Note: the confidence level is indicated by * for 90% level, ** for 95% level and *** for 99% level.

Table 16: Results EGARCH model

Mean	Zero				Constant			
	Normal	T	skew T	GED	Normal	T	skew T	GED
μ					0.019***	0.047***	0.015	0.048
ω	-0.007	-0.004	-0.003	-0.011	-0.010	-0.009	0.040***	-0.017
α	0.266***	0.263***	0.266***	0.260***	0.269***	0.276***	0.067**	0.269***
γ	-0.139***	-0.160***	-0.158***	-0.150***	-0.133***	-0.148***	0.191***	-0.136***
β	0.953***	0.951***	0.949***	0.951***	0.955***	0.954***	0.835***	0.954***
ν		4.443***		1.173***		4.204***		1.151***
η			4.434***				4.511***	
λ			-0.182***				-0.173***	
logL	1126.67	1078.21	1066.36	1080.44	1126.31	1074.45	1064.75	1076.81
AIC	2261.35	2166.41	2144.72	2170.88	2262.61	2160.90	2143.51	2165.61
BIC	2280.94	2190.90	2174.11	2195.37	2287.10	2190.28	2177.79	2195.00

Note: the confidence level is indicated by * for 90% level, ** for 95% level and *** for 99% level.

Table 17: Results for each model using the Skewed Student's T distribution

	GARCH		GJR-GARCH		TGARCH		EGARCH	
	Zero	Constant	Zero	Constant	Zero	Constant	Zero	Constant
μ		0.031		0.018		0.015		0.015
ω	0.033**	0.031**	0.034**	0.032**	0.042***	0.040***	-0.003	-0.008
α	0.204***	0.2055***	0.065	0.066	0.064**	0.067**	0.266***	0.268***
γ			0.236***	0.225***	0.197***	0.191***	-0.158***	-0.153***
β	0.784***	0.781***	0.797***	0.797***	0.835***	0.835***	0.949***	0.951***
η	4.108***	4.172***	4.224***	4.257***	4.485***	4.511***	4.434***	4.464***
λ	-0.173***	-0.148***	-0.176***	-0.161***	-0.185***	-0.173***	-0.182***	-0.170***
logL	1079.24	1078.08	1071.12	1070.72	1065.05	**1064.75**	1066.36	1066.04
AIC	2168.49	2168.15	2154.24	2155.44	**2142.10**	2143.51	2144.72	2146.08
BIC	2192.97	2197.54	2183.63	2189.72	**2171.48**	2177.79	2174.11	2180.37

Note: the confidence level is indicated by * for 90% level, ** for 95% level and *** for 99% level.

Table 18: Out-of-sample results TGARCH and EGARCH models

	TGARCH		EGARCH	
	Zero	Constant	Zero	Constant
MSPE	0.3308	**0.3293**	0.3531	0.3510
QLIKE	0.4784	0.4794	**0.4774**	0.4779
MZ p-value	0.0000	0.0000	0.0000	0.0000

Table 19: Diebold-Mariano test results

MSPE	TGARCH-Z	TGARCH-C	EGARCH-Z	EGARCH-C
TGARCH-Z		3.320***	-1.765*	-1.619
TGARCH-C	-3.320***		-1.874*	-1.732*
EGARCH-Z	1.765*	1.874*		4.097***
EGARCH-C	1.619	1.732*	-4.097***	
QLIKE	**TGARCH-Z**	**TGARCH-C**	**EGARCH-Z**	**EGARCH-C**
TGARCH-Z		-0.543	0.160	0.084
TGARCH-C	-0.543		0.291	0.245
EGARCH-Z	0.160	0.291		-0.281
EGARCH-C	0.084	0.245	0.281	

Note: the confidence level is indicated by * for 90% level, ** for 95% level and *** for 99% level. The models on the left side are used as model 1 and on the top as model 2 when conduction the DM test from section 4.2.1. The Z and C stand for the zero mean and constant mean models respectively.

Table 20: LSTM neural network results

Frequency	Units	Batch	Epochs	learn rate	MSPE	QLIKE	MZ p-value
30m	30	64	100	0.0005	0.0689	0.0694	0.0000
15m	30	64	100	0.0005	0.0465	0.0550	0.2837
10m	30	64	200	0.0005	0.0312	0.0521	0.0000
6m	30	32	100	0.0005	0.0148	0.0573	0.0000
5m	50	64	100	0.00025	0.0156	0.0250	0.0000
3m	50	64	200	0.00025	0.0099	0.0421	0.0000
2m	50	64	200	0.00025	0.0023	0.0095	0.0000
1m	75	64	100	0.00025	0.0291	0.0443	0.0002
30s	75	64	200	0.00025	0.0687	0.1099	0.0000
20s	75	64	200	0.00025	0.0839	0.1841	0.0000

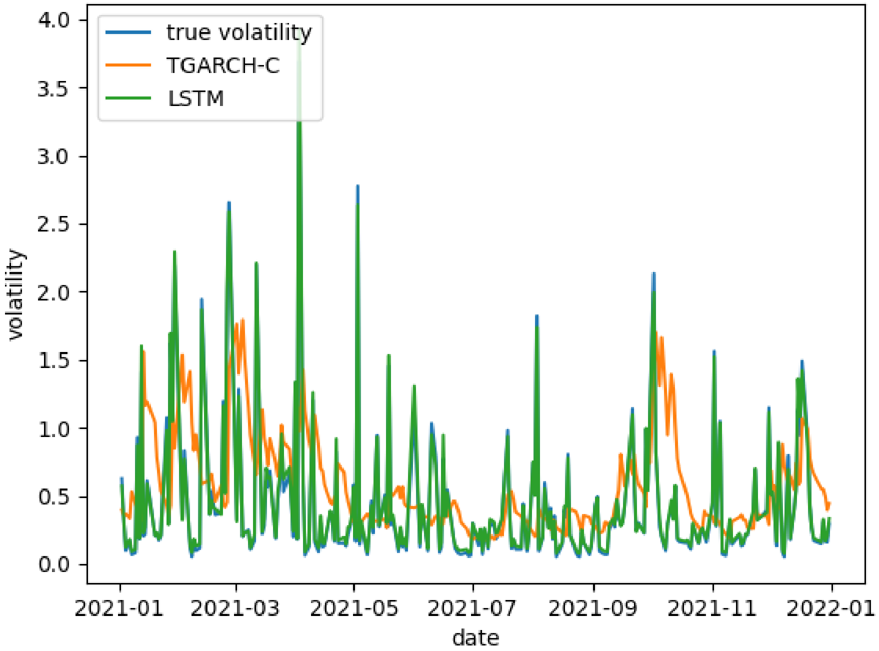

Figure 16: Plot of true volatility against forecasts of TGARHC-C and LSTM neural network using 2 minute data

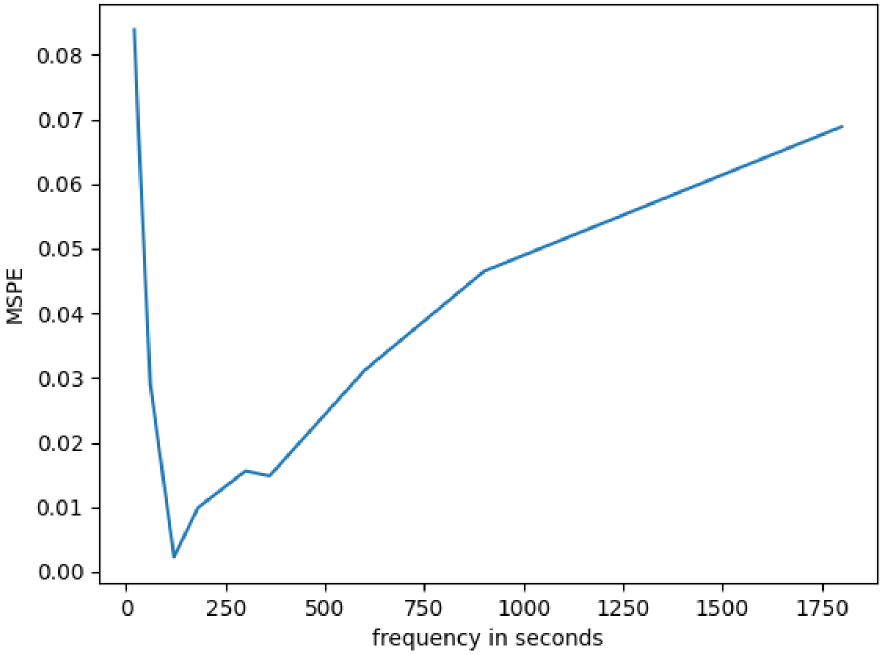

Figure 17: Plot of MSPE values against the different data frequencies in seconds

Table 21: DM test statistics comparing benchmark vs LSTM for different frequencies

	30m	15m	10m	6m	5m
MSPE	6.981***	5.948***	6.653***	6.962***	6.849***
QLIKE	9.582***	10.145***	10.432***	10.381***	10.815***
	3m	2m	1m	30s	20s
MSPE	6.848***	6.869***	6.661***	6.834***	6.754***
QLIKE	10.686***	11.295***	10.560***	9.494***	7.781***

Note: the confidence level is indicated by * for 90% level, ** for 95% level and *** for 99% level.

Table 22: DM test statistics comparing all LSTM models of different frequencies by MSPE and QLIKE

MSPE	data1800S	data900S	data600S	data360S	data300S	data180S	data120S	data60S	data30S	data20S
data1800S		0.705	1.274	1.835*	1.819*	1.991**	2.264**	1.341	0.008	-0.914
data900S	-0.705		1.051	2.215**	2.200**	2.623***	3.106***	1.391	-1.370	-2.292**
data600S	-1.274	-1.051		2.461**	2.142**	2.848***	3.455***	0.322	-3.283***	-4.170***
data360S	-1.835*	-2.215**	-2.461**		-0.350	1.792*	2.695***	-3.662***	-4.482***	-4.916***
data300S	-1.819*	-2.200**	-2.142**	0.350		2.757***	3.874***	-3.407***	-4.100***	-4.592***
data180S	-1.991**	-2.623***	-2.848***	-1.792*	-2.757***		3.320***	-4.420***	-4.267***	-4.690***
data120S	-2.264**	-3.106***	-3.455***	-2.695***	-3.874***	-3.320***		-4.563***	-4.340***	-4.745***
data60S	-1.341	-1.391	-0.322	3.662***	3.407***	4.420***	4.563***		-3.787***	-4.305***
data30S	-0.008	1.370	3.283***	4.482***	4.100***	4.267***	4.340***	3.787***		-4.626***
data20S	0.914	2.292**	4.170***	4.916***	4.592***	4.690***	4.745***	4.305***	4.626***	
QLIKE	data1800S	data900S	data600S	data360S	data300S	data180S	data120S	data60S	data30S	data20S
data1800S		2.313**	2.328**	1.424	6.433***	3.423***	8.459***	3.328***	-3.771***	-8.051***
data900S	-2.313**		0.723	-0.439	6.492***	2.905***	12.012***	2.216**	-6.546***	-10.605***
data600S	-2.328**	-0.723		-2.082**	5.634***	4.261***	13.128***	1.879*	-8.747***	-12.971***
data360S	-1.424	0.439	2.082**		5.568***	8.187***	11.557***	2.778***	-8.652***	-14.155***
data300S	-6.433***	-6.492***	-5.634***	-5.568***		-3.588***	5.726***	-5.344***	-10.176***	-12.188***
data180S	-3.423***	-2.905***	-4.261***	-8.187***	3.588***		11.065***	-0.572	-10.873***	-14.575***
data120S	-8.459***	-12.012***	-13.128***	-11.557***	-5.726***	-11.065***		-11.173***	-13.192***	-14.595***
data60S	-3.328***	-2.216**	-1.879*	-2.778***	5.344***	0.572	11.1732		-11.151***	-12.482***
data30S	-3.771***	6.546***	8.747***	8.652***	10.176***	10.873***	13.192***	11.151***		-11.396***
data20S	8.051***	10.605***	12.971	14.155***	12.188***	14.575***	14.595***	12.482***	11.396***	

Note: the confidence level is indicated by * for 90% level, ** for 95% level and *** for 99% level. The models on the left side are used as model 1 and on the top as model 2 when conduction the DM test from section 4.2.1.