FAU Discussion Papers in Economics

No. 03/2023

Forecasting Realized Volatility in Turbulent Times using Temporal Fusion Transformers

Johannes Frank
FAU Erlangen-Nürnberg

ISSN 1867-6707

Friedrich-Alexander-Universität Erlangen-Nürnberg
Institute for Economics
https://www.iwf.rw.fau.de/discussion-papers/

Forecasting Realized Volatility in Turbulent Times using Temporal Fusion Transformers

Johannes Frank[*]

February 13, 2023

Abstract

This paper analyzes the performance of temporal fusion transformers in forecasting realized volatilities of stocks listed in the S&P 500 in volatile periods by comparing the predictions with those of state-of-the-art machine learning methods as well as GARCH models. The models are trained on weekly and monthly data based on three different feature sets using varying training approaches including pooling methods. I find that temporal fusion transformers show very good results in predicting financial volatility and outperform long short-term memory networks and random forests when using pooling methods. The use of sectoral pooling substantially improves the predictive performance of all machine learning approaches used. The results are robust to different ways of training the models.

Keywords: Realized volatility, temporal fusion transformer, long short-term memory network, random forest

JEL classification: C45, C53, C58, E44

[*] Corresponding author. Friedrich-Alexander-Universität Erlangen-Nürnberg, Lange Gasse 20, 90403 Nuremberg, johannes.jf.frank@fau.de.

I would like to thank Jonas Dovern, Alexander Glas, Lena Müller and Daniel Perico Ortiz for their very useful comments and suggestions which greatly improved my paper.

1 Introduction

In a highly interconnected world, crises of all kinds can strongly and quickly affect financial markets. The Russian invasion of Ukraine in February 2022 and the associated consequences such as uncertainties in energy supply or high inflation rates led to enormous fluctuations on stock markets. Two years earlier, with the declaration of COVID-19 as a global pandemic on 11 March 2020 by the World Health Organization (WHO), great uncertainty about future developments quickly spread to international financial markets. Major stock indices like the S&P 500, the Dow Jones or the NASDAQ fell enormously and some even experienced their worst trading days since 1987. Such events emphasize the importance of being able to measure and assess financial market risks. Financial volatility is one of the most widespread measures of risk that is used in practice and describes the variation in returns of financial market instruments. The application of machine learning (ML) techniques to predict financial volatility has been growing strongly for years (see, for example, Bucci, 2020; Luong and Dokuchaev, 2018). The availability of large and detailed data sets and the increasing processing power of modern computers create excellent conditions for ML algorithms.

This paper studies the performance of temporal fusion transformer (TFT) models in forecasting weekly and monthly realized volatility (RV) of individual stocks listed in the S&P 500 index in times of crisis. The S&P 500 is a large stock market index of 500 leading companies, which covers around 80 percent of equity market capitalization in the United States. The TFT network is trained on the basis of three different sets of features. The first set contains only company-specific data. The second one is extended by data describing the general US equity development. The last set additionally includes a number of macroeconomic variables. I also study the impact of pooling strategies on the results. The models are trained using three different approaches: individually, pooling at sectoral level and pooling across the entire sample. To compare the performance with common approaches, I use random forests (RF) and long short-term memory (LSTM) networks, two state-of-the-art ML methods, as well as econometric GARCH models as a benchmark. The Diebold-Mariano test is used to check the statistical significance of the results (Diebold and Mariano, 1995). I find that the TFT approach is able to significantly outperform all benchmark models given a sufficiently large data set. This is especially true on a weekly data frequency. Pooling across sectors or across all data available in the sample greatly improves the predictive results of all ML methods. In addition, I show how taking into account the returns of large stock or volatility indices as well as macroeconomic variables reduces forecast errors in the context of RV.

The impact of TFT models on financial applications and on forecasting of RV in particular has received little academic attention so far. Lim et al. (2021) show that their TFT approach generates good predictions of daily RV. Hu (2021) finds better performance outcomes of the TFT model for predicting stock prices than support vector regression and LSTM networks. However, none of these studies examine the heterogeneity across sectors, the influence of the temporal frequency of the data, and the potential performance gains from pooling data across sectors or the entire dataset.

The remainder of this paper is organized as follows. Section 2 explains the data and describes the construction of the different sets of features as well as the target generation. Section 3 discusses the ML models and the different methods for the training processes.

The main results are presented in Section 4. Section 5 shows various robustness checks and Section 6 concludes.

2 Data

2.1 Stock market data

The stock market data is collected from Refinitiv, a global provider of financial market data. For each trading day in the period from the 1^{st} of January 2000 to the 30^{st} of June 2022, the data set contains the closing price, trading volume and the economic sector of all S&P 500 constituents listed on the last day of the sample period. The companies are classified according to the North American Industry Classification System (NAICS) consisting of 20 economic sectors, with the sample including companies from only 15 different industries. In addition, the daily closing prices of the S&P 500 index and the Chicago Board Options Exchange Volatility Index (VIX) covering the full sample period of 5660 trading days are taken from Refinitiv. The stock price data are adjusted for dividend payments and splits.

In a first step, I remove companies for which the data are not available for the entire period. This concerns, for example, companies that were founded or went public after January 2000. This leaves 355 companies for which the data are completely available. For each company $n \in \{1, ..., 355\}$, I calculate the continuously compounded return (log return) on day $t, r_{n,t}$. Weekly (monthly) returns are therefore the sum of the daily log returns occurring in a week (month). Table 1 depicts some average monthly descriptive statistics for the stock data.

Table 1: Average monthly summary statistics for the S&P 500 stocks presented by sectors

Sector	Number of stocks	Return	Skewness	Kurtosis
Manufacturing	133	0.63%	-0.54	8.21
Finance & Insurance	45	0.42%	-1.87	31.10
Utilities	26	0.46%	-2.16	27.60
Real Estate & Rental Leasing	25	0.61%	-1.15	18.75
Information	23	0.59%	-0.78	7.19
Retail Trade	20	0.81%	-0.28	5.13
Professional, Scientific & Technical Services	18	0.57%	-0.81	9.85
Mining, Quarrying, Oil & Gas Extraction	17	0.59%	-1.45	20.12
Transportation & Warehousing	13	0.58%	-1.30	11.35
Wholesale Trade	9	1.10%	-0.01	2.77
Accomodation & Food Services	7	0.92%	-0.07	18.58
Administrative, Support, Waste Management & Remediation Services	6	0.90%	-0.90	13.26
Health Care & Social Assistance	6	1.14%	-0.72	7.24
Construction	6	0.97%	-2.41	31.50
Other Services (except Public Administration)	1	0.88%	-0.52	5.11
All	355	0.63%	-0.98	15.08

Notes: This table shows the average monthly descriptive statistics for the reported NAICS sectors as the average moments of the individual associated stocks.

2.2 Macroeconomic data

In addition to the stock market data, I consider the following macroeconomic variables on a daily basis for the entire sample period: US daily news index, effective federal funds rate (EFFR), 30-year treasury constant maturity rate and two spread series regarding the ICE BofA US high yield index as well as the ICE BofA AAA US corporate index. The employment level per sector is only available on a monthly basis.[1]

The US daily news index measures policy-related economic uncertainty for the United States. Baker et al. (2016) have created this index based on the coverage of economic policy uncertainty (EPU) in over 1000 US newspapers. In line with related literature (see, for example, Shaikh, 2019), Baker et al. (2016) show the strong relationship between EPU and financial volatility. The EFFR is a short-term interest rate that is determined daily based on the banks' overnight federal funds transactions. By setting a certain target range, the EFFR thus functions as an important instrument of monetary policy. The 30-year treasury constant maturity rate, on the other hand, represents a long-term interest rate. It is defined as the nominal yield of actively traded treasury securities adjusted to a constant maturity. The values are derived from the daily yield curve by means of interpolation for the respective maturity.

Following the approach of Christiansen et al. (2012), I also include two spread series as additional regressors in the macroeconomic context. The first time series describes the option adjusted spreads of the ICE BofA AAA US corporate index. This index comprises all corporate debt instruments with a given investment grade rating AAA issued in the US domestic market. The spreads are calculated as the difference between these bonds and a spot treasury curve. The second series refers to spreads of the ICE BofA US high yield index. Based on the classification of the three major rating agencies, namely Fitch, Standard & Poors and Moody's, this index covers corporate bonds with a rating of BB or below. Since spreads are interpreted as risk premiums for investors, they are significantly higher for high-yield securities than for those with an investment grade rating. Due to the generally higher risk of corporate bonds with a lower rating, the spreads react much more sensitively in crisis situations.[2]

Since all models are trained on a weekly and monthly basis, the daily data must be transformed accordingly. For this purpose, I calculate the average over the respective time period for both interest rate time series and the EPU. In order to better capture the swings in highly volatile periods, the maximum of the relevant week or month is used for the two spread series and the trading volume. For the weekly employment level, I repeat the respective monthly value.

2.3 Target and feature generation

Andersen et al. (2001) measures daily RV as the square root of the sum of squared intra-daily returns. This approach is also used when determining weekly or monthly RV,

[1] The US daily news index is collected from www.policyuncertainty.com. The four time series regarding interest rates and spreads are taken from FRED. I receive the data on sectoral employment levels from the US Bureau of Labor Statistics.

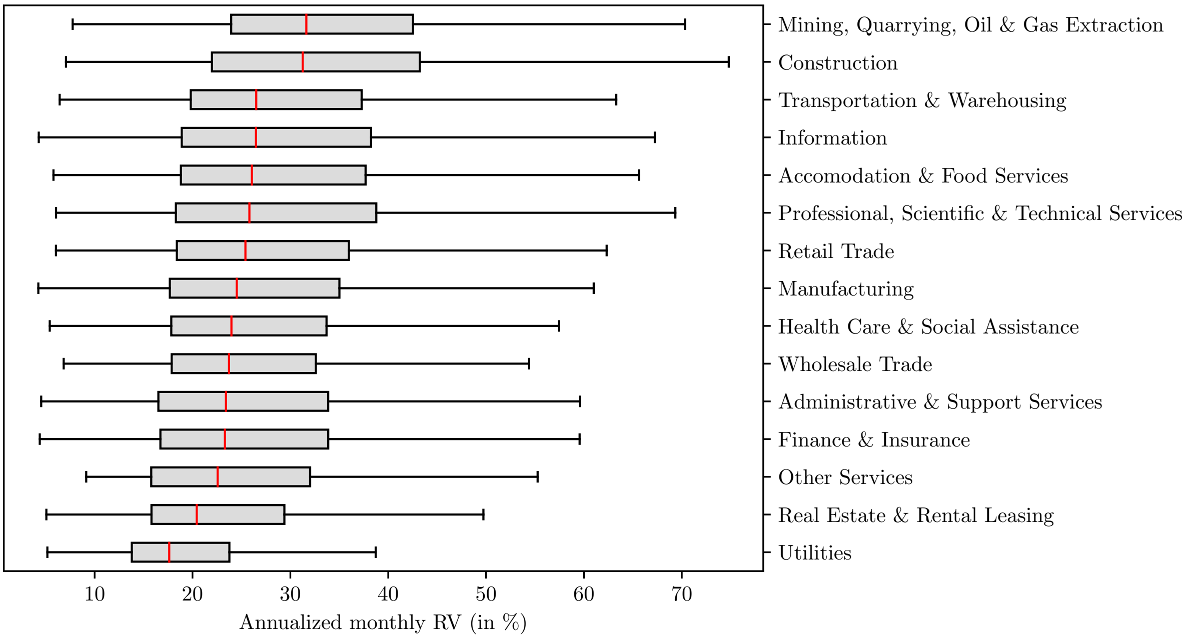
[2] Figure A.1 in the appendix illustrates the development of the two interest rates and spread series over the entire sample period.

whereby the squared daily returns are added up. In this paper, I therefore calculate the annualized weekly (monthly) RV of stock n as

$$RV_n = \sqrt{s}\sqrt{\sum_{t=1}^{T} r_{n,t}^2}, \tag{1}$$

where s denotes the number of weeks (months) per year and T is the number of trading days in the corresponding week (month). Figure 1 illustrates the distribution of annualized monthly RV for each sector.

Figure 1: Distribution of annualized monthly RV by industry



Notes: This figure depicts the distribution of annualized monthly RV per economic sector. The boxplots are sorted in descending order according to the median values.

The boxplots show the sector-specific fluctuations in annualized monthly RV, with the central level varying roughly between 17% and 31%. As expected, sectors such as Utilities have low RV dispersion as their products are of everyday importance and the associated companies tend to have relatively constant revenue expectations. This represents a rather low risk for investors in times of crisis. On the other hand, industries in the information and services sectors traditionally show high volatility. Since the number of companies per sector varies greatly in the sample, the distribution of RV in some sectors is based on few companies (see Table 1 for the number of companies in each sector).

After defining the target variable, I construct the three different sets of features F1, F2 and F3 in the next step. Table 2 presents all features and their classification into the corresponding sets.

Table 2: Model features

No.	Acronym	Explanatory variable	Included in feature set		
			F1	F2	F3
1	SQ_RET	Squared log returns	x	x	x
2	TR_VOL	Trading volume	x	x	x
3	RV_L1	$t-1$ lagged RV	x	x	x
4	RV_L2	$t-2$ lagged RV	x	x	x
5	RV_L3	$t-3$ lagged RV	x	x	x
6	RET_SPX	Log returns S&P 500		x	x
7	VIX	CBOE volatility index		x	x
8	EPU	Economic policy uncertainty index			x
9	EFFR	Effective federal funds rate			x
10	TR_30Y	30-year treasury rate			x
11	SPR_AAA	Spreads AAA firms			x
12	SPR_HY	Spreads HY firms			x
13	EL_SEC	Sectoral employment level			x

Notes: This table provides an overview of the classification of the features used into the three sets F1, F2 and F3.

3 Methodology

Recurrent neural networks (RNN) or LSTM networks, which store information about previous time steps in a memory, have been established as state-of-the-art methods for learning long-term dependencies on sequential data. Vaswani et al. (2017) propose a model architecture with an encoder-decoder structure called transformer, which is no longer based on recurrent units but on the self-attention mechansim. The use of this deep learning mechanism helps the model to dynamically decide how much attention to give to each part of the input sequence by assigning specific weights. To take into account the order of the sequential data, the transformer network uses a special strategy called positional encoding. Transformers changed existing ML approaches to natural language processing (NLP) in a sustainable way since they not only achieve very good results but also proceed more computationally efficient than the previous models due to parallelizable training (Dai et al., 2019). However, the originally proposed encoder-decoder structure is specifically designed for machine translation and therefore not suitable for time series forecasting without adaptation. In the meantime, numerous variations of transformer models for time series forecasting exist, which also show promising results (Wen et al., 2022; Wu et al., 2020; Farsani and Pazouki, 2021). Lim et al. (2021) introduce the TFT model, which combines attention mechanism and LSTM structures in its architecture and is able to process different types of data. They show that their new model performs well in a variety of applications. López Santos et al. (2022) use the TFT structure to predict hourly day-ahead power generation by solar photovoltaic systems. Sappl et al. (2021) apply the model to data from biology by forecasting biogas production rates in anaerobic digesters.

In the following, I explain the TFT approach, the benchmark models as well as the main steps of my methodology in detail. Finally, I discuss the measures used to evaluate the forecasts.

3.1 Temporal fusion transformers

TFT models are deep neural networks (DNN) for multi-horizon forecasting, which are based on the attention mechanism. For the given time series data set with N individual stocks, the architecture allows for a set of categorical variables $c_n \in \mathbb{R}^{m_c}$ and time-varying covariates $X_{n,t} = [z_{n,t}^T, x_{n,t}^T]$, where $z_{n,t} \in \mathbb{R}^{m_z}$ are known only up to the present and $x_{n,t} \in \mathbb{R}^{m_x}$ are a priori known into the future. The corresponding target variable is given by $RV_{n,t} \in \mathbb{R}^+$. Let t denote an arbitrary time step with a given lookback window k and a maximum step-ahead window $\mathcal{T}_{max}$. The model takes the described input variables and computes the $\mathcal{T}$-step-ahead forecast of RV for stock n. The TFT architecture consists of different building blocks to filter relevant information of the features. The following five main units are used for this purpose:

1. Gated residual network (GRN): A special gating mechanism, consisting of several layers, selects unneeded parts of the model architecture to reduce complexity.

2. Variable selection network (VSN): For each time step t, this component chooses meaningful features and assigns corresponding weights to them.

3. LSTM encoder-decoder structure: This layer architecture allows the model to recognize the temporal ordering of the data and thus also the short- and long-term dependencies of the input variables. For the categorical features, the TFT network uses separate encoders.

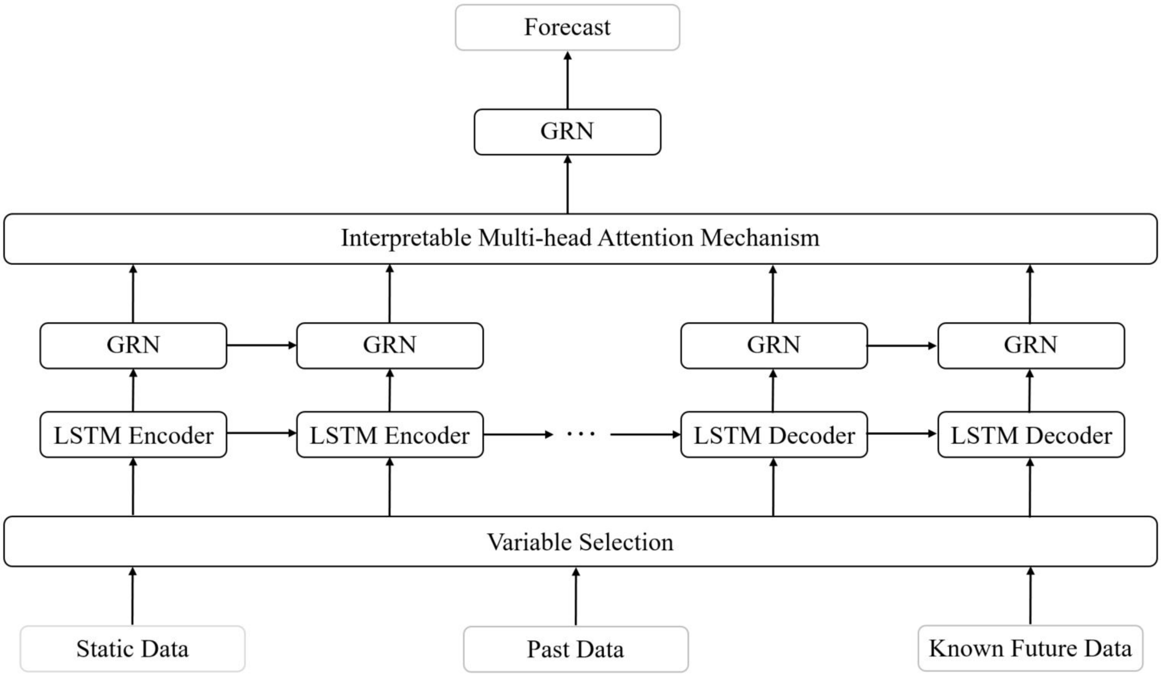
4. Interpretable multi-head attention mechanism: This novel approach provides feature interpretability by adjusting the common mechanism.

5. Quantile regression forecasts: The prediction of several q^{th} percentiles enables the calculation of forecast intervals on top of point forecasts.

Figure 2 illustrates a simplified version of the TFT architecture.

Following Lim et al. (2021), I choose a lookback window k of one year and set as a maximum step-ahead window $\mathcal{T}_{max}$ the complete test data set of 30% of the available data. Table 3 shows the general settings for the TFT network selected in this paper.

Figure 2: Simplified TFT structure



Notes: Own representation based on Sappl et al. (2021). The gray boxes show inputs and outputs. The black boxes are components of the model.

Table 3: Configuration of the TFT model

Chosen hyperparameters of the TFT architecture	
Number of LSTM layers	2
Attention head size	1
Hidden size	160
Minibatch size	64
Dropout regulation	10% at each update
Maximum number of training epochs	500
Loss function	quantile loss function
Learning rate	0.01

Notes: This table lists the selected hyperparameters of the TFT network.

3.2 Benchmark models

As benchmark approaches I use RF, LSTM networks as well as GARCH models. Breiman (2001) introduced the RF, a powerful nonparametric ML algorithm that is based on decision trees (DT). Binary DT can be used for classification and regression tasks and are built by repeated splits (tree nodes) of the training set based on a certain splitting criterion (Breiman et al., 1984). Starting at the root node, each further (inner) node represents a binary decision leading to a split of the data into disjoint subsets. This process ends when no more splits are possible or another stopping criterion such as the predefined maximum depth of a DT is reached. Individual DT are not very robust, resulting in forecasts with low bias but high variance. Ensemble methods such as bagging, boosting or RF are possible approaches to improve their predictive performance (James et al., 2013). The construction of a RF is directly based on the idea of bagging: aggregating the forecasts of a large number of bootstrapped DT. In contrast to the bagging technique, however, only $m < p$ input variables are randomly selected as splitting candidates from the p available features. The hyperparameter m can be chosen differently, but for regression tasks typically $m = p/3$ features are considered (Hastie et al., 2017). This procedure reduces the correlation between the DT. In addition to the randomly drawn bootstrap samples, the RF only chooses a random subset of all available features. I select for the main hyperparameters of the RF the number of trees $B = 500$, the number of randomly selected features $m = p/3$ and a maximum tree depth $J = 8$.[3] I choose the mean squared error (MSE) as the splitting criterion for selecting the respective nodes, which corresponds to the default setting in scikit-learn for regression tasks with RF.

Hochreiter and Schmidhuber (1997) propose LSTM networks in their initial form to prevent the well-known problems of vanishing and exploding gradients often encountered when training RNN. LSTM models add another cell component, the cell state, which enables them to remember information over a long period of time. The cell state regulates the inflow and outflow of information and this process takes place via three specially constructed gates: forget gate, input gate and output gate. Considering the current period t, the memory cell and thus each gate receives input data in the form of the feature vector x_t and the previous hidden state h_{t-1}. The extent to which the cell state changes is determined by the filter mechanism of each gate, which is controlled by sigmoid functions to ensure that the values always lie between zero and one. The specific proportion per gate determines how strongly the available information at time step t influences the cell state.

For the training process, I first prepare the data accordingly by two further steps: scaling and transfer into sequential structure. Scaling features is a common and recommended approach, as it has a significant impact on the efficiency of the training process (Hastie et al., 2017). I scale the entire training data set by applying normalization. Since LSTM networks need a special sequential structure for the training process, i.e., temporally consecutive observations of the corresponding features, the training set is converted into the required three-dimensional form. I choose a quarter as a time step, i.e., three months or 13 weeks are considered as previous periods per sample. During the training process, the entire data set is iteratively fed into the LSTM network for a predefined number of epochs. In each individual epoch, the data set is divided into so-called batches, which consist of one or more samples. The size of the batches determines how often the weights of the LSTM network are adjusted during an epoch. I choose a batch size of 32, which corresponds to the default setting in keras, and a number of 500 epochs. The adjustment of the respective weights is carried out via the *Adam* algorithm (Kingma and Ba, 2017), a stochastic gradient descent method for minimizing a defined loss function (here: MSE). This optimizer was chosen because it has several advantages over other common optimization algorithms (Kingma and Ba, 2017). To avoid overfitting, I additionally implement

[3] Rounded up to the next larger number, the tuning parameters for the three feature sets are $m_1 = 2$, $m_2 = 3$ and $m_3 = 5$.

dropout regulation (Gal and Ghahramani, 2016) and early stopping. I set the dropout ratio to 10% and employ early stopping with a patience of 10.

Besides the two ML benchmarks, I also compare the prediction results with traditional generalized autoregressive conditional heteroskedasticity (GARCH) models. Bollerslev (1986) introduces the GARCH model, an extension of the ARCH model proposed by Engle (1982) as a first systematic approach for describing time-varying conditional volatility. The GARCH(a, b) model describes the current conditional variance as a function of squared error terms of the a previous periods and conditional variances of the b previous periods. There are many variations and extensions of the original GARCH(a, b) model. Hansen and Lunde (2005) provide an excellent overview by comparing 330 GARCH-type models in the performance of out-of-sample volatility forecasts. This paper uses the famous GARCH(1,1) and the GARCH-X model.[4] The GARCH-X model is introduced by Brenner et al. (1996) by inserting additional explanatory variables directly into the variance equation of the GARCH(1,1) model. The idea behind this is that external regressors such as macroeconomic variables (see, for example, Brenner et al., 1996; Engle and Patton, 2001) or additional information from the time series process itself (see, for example, Engle and Gallo, 2006) can describe the volatility process more precisely. In this paper, the GARCH(1,1)-X model adds the following eight additional covariates to the variance equation of the GARCH(1,1) model: RET_SPX, VIX, EPU, EFFR, TR_30Y, SPR_AAA, SPR_HY and EL_SEC.[5] All parameters of the GARCH models are determined using rolling window estimation. The first rolling window contains the data points from 1 to m and then forecasts the volatility for the period $m + 1$. The window is then rolled forward one period so that the second rolling window extends from observation 2 to $m+1$ followed by a prediction for period $m + 2$. This procedure is repeated until the last out-of-sample forecast at time T is reached. To ensure comparability between the data sets for training the ML models, the window sizes are 70% of the available data.

3.3 Approaches of model training

First, I split the three different sets of features into non-overlapping training and test sets. While the proportion of training data in empirical studies typically ranges between 60% and 80%, other authors also suggest values of 50% (see, for example, Hastie et al., 2017). I follow the usual procedure for handling time series data in the context of ML and use a time-based split, where 70% of the data is used to train the models. When training the TFT and LSTM networks, I additionally hold back 10% of the training data for validation purposes. The remaining 30% of the data is used as a test set, i.e., this part is reserved to validate the performance of the trained ML models with so far unknown inputs. In this way, I check whether the model adequately represents functional relationships between the features and the target and can also apply this to unseen information. The training set therefore includes data from January 2000 to October 2015. The test data thus cover about 6.75 years until the end in June 2022. Figure A.2 in the appendix illustrates the subdivision into training and test period using the example of the VIX.

In a second step, I train all models mentioned above using three different approaches on a weekly and monthly basis with all feature sets. For each method, this results in a

[4] The corresponding equations for the two models can be found in the appendix.

[5] The abbreviations of the model features refer to Table 2.

total of 18 training processes.[6] The first approach uses only data from the respective company (*individual approach*). This makes the amount of available data for the respective ML model very limited, but it might be better able to capture certain company-specific aspects. However, the question remains whether this data basis is sufficient for parameter-intensive ML models. The second approach enlarges the respective training datasets by the available data of all companies operating in the corresponding sector of the target company (*sectoral pooling*). Now, the model has considerably more data available in some cases, and yet there is not much heterogeneity among the companies. Since firms in the same industry are exposed to similar risks, the additional data can be helpful in predicting the individual RV. To ensure that the number of companies per sector is not too small, I combine sectors whose number of companies in the sample is less than ten or assign them to similar industries. The grouping of sectors is based on content-related aspects and the correlations of sectoral unemployment. Thus, the number of sectors for this pooling approach is reduced from 15 to 10.[7] In the last approach, the training data set consists of all available data from the 355 stocks (*overall pooling*). The models are not trained individually for all stocks, but a pooled data set with all available companies is used in each case. However, the data come from a wide variety of industries, so that the training set is clearly more heterogeneous and it is therefore more difficult for the ML model to identify functional relationships at the level of individual stocks.

3.4 Forecast evaluation

I evaluate the performance of the different models by comparing two statistical error measures, which assess the accuracy of the forecasts. I use the root mean squared error (RMSE) and the mean absolute error (MAE), two common scale-dependent measures, since they are prominent in the context of RV prediction (see, for example, Kim and Won, 2018; Luong and Dokuchaev, 2018). Due to the large number of different model combinations, I only present the values of the RMSE in Section 4. The appendix contains further results. For each company n, the two measures are formally given by the following equations:

$$RMSE_n = \sqrt{MSE_n} = \sqrt{\frac{1}{T}\sum_{t=1}^{T}(e_t)^2}, \tag{2}$$

$$MAE_n = \frac{1}{T}\sum_{t=1}^{T}|e_t|, \tag{3}$$

where e_t denotes the forecast error at time t for a given set of T true RV values and available predictions.

Since it is not practical to look at all individual error measures, for the sake of clarity I

[6] This is the combination of three training approaches with three sets of features for two frequencies.

[7] The following sectors were combined: (1) Retail Trade, Wholesale Trade, (2) Real Estate & Rental Leasing, Construction, (3) Accomodation & Food Services, Administrative, Support, Waste Management & Remediation Services, Health Care & Social Assistance, Other Services (except Public Administration).

will only show the average of these evaluation measures over all 355 forecasts (mean RMSE and mean MAE). For comparison with the various benchmarks, I also report relative measures by setting the evaluation measure of the model in relation to the benchmark (Hyndman and Koehler, 2006). If they are smaller than one, the respective model was able to beat the benchmark.

This comparison does not indicate whether the result is statistically significant. Diebold and Mariano (1995) propose to test for equal forecast accuracy, i.e., to check whether the two measures are statistically significantly different or not. The idea of the Diebold-Mariano test is based on the loss differential between the two models at period t, which is given by

$$d_t^{1,2} = L(e_t^1) - L(e_t^2), \tag{4}$$

where e_t^i denotes the forecast error for model i and $L(\cdot)$ describes an arbitrary loss function. In this paper, I use the common quadratic loss function. The null hypothesis states that both models have the same forecasting accuracy, i.e., $E(d_t^{1,2}) = 0$.

4 Results

This section discusses the main findings of this paper. I first present the results of the TFT models. I will then explain the performance results of the different benchmarks and also discuss the differences regarding the sets of features and varying training approaches.

4.1 Forecasts of the TFT

Table 4 shows the mean RMSE values for all combinations of training sets, frequencies and approaches mentioned above. It is evident that the model performs significantly better on a weekly frequency than with monthly data. One possible explanation for this finding could be the much smaller data base. Since the TFT network is a DNN and thus learns dependencies using a complex architecture, a training sample that is too small can have a negative impact on the learning process. Similar results from the LSTM model give further support for this explanation. Comparing the performance over the three different training approaches, a clear reduction of the mean RMSE values is recognizable by the use of the two pooling methods. Both pooling approaches substantially improve individual RV forecasts, with overall pooling producing the best results. With a sufficiently large training sample, the TFT network appears to be able to address the heterogeneity across sectors. A comparison of the three sets of features clearly shows that, regardless of temporal frequency and approach chosen, F2 leads to the best prediction results. Taking into account the holistic development of the US stock market thus improves the forecasting results. The returns of the S&P 500 index and the VIX show certain predictive power of individual RV. The inclusion of macroeconomic variables is also helpful since the mean RMSE values of feature set F3 are consistently lower than those of F1.

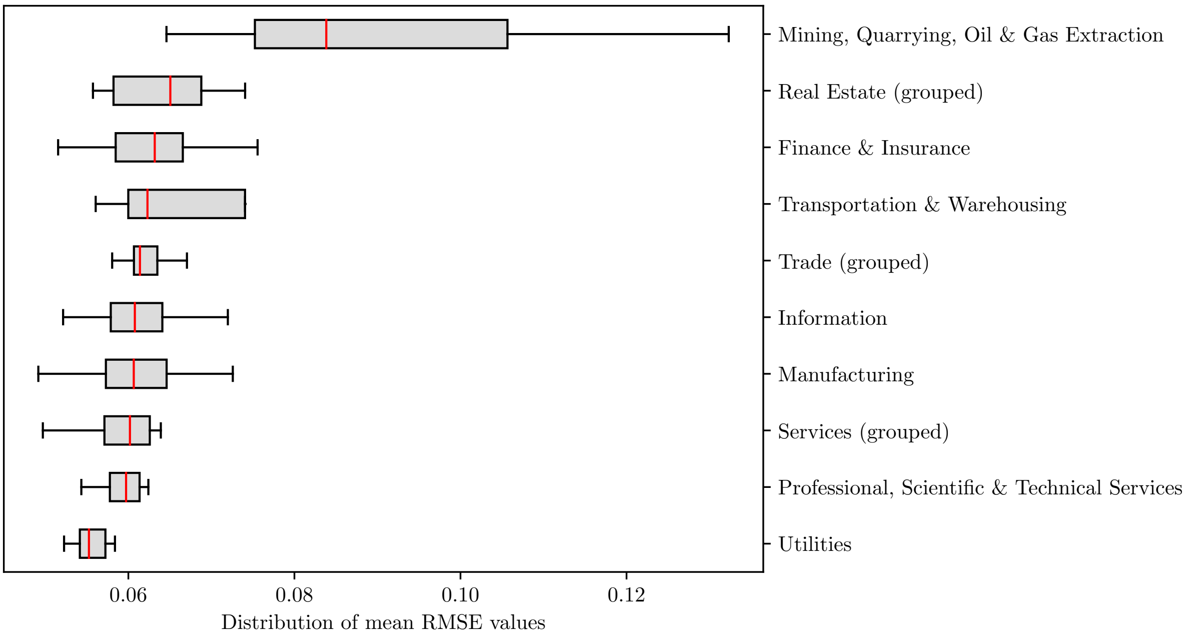
Table 4: Performance measurement of the TFT model

Approach	Frequency	Feature set	Mean RMSE
individual	weekly	F1	0.0902
		F2	**0.0857**
		F3	0.0879
	monthly	F1	0.1410
		F2	0.1376
		F3	0.1393
sectoral pooling	weekly	F1	0.0716
		F2	**0.0692**
		F3	0.0728
	monthly	F1	0.1119
		F2	0.1099
		F3	0.1107
overall pooling	weekly	F1	0.0711
		F2	**0.0699**
		F3	0.0704
	monthly	F1	0.1018
		F2	0.1003
		F3	0.1008

Notes: This table displays the mean RMSE values of the TFT models for all possible combinations of approach, frequency and sets of features. The lowest RMSE values per approach are highlighted in each case.

Even though the TFT model shows the best results on average for overall pooling, a detailed analysis of the sectoral pooling is necessary since the results of this approach differ across the industries. Figure 3 illustrates the sectoral mean RMSE values of the TFT model for the best-performing set of features F2 on a weekly basis. The distribution of the RMSE values shows that with sectoral pooling, the prediction accuracy tends to decline for sectors with higher volatility. The high dispersion in the sector 'Mining, Quarrying, Oil & Gas Extraction' is nevertheless a major outlier to this extent. As Figure 1 already shows, the median RV in this industry is also quite high, but this RMSE distribution is still surprising. A plausible explanation could be the sharp increase in volatility of the corresponding energy companies due to the Russian invasion of Ukraine in February 2022. An event with such an impact is unique for this industry in terms of the training period. Considering the three sectors grouped due to an insufficient number of firms, pooling also seems to provide good results. In particular, the forecasts in the newly constructed trade sector show a low error dispersion, which can be explained by the content coverage and thus very similar risks of wholesale and retail trade. Despite the manual aggregation of individual sectors, it is important to note that the number of companies per sector still varies considerably. For instance, following the NAICS classification, 133 companies, and thus more than one-third of the entire sample, are assigned to the manufacturing sector. This sector includes companies whose activities are extremely heterogeneous as they operate in different markets.[8] One could assume that this fact has a negative impact on the predictive performance of the models when sectoral pooling is applied. Figure 3 shows that this does not seem to be the case as the median of the RMSE values in the corresponding sector is rather low.

Figure 3: Distribution of mean RMSE values by industry



Notes: This figure illustrates the distribution of the mean RMSE values by economic sector when using the TFT network with the best-performing feature set F2 to forecast weekly RV.

4.2 Benchmark models

I now turn to the analysis of the three benchmark models. First of all, it can be generally stated that the two pooling approaches also lead to a considerable improvement in the predictions when using RF and LSTM networks. Overall pooling shows the best results for both ML benchmark models, even if the differences are marginal in some cases. The RF is the only ML approach that shows similar outcomes on a weekly and monthly basis, although the performance is slightly better on a weekly frequency. Furthermore, the forecasts of the RF vary less across the different sets of features. Looking at the results of the LSTM network separately, we see that the mean RMSE values become much more robust with increasing pooling degree. The different sets of features then influence the values only slighty. Table 5 reports the relative mean RMSE values for the ML benchmark models, i.e., the evaluation measures of the corresponding benchmark model in relation to the TFT model. If the measures are smaller than one, the respective model has been able to beat the TFT approach. If they are greater than one, the TFT network has performed better than the benchmark model. As explained in detail above, the TFT network performs rather poorly in the individual approach due to the very limited amount

[8] For example, the manufacturing sector includes the companies Apple, Boeing and Coca-Cola.

of training data. Therefore, the benchmark models partially beat the TFT forecasts when using this training approach. However, when the data are pooled across sectors or the entire sample, the outcomes of the TFT model are for the most part significantly better than those of the other ML methods. This is especially true for weekly forecasts, where the evaluation measures now differ greatly. Table C.1 in the appendix lists all corresponding test statistics for the Diebold-Mariano test and shows that the TFT network, with a few exceptions, is able to make significantly better predictions of the individual RV across all model variants. A comparison of the three sets of features shows that F2, which includes data on the S&P 500 index and the VIX in addition to company-specific characteristics, leads to the the lowest RMSE values across almost all benchmark models and training methods.

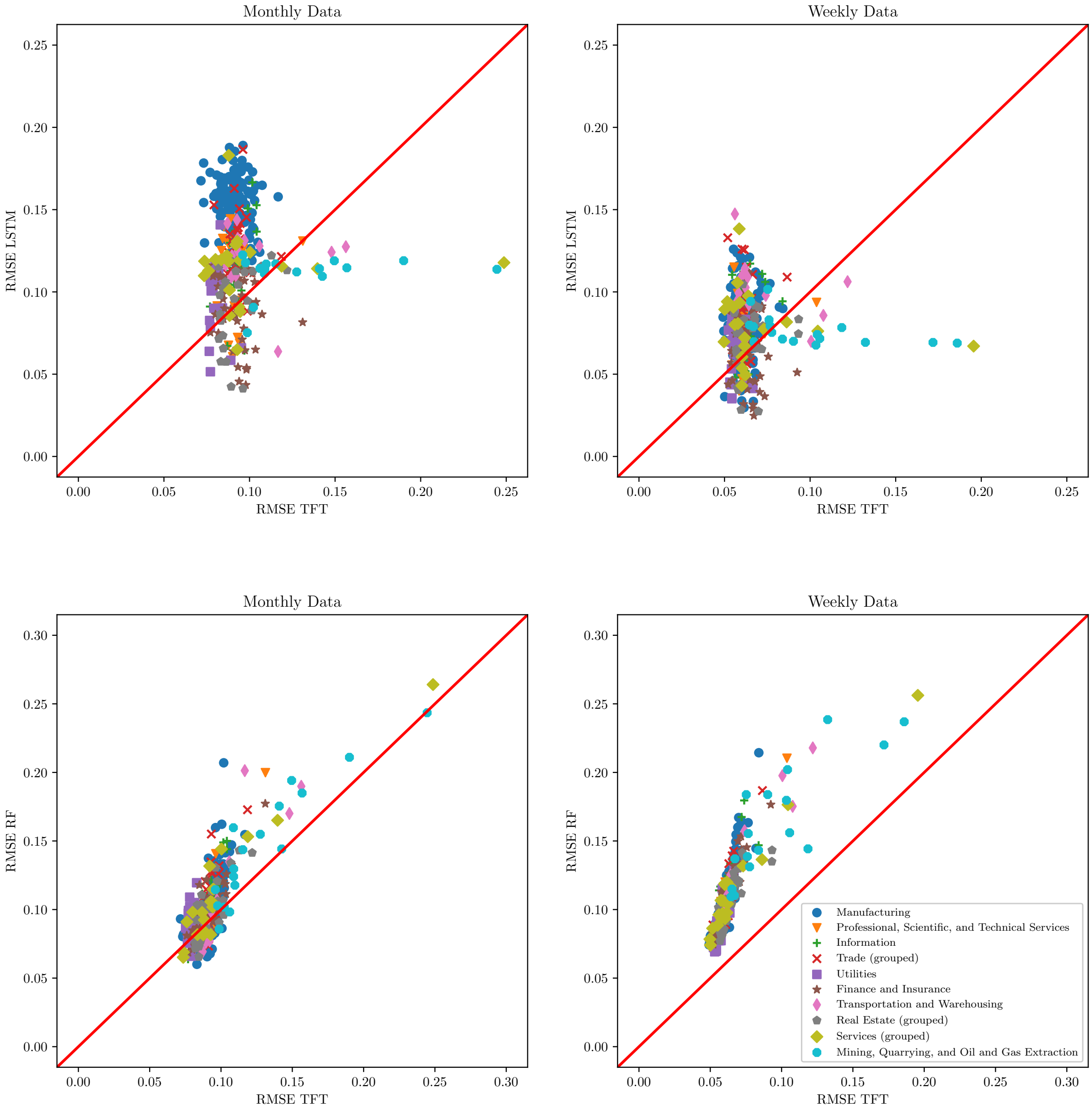
Table 5: Relative performance measurement of the ML benchmark models

Approach	Frequency	Feature set	Relative Mean RMSE	
			RF	LSTM
individual	weekly	F1	**1.3947**	0.9457
		F2	**1.3676**	0.9218
		F3	**1.3811**	**1.0683**
	monthly	F1	0.9390	**1.1837**
		F2	0.8132	**1.0916**
		F3	0.7925	**1.4795**
sectoral pooling	weekly	F1	**1.6383**	**1.0992**
		F2	**1.6069**	**1.0867**
		F3	**1.5440**	**1.1099**
	monthly	F1	**1.0974**	**1.1090**
		F2	0.9454	**1.0992**
		F3	0.9575	**1.1545**
overall pooling	weekly	F1	**1.6273**	**1.2208**
		F2	**1.5837**	**1.1960**
		F3	**1.5668**	**1.2670**
	monthly	F1	**1.1807**	**1.1110**
		F2	**1.0269**	**1.0977**
		F3	**1.0516**	**1.1329**

Notes: This table displays the realtive mean RMSE values of the ML benchmark models in relation to the TFT network. RMSE values greater than one show the cases where the TFT model beats the benchmark, and are therefore highlighted.

To illustrate the differences between the various models in predicting RV, Figure 4 plots all RMSE values for the individual sectors. All models shown are based on the best-performing set of features F2. It is noticeable that the RMSE values of the TFT network correlate positively with those of the RF, while they are uncorrelated with the performance measures of the LSTM model. As expected, the RF produces similarly poor forecast results in the sector ‘Mining, Quarrying, Oil & Gas Extraction’ that I have already mentioned above. Negative performance outliers can also be observed in this respect in the sectors ‘Services’ and ‘Transportation and Warehousing’. In contrast, the LSTM network has problems in predicting individual RV of companies in the sector ‘Manufacturing’. One possible explanation could be the great heterogeneity of the business areas of the associated companies.

Figure 4: Scatterplot of monthly and weekly RMSE values by industry



Notes: This figure shows the scatterplots of the RMSE values of the TFT model and the two ML benchmarks on a monthly (left plots) and weekly (right plots) frequency split by economic sector. The subfigures in the upper row refer to the LSTM network as a benchmark, the subfigures in the lower row refer to RF.

Comparing the performance of the TFT method with econometric GARCH models, the predictions are significantly better regardless of selected frequency, training approach and set of features. Table 5 in the appendix reports the corresponding performance measures for the GARCH(1,1) and the GARCH(1,1)-X model. In contrast to all ML approaches, both GARCH models perform better on a monthly data basis.

5 Robustness

Different factors such as the chosen hyperparameters or even randomness play an important role for the performance of all ML models. The initialization of weights during the training process of the TFT and LSTM networks or random feature selection of the RF influence the results, regardless of the chosen model configuration. In order to verify that the results are robust, I carry out a series of robustness checks. Following Schnaubelt et al. (2020), I implement the following two checks: changes of seed values and implementation of various model configurations.

For replication purposes, I set constant seed values for the training processes of all models presented so far. It is therefore possible that the results are partly driven by the choice of a specific seed value. To rule this out, I train all models with three different seed variations. Table D.1 in the appendix displays the average performance measures for all three seed variations. I find that the TFT and LSTM networks react more sensitively to changed seed values than the RF models. Overall, however, the results obtained with different seed values do not change with respect to the sets of features or training approaches and therefore do not affect the model ranking. I conclude that the setting of the seed value has a small influence on the results of the models and that they are robust.

The second robustness check examines whether the results are significantly dependent on the specific construction of the different ML models. For this purpose, I implement several variations of the ML approaches. The TFT networks are trained with an increased attention head size of 4, a new state size of 80 and an adjusted dropout ratio of 30%. For the LSTM networks, I change the number of neurons in the hidden layer from 100 to 200. The resulting increase in the number of parameters not only extends the training time, but should also improve the network's ability to learn complex functional relationships. I also run the RF model with a different number of trees and a lower depth per tree. Since increasing the number of trees does not lead to overfitting (Hastie et al., 2017), I double the number to 1000 trees and set the maximum depth per tree to 4. Table D.2 in the appendix shows how the changed model variations influence the performance of the individual methods for predicting RV. The more complex model architecture leads to marginally worse results for the TFT model on a weekly frequency, but noticeably improves the results on a monthly data basis. The RF predictions, on the other hand, become less accurate with the new setting, regardless of the frequency. A depth of 4 per tree therefore appears to be insufficient for specific predictions of individual RV. The results of the LSTM network with doubled number of neurons lead to a slight improvement on monthly frequency. As expected, the overall outcomes of the different ML models vary more than for changed seed values, but are also robust with respect to the ranking of the methods.

6 Conclusion

This paper investigates the performance of the TFT model, a novel DNN architecture for multi-horizon forecasting, in predicting annualized weekly and monthly RV of individual stocks listed in the S&P 500 index. This network combines different types of input variables to learn short- and long-term dependencies based on a specific gating procedure.

For the training process of the different ML methods, I use three different sets of features as well as three different training approaches. The former allows statements to be made about the predictive power of individual input variables, the latter about the data basis to be selected for the models. The TFT network is able to beat econometric GARCH benchmarks, regardless of the training process or the frequency chosen. The TFT approach also significantly outperforms two ML benchmarks, namely LSTM networks and RF, when using sectoral and overall pooling. The use of pooling methods generates the model a much larger data set, but also leads to growing heterogeneity in the data. I find that sectoral and overall pooling improves the prediction performance for all ML models. The use of exclusively company-specific data in the individual approach, and thus a rather small data set, means that the TFT network still clearly beats classic GARCH models, but less frequently common ML approaches. My results confirm that ML algorithms are very useful for forecasting financial volatility. In particular, they suggest that the novel and flexible TFT approach might become helpful in asset pricing and risk management.

References

Andersen, T. G., Bollerslev, T., Diebold, F. X., and Ebens, H. (2001). The distribution of realized stock return volatility. *Journal of Financial Economics*, 61: 43–76.

Baker, S. R., Bloom, N., and Davis, S. J. (2016). Measuring economic policy uncertainty. *The Quartely Journal of Economics*, 131(4): 1593–1636.

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

Breiman, L. (2001). Random forests. *Machine Learning*, 45(1): 5–32.

Breiman, L., Friedman, J., Olshen, R. A., and Stone, C. J. (1984). *Classification and regression trees*. New York: Chapman & Hall.

Brenner, R. J., Harjes, R. H., and Kroner, K. F. (1996). Another look at models of the short-term interest rate. *The Journal of Financial and Quantitative Analysis*, 31(1): 85–107.

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

Christiansen, C., Schmeling, M., and Schrimpf, A. (2012). A comprehensive look at financial volatility prediction by economic variables. *Journal of Applied Econometrics*, 27(6): 956–977.

Dai, Z., Yang, Z., Yang, Y., Carbonell, J., Le V, Q., and Salakhutdinov, R. (2019). Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*: 2978–2988.

Diebold, F. X. and Mariano, R. S. (1995). Comparing predictive accuracy. *Journal of Business and Economic Statistics*, 13(3): 253–263.

Engle, R. F. (1982). Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation. *Econometrica*, 50(4): 987–1007.

Engle, R. F. and Gallo, G. M. (2006). A multiple indicators model for volatility using intra-daily data. *Journal of Econometrics*, 131(1): 3–27.

Engle, R. F. and Patton, A. J. (2001). What good is a volatility model? *Quantitative Finance*, 1: 237–245.

Farsani, R. M. and Pazouki, E. (2021). A Transformer Self-Attention Model for Time Series Forecasting. *Journal of Electrical and Computer Engineering Innovations*, 9(1): 1–10.

Gal, Y. and Ghahramani, Z. (2016). A theoretically grounded application of dropout in recurrent neural networks. *Advances in Neural Information Processing Systems*: 1019–1027.

Hansen, P. R. and Lunde, A. (2005). A forecast comparison of volatility models: Does anything beat a GARCH(1,1)? *Journal of Applied Econometrics*, 20(7): 873–889.

Hastie, T., Tibshirani, R., and Friedman, J. (2017). *The elements of statistical learning: Data mining, inference, and prediction.* 2nd edition. New York: Springer.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(3): 1735–1780.

Hu, X. (2021). Stock Price Prediction Based on Temporal Fusion Transformer. *3rd International Conference on Machine Learning, Big Data and Business Intelligence*: 60–66.

Hyndman, R. J. and Koehler, A. B. (2006). Another look at measures of forecast accuracy. *International Journal of Forecasting*, 22(4): 679–688.

James, G., Witten, D., Hastie, T., and Tibshirani, R. (2013). *An introduction to statistical learning: With applications in R.* New York: Springer.

Kim, H. Y. and Won, C. H. (2018). Forecasting the volatility of stock price index: A hybrid model integrating LSTM with multiple GARCH-type models. *Expert Systems with Applications*, 103(1): 25–37.

Kingma, D. P. and Ba, J. (2017). Adam: A method for stochastic optimization. *arXiv*: 1412.6980.

Lim, B., Arık, S. Ö., Loeff, N., and Pfister, T. (2021). Temporal Fusion Transformers for interpretable multi-horizon time series forecasting. *International Journal of Forecasting*, 37(4). PII: S0169207021000637: 1748–1764.

López Santos, M., García-Santiago, X., Echevarría Camarero, F., Blázquez Gil, G., and Carrasco Ortega, P. (2022). Application of Temporal Fusion Transformer for Day-Ahead PV Power Forecasting. *Energies*, 15(14): 5232.

Luong, C. and Dokuchaev, N. (2018). Forecasting of realised volatility with the random forests algorithm. *Journal of Risk and Financial Management*, 11(4): 61–76.

Sappl, J., Harders, M., and Rauch, W. (2021). Vorhersage von Zeitserien der Biogasproduktion in anaeroben Faultürmen mit einem Temporal Fusion Transformer. *Österreichische Wasser- und Abfallwirtschaft*, 73. PII: 770: 329–336.

Schnaubelt, M., Fischer, T., and Krauss, C. (2020). Separating the signal from the noise - financial machine learning for twitter. *Journal of Economic Dynamics and Control*, 114: 103895.

Shaikh, I. (2019). On the relationship between economic policy uncertainty and the implied volatility index. *Sustainability*, 11(6): 1628.

Tsay, R. S. (2010). *Analysis of financial time series.* 3rd edition. Hoboken: John Wiley & Sons.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention Is All You Need. *31st Conference on Neural Information Processing Systems*: 6000–6010.

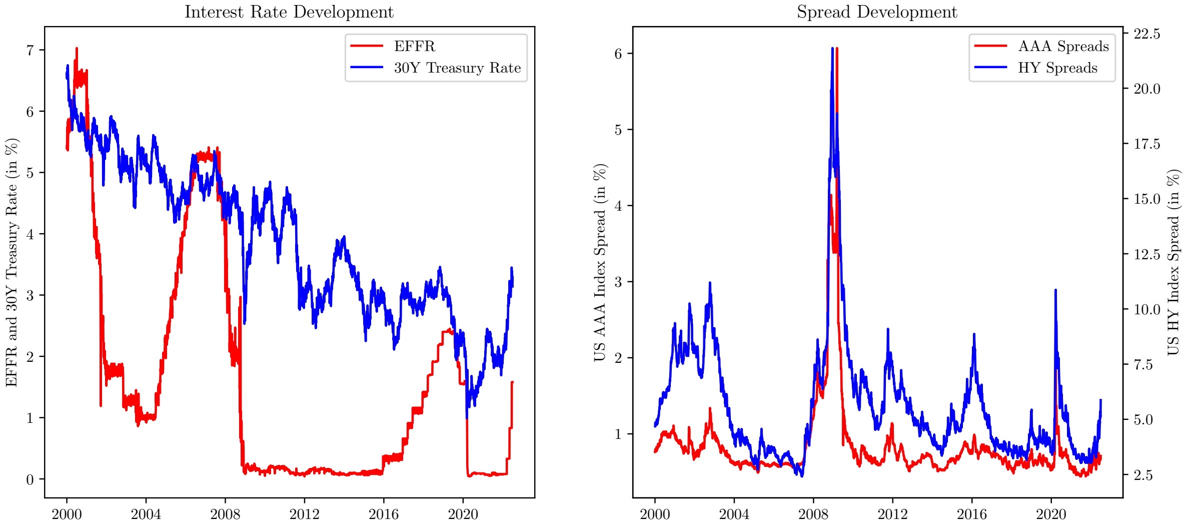
Wen, Q., Zhou, T., Zhang, C., Chen, W., Ma, Z., Yan, J., and Sun, L. (2022). Transformers in Time Series: A Survey. *arXiv:2202.07125v3.*

Wu, N., Green, B., Ben, X., and O'Banion, S. (2020). Deep Transformer Models for Time Series Forecasting: The Influenza Prevalence Case.

Appendix

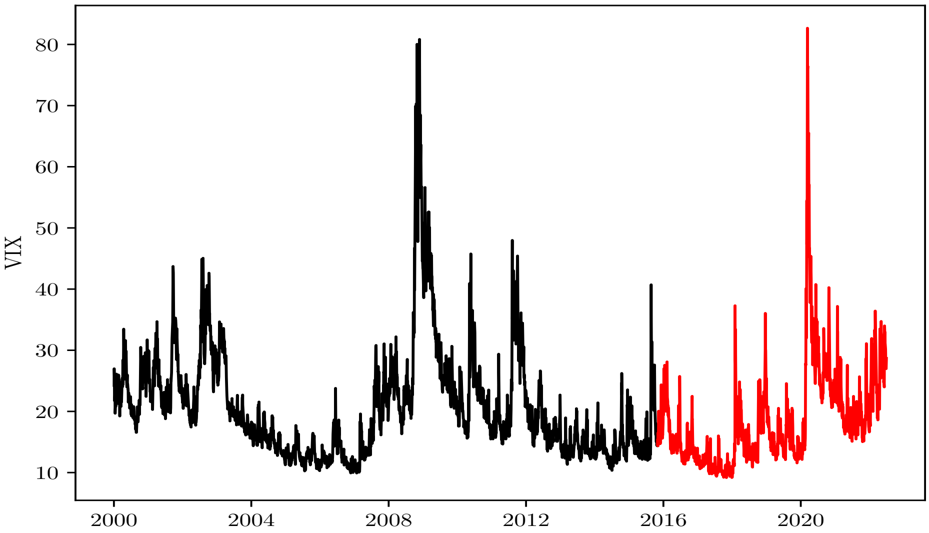
A Additional figures

Figure A.1: Development of interest rates and spreads



Notes: The left plot presents the development of the EFFR and the 30-year treasury constant maturity rate, the right plot the development of AAA and HY spreads.

Figure A.2: Training and test period



Notes: This plot shows the split of the time series into training (black) and test period (red), using the VIX as an example.

B GARCH models

This section provides additional information on the GARCH benchmarks and lists the results achieved in detail. In general, the GARCH(a, b) model can be written as

$$\sigma_t^2 = \alpha_0 + \sum_{i=1}^{a} \alpha_i \epsilon_{t-i}^2 + \sum_{j=1}^{b} \beta_j \sigma_{t-j}^2, \qquad \epsilon_t = \sigma_t Z_t, \tag{5}$$

with $Z_t \overset{iid}{\sim} \mathcal{N}(0,1)$. The parameter constraints to meet the non-negativity and stationarity condition of the variance are: $\alpha_0 > 0$, $\alpha_i \geq 0$, $\beta_j \geq 0$ and $\sum_{i=1}^{max(a,b)}(\alpha_i + \beta_i) < 1$ (Tsay, 2010). This paper focuses on the famous GARCH(1,1) and the GARCH-X model, where additional explanatory variables are included in the variance equation. The variance equation of a GARCH(1,1)-X model is therefore given by

$$\sigma_t^2 = \alpha_0 + \alpha_1 \epsilon_{t-1}^2 + \beta_1 \sigma_{t-1}^2 + \sum_{l=1}^{r} \gamma_l x_{l,t-1}^2, \tag{6}$$

where x_l is one additional covariate.

Table 5 reports the mean RMSE and mean MAE values of both GARCH models for the RV forecasts of all 355 stocks.

Table 5: Performance measurement of the GARCH models

Model	Frequency	Mean RMSE	Mean MAE
GARCH(1,1)	weekly	0.1694	0.1140
	monthly	0.1543	0.0915
GARCH(1,1)-X	weekly	0.2047	0.1270
	monthly	0.1559	0.0930

C Results of the Diebold-Mariano test

This section shows in detail the results of the Diebold-Mariano test. Table C.1 reports the respective test statistics and the corresponding p-values for all model variants. In each case, the performance of the TFT models is compared with the benchmark models listed in the last three columns of the table.

Table C.1: Results of the Diebold-Mariano test

Approach	Frequency	Feature set	DM test statistic		
			RF	LSTM	GARCH
individual	weekly	F1	-39.76 (0.00)	2.36 (0.00)	-61.55 (0.00)
		F2	-40.23 (0.00)	3.14 (0.00)	-
		F3	-37.75 (0.00)	-5.74 (0.00)	-75.78 (0.00)
	monthly	F1	2.91 (0.00)	-10.79 (0.00)	-10.30 (0.00)
		F2	10.75 (0.00)	-7.25 (0.00)	-
		F3	11.40 (0.00)	-40.11 (0.00)	-7.37 (0.00)
sectoral pooling	weekly	F1	-47.84 (0.00)	-7.40 (0.00)	-
		F2	-49.38 (0.00)	-6.56 (0.00)	-
		F3	-43.36 (0.00)	-7.48 (0.00)	-
	monthly	F1	-7.36 (0.00)	-7.43 (0.00)	-
		F2	2.44 (0.02)	-7.40 (0.00)	-
		F3	0.96 (0.34)	-9.17 (0.00)	-
overall pooling	weekly	F1	-50.04 (0.00)	-13.89 (0.00)	-
		F2	-49.93 (0.00)	-11.90 (0.00)	-
		F3	-48.38 (0.00)	-16.01 (0.00)	-
	monthly	F1	-11.06 (0.00)	-7.83 (0.00)	-
		F2	-4.01 (0.00)	-7.10 (0.00)	-
		F3	-5.64 (0.00)	-8.11 (0.00)	-

Notes: This table shows the DM test statistics and the corresponding p-values. The predictions of the TFT models are compared with the listed benchmarks. If the test statistics are positive, the benchmark beats the TFT model.

D Results of the robustness checks

Table D.1: Average performance measures with seed variations

Approach	Frequency	Feature set	Avg. Mean RMSE		
			TFT	RF	LSTM
individual	weekly	F1	0.0905	0.1259	0.0858
		F2	0.0855	0.1172	**0.0798**
		F3	0.0882	0.1212	0.0933
	monthly	F1	0.1469	0.1321	0.1657
		F2	0.1389	0.1118	0.1493
		F3	0.1402	0.1107	0.2045
sectoral pooling	weekly	F1	0.0740	0.1174	0.0796
		F2	**0.0699**	0.1109	0.0766
		F3	0.0715	0.1127	0.0818
	monthly	F1	0.1104	0.1234	0.1255
		F2	0.1091	0.1041	0.1213
		F3	0.1116	0.1061	0.1297
overall pooling	weekly	F1	0.0712	0.1161	0.0851
		F2	**0.0708**	0.1103	0.0842
		F3	0.0716	0.1106	0.0911
	monthly	F1	0.1003	0.1206	0.1169
		F2	0.0998	0.1025	0.1126
		F3	0.1020	0.1049	0.1108

Notes: This table displays the mean RMSE values of all ML models as an average over all three seed variations. The lowest values per approach are highlighted in each case.

Table D.2: Performance measures with different model structures

Approach	Frequency	Feature set	Mean RMSE		
			TFT	RF	LSTM
individual	weekly	F1	0.0921	0.1358	0.0889
		F2	**0.0820**	0.1254	0.0831
		F3	0.0842	0.1305	0.0995
	monthly	F1	0.1472	0.1404	0.1629
		F2	0.1421	0.1211	0.1480
		F3	0.1460	0.1233	0.1922
sectoral pooling	weekly	F1	0.0722	0.1191	0.0788
		F2	**0.0688**	0.1121	0.0749
		F3	0.0692	0.1132	0.0798
	monthly	F1	0.1098	0.1246	0.1232
		F2	0.1080	0.1051	0.1192
		F3	0.1103	0.1072	0.1244
overall pooling	weekly	F1	0.0699	0.1213	0.0833
		F2	**0.0692**	0.1179	0.0813
		F3	0.0721	0.1181	0.0875
	monthly	F1	0.1009	0.1266	0.1138
		F2	0.1033	0.1108	0.1109
		F3	0.1061	0.1123	0.1092

Notes: This table displays the mean RMSE values of all ML models with a different model structure. The TFT network has now a head size of 4, a state size of 80 and a dropout ratio of 30%. I increased the number of neurons in the LSTM hidden layer from 100 to 200. For the RF models, I doubled the number of trees to 1000 and reduced the depth per tree to 4. The lowest values per approach are highlighted in each case.