Computational and Applied Mathematics (2022) 41:66
https://doi.org/10.1007/s40314-022-01765-x

Frequency-based ensemble forecasting model for time series forecasting

Waddah Saeed[1]

Received: 13 April 2021 / Revised: 19 December 2021 / Accepted: 6 January 2022 /
Published online: 7 February 2022

Abstract
The M4 forecasting competition challenged the participants to forecast 100,000 time series with different frequencies: hourly, daily, weekly, monthly, quarterly, and yearly. These series come mainly from the economic, finance, demographics, and industrial areas. This paper describes the model used in the competition, which is a combination of statistical methods, namely auto-regressive integrated moving-average, exponential smoothing (ETS), bagged ETS, temporal hierarchical forecasting method, Box-Cox transformation, ARMA errors, Trend and Seasonal components (BATS), and Trigonometric seasonality BATS (TBATS). Forty-nine submissions were evaluated by the organizers and compared with 12 benchmarks and standards for comparison forecasting methods. Based on the results, the proposed model is listed among the 17 submissions that outperform the 12 benchmarks and standards for comparison forecasting methods, ranked 15th on average and 4th with the weekly time series. In addition, a further comparison was conducted between the proposed model and other forecasting methods on forecasting EUR/USD exchange rate and Bitcoin closing price time series. It is apparent from the results that the proposed model can produce accurate results compared to many forecasting methods.

Keywords Time series · Forecasting · M4 competition · Frequency · Ensemble model · Statistical methods

Mathematics Subject Classification 60G25 · 62M10

1 Introduction

Time series is a type of data that is observed sequentially over time. Examples of time series include unemployment rates, stock prices, sales demand, birth rates, temperatures, and website traffic.

Communicated by Clémentine Prieur.

✉ Waddah Saeed
waddah.waheeb@uia.no

[1] Department of Information and Communication Technology, University of Agder, Jon Lilletuns vei 9, 4879 Grimstad, Norway

Time series forecasting is an interesting area for researchers in academia and industry. Generally speaking, time series forecasting methods are preferred for forecasting compared to explanatory or mixed models for some reasons (Hyndman and Athanasopoulos 2018) such as they are reasonably useful in modeling systems that are not understandable, and they do not need to know or forecast the future values of various predictors when forecasting the variable of interest, unlike explanatory models. Furthermore, time series forecasting methods may produce accurate forecasts compared to explanatory o mixed models.

Forecasting competitions have contributed significantly to enlightening our knowledge of what forecasting methods work (Hyndman 2020). The purpose of the M4 forecasting competition is to improve forecasting accuracy and As a result, advancing the theory and practice of forecasting (Makridakis et al. 2020).

The M4 forecasting competition challenged the participants to forecast the future values for 100,000 time series (Makridakis et al. 2020). The organizer collected diverse time series, mainly related to economic, finance, demographics, and industrial areas. Different frequencies were considered in the competition, namely hourly, daily, weekly, monthly, quarterly, and yearly.

According to the organizers, 49 valid submissions were received (Makridakis et al. 2020). Only 17 of 49 submissions outperform 12 benchmarks and standards for comparison forecasting methods using the selected accuracy measure. The majority of these 17 submissions used combinations of mostly statistical methods (Makridakis et al. 2020).

One of the useful things of competitions is sharing the methods used by the participants, which can help create benchmark forecasting methods. Therefore, newly proposed time series forecasting methods can be compared with these benchmark forecasting methods; this was done for many years with previously published benchmark methods (Hyndman 2020). Furthermore, it has been argued that publication consideration for any new general time series forecasting method might be given if it shows very close or better forecasting performance compared to existing forecasting methods when evaluated using a subset of the M4 data (Hyndman 2020).

With this in mind, the key contributions in this work can be summarized as follows:

- Describe the proposed model used to forecast the 100,000 time series used in the M4 forecasting competition. This model is ranked 15th on average and 4th with the weekly time series. This model used a combination of statistical forecasting methods to produce the forecasts.
- Analyze and discuss the obtained results compared to benchmarks and standards for comparison forecasting methods selected by the organizers.
- Conduct a further comparison between the proposed model and other forecasting methods on forecasting EUR/USD exchange rate and Bitcoin closing price time series.

The present paper consists of four main parts. The first part describes the M4 forecasting competition, including time series data and forecast horizon for each frequency, the benchmark and standard forecasting methods used by the organizers, and the accuracy measure used to evaluate and rank the submissions. The second part details the algorithm behind the model used by the author in the competition, stating its phases, forecasting method components, and the combination strategy. The next part presents, analyses, and discusses the results. In addition, a comparison between the proposed model and other published results using other time series data was provided. A summary of the paper and conclusions are given in the last section.

Table 1 Number of time series and forecasting horizon per data frequency (Makridakis et al. 2020)

	Hourly	Daily	Weekly	Monthly	Quarterly	Yearly
Number of time series	414	4,227	359	48,000	24,000	23,000
Forecast horizon	48	14	13	18	8	6

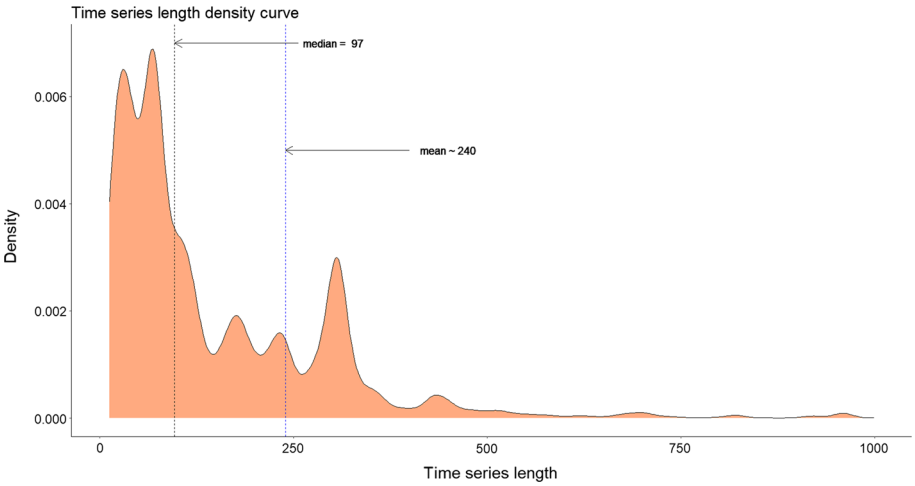


Fig. 1 Time series length density curve

2 The background

The 100,000 time series of the dataset contains different frequencies, including hourly, daily, weekly, monthly, quarterly, and yearly data. The dataset is divided into a training set provided at the beginning of the competition, and a test set kept with the organizers and released after days from the submission deadline. Table 1 shows the number of time series and forecasting horizon per data frequency.

Figure 1 shows time series length density curve for 97,128 time series in the M4 dataset, which has less than or equal to 1,000 observations. As it can be seen from this figure, around half of the time series in the dataset (i.e., 50,371 time series) with less than or equal to 97 observations (i.e., the median). Due to the short nature of most time series in the M4 dataset and time and resource constraints, machine learning methods (e.g., neural networks) were not considered a component in the proposed model.

The benchmarks and standards for comparison forecasting methods used in the competition are as follows:

- Statistical benchmarks: Naïve 1, Naïve 2, Naïve S, single exponential smoothing (SES), Holt, damped, Comb, and Theta.
- Machine learning benchmarks: multi-layer perceptron (MLP) and recurrent neural network (RNN).
- Standards for comparison: auto-regressive integrated moving-average (ARIMA) and exponential smoothing (ETS).

Naïve 1 is a random walk model that produces forecasts equal to the last observed value of the series. The difference between Naïve 1 and Naïve 2 (Makridakis et al. 1998) is that

the data in Naïve 2 are seasonally adjusted by applying a multiplicative decomposition. For Naïve S, the forecasts are equal to the most recent observation of the same period.

The following three exponentially smoothing methods were used as statistical benchmarks: SES (Gardner 1985), Holt (Gardner 2006), and damped (Gardner 2006). The produced forecasts using these methods are weighted averages of past values, with the weights decreasing exponentially as the values come from further in the past (Hyndman and Athanasopoulos 2018). SES method assumes no trend or seasonality in the data. Forecasts using the SES method take the same value equal to the last estimated level (i.e., smoothed value). For the Holt method, forecasts equal the last estimated level plus h times the last estimated trend value (Hyndman and Athanasopoulos 2018). Instead of producing forecasts that are a linear function of h, the curve is damped (flattens over time) with the Damped method (Hyndman and Athanasopoulos 2018). Seasonal adjustments are considered as per Naïve 2 with the three methods. Finally, the Comb method is calculated by taking the average forecasts of SES, Holt, and Damped methods.

Theta method (Assimakopoulos and Nikolopoulos 2000) was used for the first time in the M3 competition. According to Assimakopoulos and Nikolopoulos (2000), the main idea of this method is about modifying the local curvature of the time series with a coefficient called Theta, and this is done by applying this coefficient to the second difference of the time series. Two Theta lines are used for the resulting new time series; the first line is extrapolated using linear regression, while the second one uses the SES method (Makridakis et al. 2020). After the extrapolation, the final forecasts are combined using equal weights. Seasonal adjustments are also considered with the Theta method as per Naïve 2.

Two machine learning methods were used as benchmarks: multi-layer perceptron (MLP) and recurrent neural network (RNN). An MLP is a class of feedforward neural networks in which the links between the nodes do not form a cycle as with RNNs. An MLP typically has three layers as follows: an input layer, a hidden layer, and an output layer. The weights that link between the nodes in the MLP are trained using a learning technique called backpropagation algorithm (Rumelhart et al. 1986). In the competition, an MLP was used with 6 hidden nodes in the hidden layer. Adam (Kingma and Ba 2014) was used as a solver for weight optimization for MLP. Turning now to the RNN, another class of neural network that is more suitable for modeling sequence data (e.g., time series). The RNN used in the competition is a fully-connected RNN in which the output from the previous time step is fed to the next timestep. The settings used with RNN are 6 output space units in the first layer and 1 in the last layer, and RMSprop as an optimizer (Hinton et al. 2012). Detrending and deseasonalization processes are applied to facilitate the extrapolation for both methods.

With ARIMA (Box et al. 2015), the forecasts are calculated by linearly combining the previous values of the variable and the past forecast errors. A variation of auto ARIMA algorithm (Hyndman and Khandakar 2008) was used in the competition. This algorithm combines unit root tests, minimization of the Akaike information criterion corrected for small sample bias (AICc), and maximum likelihood estimation (MLE) to obtain the best ARIMA model (Hyndman and Athanasopoulos 2018) for the given time series. With ETS (Hyndman et al. 2002), an automatic exponential smoothing state-space model is generated for the given time series. The forecasts from ETS are equal to the medians of the forecast distributions (Hyndman and Athanasopoulos 2018). The organizers included both auto ARIMA and ETS due to their popularity in forecasting studies (Makridakis et al. 2020).

Regarding the performance measure, the organizers used the average of two accuracy measures, referred to as the overall weighted average (OWA). The first one is the symmetric mean absolute percentage error (sMAPE) (Makridakis 1993) and the second one is the mean absolute scaled error (MASE) (Hyndman and Koehler 2006). The former is given by Eq. (1)

and the latter by Eq. (2).

$$\text{sMAPE} = \frac{2}{h}\sum_{t=n+1}^{n+h}\frac{|Y_t - \hat{Y}_t|}{|Y_t| + |\hat{Y}_t|} * 100(\%) \tag{1}$$

$$\text{MASE} = \frac{1}{h}\frac{\sum_{t=n+1}^{n+h}|Y_t - \hat{Y}_t|}{\frac{1}{n-m}\sum_{t=m+1}^{n}|Y_t - Y_{t-m}|}, \tag{2}$$

where Y_t is the observation value at point t, $\hat{Y}_t$ the forecast, n the number of in-sample observations, h the forecasting horizon, and m the period between consecutive observations for each data frequency, i.e., 12 for monthly, 4 for quarterly, 24 for hourly and 1 for yearly, weekly and daily data. It is good to note that the organizers used Naïve S instead of Naïve 1 in the MASE because it provides a more reasonable scaling option for seasonal series (Makridakis et al. 2020). Interested readers about the details of the M4 competition may refer to (Makridakis et al. 2020).

3 The proposed model

Due to time and resource limits, the author investigated a few forecasting methods using the training set provided by the organizers. These forecasting methods are Temporal HIErarchical Forecasting (THIEF) (Athanasopoulos et al. 2017), BATS and TBATS (Livera et al. 2011), Bagged ETS (BaggedETS), auto ARIMA (Hyndman and Khandakar 2008),and ETS (Hyndman et al. 2002).

The idea behind using THIEF, BATS, TBATS, and BaggedETS is to utilize recent forecasting methods that are found useful to model and forecast time series due to some advantages found in these methods, as described later in this section. ARIMA and ETS were used because they are commonly used in forecasting studies.

The selection of the forecasting methods for each frequency was made as follows. Each time series was split into two sets, fitting and validation sets. The number of observations in the validation set equals the needed forecasts for that series in the competition. The forecasting methods were fitted then ranked by their performance on the validation set. Based on that, it was found by the author that the performance of these forecasting methods varies when were analyzed based on the frequency of the time series. Due to the size of the dataset and time and resources limits, different combinations of these methods were tried for each frequency. The best combination in terms of the overall weighted average (OWA) was used to produce the final forecasts. Combining forecasts is found useful to improve the forecasting accuracy, especially when we are uncertain about which method is most accurate (Armstrong 2001). Furthermore, it is found that combined forecasts are sometimes more accurate than their most accurate components (Armstrong 2001).

The forecasting methods for each frequency in the proposed model are shown in Fig. 2. These methods were applied as implemented in the forecast v. 8.3 and thief v. 0.3 for R programming language (Hyndman et al. 2018; Hyndman and Kourentzes 2018; Team et al. 2018). A brief description of these methods, except for ETS (Hyndman et al. 2002) which was described in the previous section, are as follows:

– THIEF (Athanasopoulos et al. 2017): THIEF method is based on the concept of temporal hierarchies for modeling and forecasting time series. This method takes a time series then generates all possible temporal aggregations for that series. For example, a quarterly time series is aggregated to biannual and annual time series. Next, the forecasts for

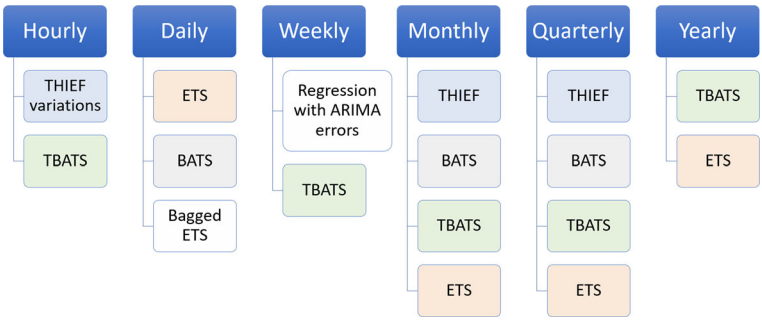


Fig. 2 Forecasting methods for each frequency

each aggregation level are reconciled from these different aggregation levels using the hierarchical reconciliation algorithm proposed in (Athanasopoulos et al. 2017). Since each temporal aggregation level contains different features of the time series, the resulting independent forecasts contain different information, which in turn results in producing accurate and robust forecasts (Athanasopoulos et al. 2017). THIEF method showed more accurate results over conventional forecasting methods due to the advantages of forecast combinations and temporal aggregation (Athanasopoulos et al. 2017). In the proposed model, for hourly data, the forecasts for each aggregation level were produced using ETS, auto ARIMA, and Naïve S. While ETS was used with monthly and quarterly data. These choices are based on trial and error.

- BATS and TBATS (Livera et al. 2011): These two forecasting methods are capable of modeling time series with complex seasonal patterns (Livera et al. 2011). The identifier BATS in these two methods is an acronym for the key features incorporated in these methods: Box-Cox transformation, ARMA errors, Trend and Seasonal components, while the initial T in TBATS connotes the use of the Trigonometric seasonality in the method (Livera et al. 2011). Different combinations are used with these two methods based on the key features incorporated in these methods. The selection of the best model is made using the selected information criterion (Livera et al. 2011). One of the differences between BATS and TBATS is that TBATS can be used to model non-integer seasonal frequencies because it relies on trigonometric functions (Livera et al. 2011). As it can be seen in Fig. 2, BATS and/or TBATS were used with each frequency.
- Bagged ETS (BaggedETS) (Bergmeir et al. 2016): BaggedETS method takes a time series then creates variations on it. To achieve that, each time series is Box-Cox-transformed, and then decomposed using STL (Seasonal and Trend decomposition using Loess) Cleveland et al. (1990) into trend, seasonal and remainder components. Following that, the remainder component is shuffled to obtain bootstrapped remainder series (Hyndman and Athanasopoulos 2018). After that, the trend and season components are combined with the bootstrapped remainder series, and then reversing the Box-Cox transformation in order to create variations on the original series (Hyndman and Athanasopoulos 2018). The final forecasts produced by the BaggedETS method are obtained by averaging the forecasts from each of the additional time series. According to (Petropoulos et al. 2018), bagging can deal with data, model, and parameter uncertainties. The baggedETS method was used in the proposed model to model and forecast the daily time series in the M4 competition.

– Regression with ARIMA errors: This method was inspired by the work presented in https://robjhyndman.com/hyndsight/forecasting-weekly-data/. Here, Fourier terms model the seasonal pattern (Hyndman and Athanasopoulos 2018). The short-term dynamics in a time series are handled by an ARMA error (Hyndman and Athanasopoulos 2018). The number of Fourier terms and the order of the ARIMA method in the proposed model was selected by minimizing the information criterion.

The proposed model consists of two phases, the computing weights phase and the forecasting phase. In the computing weights phase, which is shown in the flowchart depicted in Fig. 3, each time series provided by the organizers (y_i) was split into two sets, fitting set (u_i) and validation set (t_i). The number of observations in the fitting set equals its original length provided by the organizers $|y_i|$ minus the needed forecasts for that series (h_i), which are used as a validation set. The selected statistical methods were used to fit the fitting set based on the frequency. If any of the forecasts ($\hat{u}_{i,j}$) from these methods are less than 0, new forecasts are produced after fitting the logarithm of the fitting data. After that, the forecasts from the statistical methods are combined by columns in a matrix ($\hat{u}_{i,\text{comb}}$). Finally, by using the polynomial potential aggregation rule with different learning rates for each method (ML-Poly) (Gaillard and Goude 2015), a matrix of weights (W_i) is created with M rows and x columns, which represents the number of needed forecasts and the number of fitted methods, respectively.

As discussed above, in the computing weights phase, ML-Poly is used to compute the weights which will be used to combine the forecasts in the forecasting phase to produce the submitted forecasts. ML-Poly, which was introduced by Gaillard et al. (2014), has multiple learning rates which are added to the polynomially weighted average algorithm described by Cesa-Bianchi and Lugosi (2003). There are other aggregation rules that can be also applied to combine the forecasts such as exponentially weighted average forecaster, the fixed share forecaster, and the ridge regression forecaster (Gaillard and Goude 2015). However, according to Gaillard and Goude (2015), with ML-Poly it is possible to achieve the same level of performance as other aggregation rules and it is also much faster than the empirical tuning described by Devaine et al. (2013) and used for the other rules.

ML-Poly is used as implemented in opera package (Gaillard and Goude 2016) and the learning rates are calibrated using theoretical values (Gaillard and Goude 2016). The aggregation rule is similar to the gradient descent aggregation rule (i.e., loss.gradient = TRUE), and the square function is used as a loss function. A mixture object is created based on those settings. Following that, sequential predictions and updates of the mixture object are made based on the observations in the validation set (t_i). At this stage, the weights matrix is created and used in the forecasting phase.

In the forecasting phase, which is shown in Fig. 4, the time series provided by the organizers (y_i) was used to fit the selected forecasting methods. As in the computing weights phase, if any of the forecasts ($\hat{y}_{i,j}$) from these methods are less than 0, new forecasts are produced after fitting the logarithm of the fitting data. Following that, the forecasts from the statistical methods are combined by columns in a matrix ($\hat{y}_{i,\text{comb}}$). Finally, element-wise multiplication between the computed weights (W_i) and ($\hat{y}_{i,\text{comb}}$) was used to produce the submitted forecasts ($\hat{y}_i$).

The pseudo-code for the proposed model, which shows how these two phases were used to produce the submitted forecasts for all time series for any frequency, is shown in Algorithm 1.

It is good to note that the M4 dataset can be found in https://github.com/Mcompetitions/M4-methods/tree/master/Dataset, while the source code for the model is available in https://

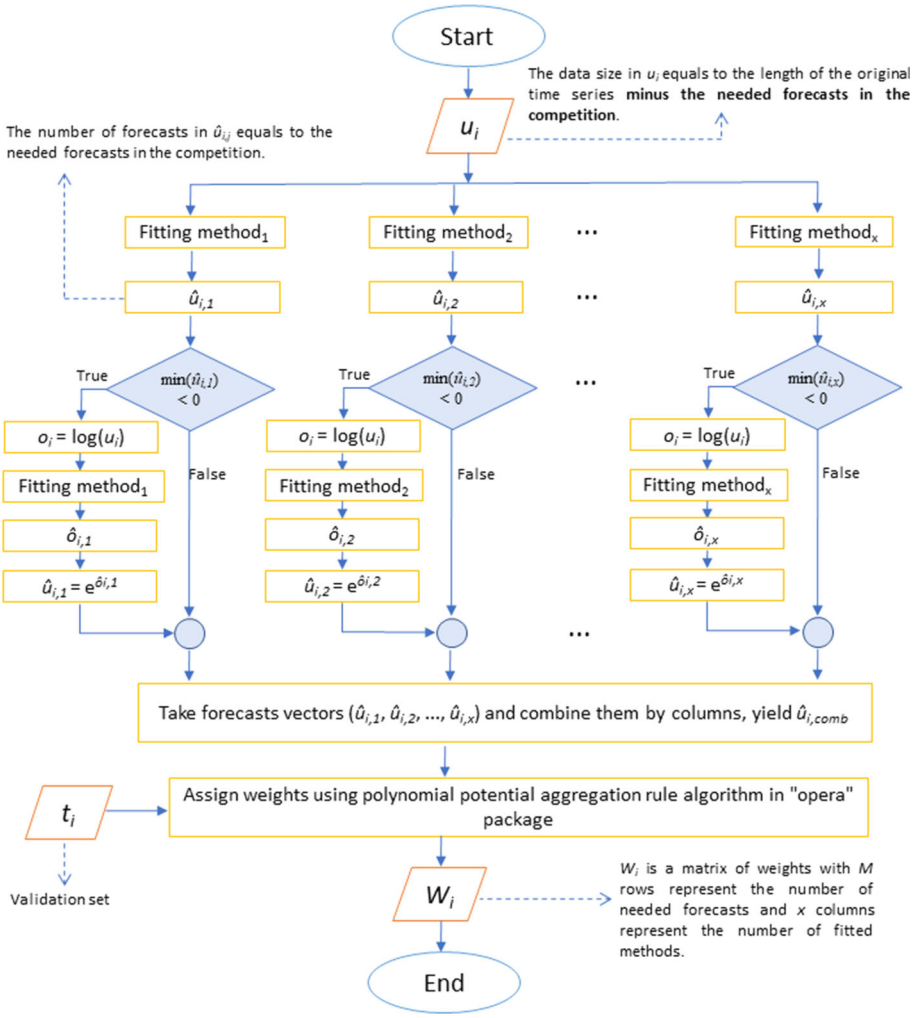


Fig. 3 Computing weights flowchart for one time series

Algorithm 1 Pseudo code for the proposed model.

Inputs:

- $y_1, y_2, ..., y_N$: N Time series provided by the organizers for the selected frequency. Each y_i is divided into fitting set (u_i) and validation set (t_i).

Outputs:

- $\hat{y}_1, \hat{y}_2, ..., \hat{y}_N$: Submitted forecasts.

for i = 1 to N **do**

Computing weights phase:

Follow the flowchart shown in Fig. 3 to compute the weights (W_i)

Forecasting phase:

Follow the flowchart shown in Fig. 4 to compute the submitted forecasts.

end for

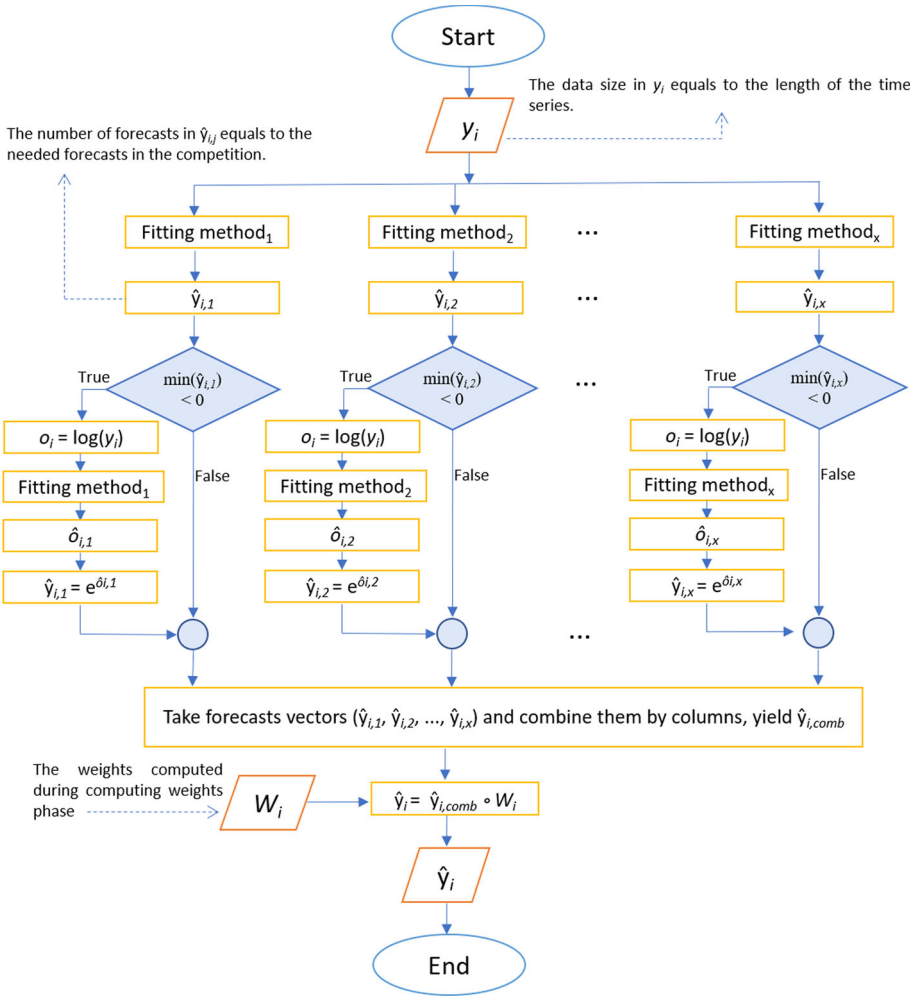


Fig. 4 The forecasting flowchart of the proposed model for one time series

github.com/Mcompetitions/M4-methods/tree/master/243%20-%20Waddah-Waheeb. The source codes with the settings used for the benchmarks and standards for comparison forecasting methods mentioned in Sect. 2 are found in https://github.com/Mcompetitions/M4-methods. In addition, the analysis provided in this paper is available in https://github.com/Waddah-Saeed/Frequency-based-ensemble-forecasting-model.

4 Results and discussions

4.1 Using M4 data

The forecasting performance comparison between the proposed model and other forecasting methods used by the organizers is shown in Table 2. It can be seen from the table that the

Table 2 Forecasting performance comparison (Makridakis et al. 2020)

Method	OWA_{Hourly}	OWA_{Daily}	OWA_{Weekly}	$OWA_{Monthly}$	$OWA_{Quarterly}$	OWA_{Yearly}
Statistical benchmarks						
Naïve 1	3.593	1	1	1.096	1.066	1
Naïve 2	1	1	1	1	1	1
Naïve S	0.627	1	1	1.147	1.153	1
SES	0.990	1	0.975	0.951	0.970	1.003
Holt	2.749	0.995	0.966	0.988	0.932	0.947
Damped	1.141	0.997	0.917	0.924	0.893	0.890
Comb	1.556	**0.978**	0.926	0.920	0.890	**0.867**
Theta	1.006	0.999	0.971	0.907	0.917	0.872
Standards for comparison						
ARIMA	0.577	1.044	0.932	**0.903**	0.898	0.892
ETS	0.852	0.996	0.931	0.915	0.891	0.903
Machine learning benchmarks						
MLP	0.921	3.509	3.608	1.749	1.684	1.288
RNN	1.036	1.930	1.755	1.587	1.508	1.308
Proposed model						
Proposed	**0.507**	0.999	**0.779**	0.927	**0.880**	0.880

Best results in boldface

proposed model shows better forecasting accuracy than all the models with hourly, weekly, and quarterly data. On the other hand, the Comb method performs better than the proposed model with daily and yearly data. With monthly data, the ARIMA method outperforms the proposed model. However, the overall performance of the proposed model with all frequencies is better than all the methods, as shown in Table 3. Since the best performance for the proposed model in the competition is achieved with weekly data, the improvement percentage to the compared forecasting methods with weekly data is reported in Table 3. It can be seen from Table 3 that the performance of machine learning benchmarks are not good. This could be attributed to the length of the time series as discussed in Sect. 2.

The reasons for the good forecasting performance for the proposed model can be attributed to the characteristics found in the forecasting methods used in the proposed model, which are combining forecasts and the usage of different forecasting methods for each frequency. Combining forecasts is found useful to improve the forecasting accuracy, especially when we are uncertain about which method is most accurate (Armstrong 2001), which was one of the challenges the author faced during the investigation. Utilizing various forecasting methods could be one of the reasons due to some advantages found in these methods, such as forecast combinations and the temporal aggregation found in THEIF method (Athanasopoulos et al. 2017), the ability to model complex seasonal time series as with BATS and TBATS (Livera et al. 2011), and the ability of BaggedETS in dealing with different uncertainties (Petropoulos et al. 2018).

Using the results of the computing weights phase, another analysis was conducted to find forecasting methods' contribution percentage to the calculation of the final forecasts. The percentages were calculated as follows. First, calculate how many times each method has a weight greater than zero in each produced forecast. Second, dividing the obtained results from the first step by the number of produced forecasts. The last step is to divide the results

Table 3 Improvement percentage to the compared forecasting methods

Method	OWA_{Total}	Improvement percentage (%)	OWA_{Weekly}	Improvement percentage (%)
Proposed	0.894	–	0.779	–
Statistical benchmarks				
Naïve 1	1.058	15.50	1	22.1
Naïve 2	1	10.60	1	22.1
Naïve S	1.078	17.07	1	22.1
SES	0.975	8.31	0.975	20.1
Holt	0.971	7.93	0.971	19.77
Damped	0.907	1.43	0.917	15.05
Comb	0.898	0.45	0.926	15.87
Theta	0.897	0.33	0.971	19.77
Standards for comparison				
ARIMA	0.903	1	0.932	16.42
ETS	0.908	1.54	0.931	16.33
Machine learning benchmarks				
MLP	1.642	45.55	3.608	78.41
RNN	1.482	39.68	1.755	55.61

Table 4 Forecasting methods contribution percentage to the calculation of the final forecasts

Frequency	Method	Contribution percentage (%)
Hourly	THIEF variations	50–69
	TBATS	46
Daily	BATS	56
	ETS	52
	Bagged ETS	51
Weekly	Regression with ARIMA errors	67
	TBATS	64
Monthly	THIEF	54
	BATS	53
	TBATS	53
	ETS	50
Quarterly	BATS	60
	TBATS	59
	ETS	58
	THIEF	56
Yearly	TBATS	67
	ETS	64

from the previous step by the number of time series for that frequency then multiply it by 100. These calculated percentages are shown in Table 4. The results revealed that the forecasting methods contributed differently in the final forecasts, and the majority of the forecasting methods contribution percentages fall in the range 50–69%.

4.2 Using EUR/USD exchange rate and Bitcoin closing price data

A further comparison was conducted between the proposed model and other forecasting methods to forecast the daily EUR/USD exchange rate and Bitcoin closing price time series. For EUR/USD time series, the one-step-ahead forecasting performance of the proposed model in root mean squared error (RMSE) is compared with results reported in (Waheeb et al. 2016, 2018; Waheeb and Ghazali 2019, 2020). This time series covers the period from 3 January 2005 to 31 December 2007. The data was divided into the following three sets: the fitting set consists of 469 data points, 156 data points used in the computing weights phase, and 156 data points for the out-of-sample set. As for the Bitcoin closing price time series, the multi-step ahead forecasting performance comparison is done with the work reported in (Waheeb et al. 2020). The period between November 24, 2018, and October 28, 2019, was used for the training, where the last 14 data points were used for the computing weights phase. The out-of-sample data are the next 14 days (i.e., from November 11, 2019, to November 24, 2019). Due to non-stationary in the time series, the first difference was used to make it stationary.

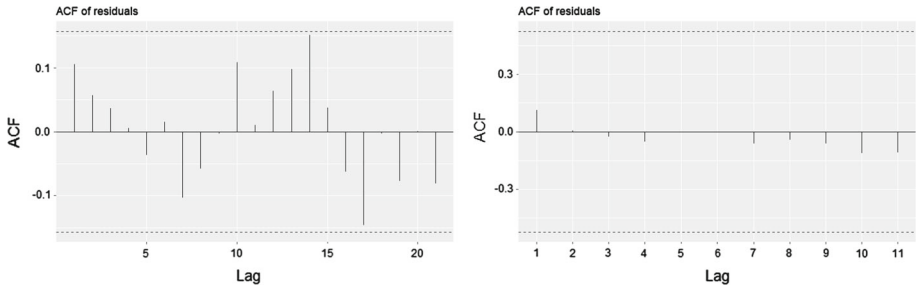
Instead of using the validation set only to compute the weights, both fitting and validation sets were used in another experiment. In addition, the mean and median combination of the point forecasts of the three methods used with the daily frequency were calculated.

Table 5 One-step ahead forecasting comparison on EUR/USD exchange rate time series

Method	RMSE	Improvement percentage (%)
Ridge polynomial neural network (RPNN) (Waheeb et al. 2016)	0.0072	59.72
Dynamic RPNN (Waheeb et al. 2018)	0.0068	55.88
Genetic algorithm + functional link neural network (Waheeb and Ghazali 2019)	0.0067	56.72
RPNN with error feedback (Waheeb et al. 2016)	0.0052	44.23
Mean combination of the point forecasts	0.0030	3.33
Proposed model (weights computed based on fitting + validation sets)	0.0030	3.33
Median combination of the point forecasts	0.0030	3.33
RPNN with error-output feedback (Waheeb and Ghazali 2020)	0.0029	–
Proposed model (weights computed based on validation set)	0.0029	–

Table 6 Multi-step ahead forecasting comparison on Bitcoin closing price time series

Method	sMAPE	Improvement percentage (%)
ARIMA (Waheeb et al. 2020)	10.21204	77.14
ETS (Waheeb et al. 2020)	10.1895	77.09
Theta (Waheeb et al. 2020)	10.15581	77.01
SES (Waheeb et al. 2020)	10.09934	76.89
Damped (Waheeb et al. 2020)	10.09788	76.88
Extreme learning machines (Waheeb et al. 2020)	8.678535	73.10
Multi-layer perceptron (Waheeb et al. 2020)	8.458624	72.40
Mean combination of the point forecasts	2.365618	1.32
Proposed model (weights computed based on fitting + validation sets)	2.365618	1.32
Proposed model (weights computed based on validation set)	2.334444	–
Median combination of the point forecasts	2.298935	− 1.52



(a) EUR/USD exchange rate. **(b)** Bitcoin closing price.

Fig. 5 ACF of the residuals from the proposed model

The comparison results are shown in Tables 5 and 6. It is apparent from the two tables that the proposed model outperforms many forecasting methods. Furthermore, the results from both tables show that using a validation set to compute the weights results in better forecasts than using the whole training set (i.e., fitting and validation sets). However, it still shows close results to the proposed model. The mean combination of the point forecasts, which uses equal weights, does not help to produce more accurate results. On the contrary, the point forecasts' median combination is better than the proposed model with the Bitcoin closing price time series. The autocorrelation function (ACF) plots of the residuals from the proposed model are shown in Fig. 5. The ACF plots show the lack of correlation which suggests that the forecasts are good.

4.3 Limitation

Aside from the improved forecasting accuracy, the time needed for the two phases is a limitation of the proposed solution. In the first phase, the fitting is done using the fitting set. Then, in the second phase, the fitting is done using the fitting and validation sets as one set. However, in some applications, a model that produces more accurate forecasts can be preferred than a model that can be trained faster.

4.4 Possible changes for better forecasting performance

As a final remark, it is worth commenting that many possible changes could have been made to improve the performance of the proposed model. First, use the median combination of the point forecasts. This can help reduce the time needed in the two phases and could help produce more accurate forecasts (e.g., Table 6). Second, avoid using the entire training data with long time series because the farthest observations might not be relevant. Third, instead of producing new forecasts after fitting the logarithm of the fitting data when the obtained forecasts are less than 0, consider using 0 as forecasts immediately. This can make the model much simpler and could enhance the forecasting performance.

5 Conclusions

This paper has described the model used in the M4 forecasting competition. It is a combination of statistical methods, namely auto-regressive integrated moving-average (auto ARIMA), exponential smoothing (ETS), bagged ETS, temporal hierarchical forecasting method, Box–Cox transformation, ARMA errors, Trend and Seasonal components (BATS), and Trigonometric seasonality BATS (TBATS). The model consists of two phases, the computing weights phase and the forecasting phase. The former is to compute the weights used in the latter stage to produce the forecasts. Based on the obtained results with M4 data, one can notice that the overall forecasting performance of the proposed model outperforms the 12 forecasting methods selected by the organizers. Furthermore, the proposed model is ranked 15th on average and 4th with the weekly time series among 49 submissions to the competition. A further comparison was conducted between the proposed model and other forecasting methods on forecasting EUR/USD exchange rate and Bitcoin closing price time series. It is apparent from the results that the proposed model performs better than many forecasting methods. Possible reasons for the good performance of the proposed model are combining forecasts and the usage of different forecasting methods for each frequency.

Funding Open access funding provided by University of Agder.

Declarations

Conflict of interest The author declares that he has no conflict of interest.

References

Armstrong JS (2001) Combining forecasts. Principles of forecasting. Springer, Berlin, pp 417–439

Assimakopoulos V, Nikolopoulos K (2000) The theta model: a decomposition approach to forecasting. Int J Forecast 16(4):521–530. https://doi.org/10.1016/S0169-2070(00)00066-2

Athanasopoulos G, Hyndman RJ, Kourentzes N, Petropoulos F (2017) Forecasting with temporal hierarchies. Eur J Oper Res 262(1):60–74. https://doi.org/10.1016/j.ejor.2017.02.046

Bergmeir C, Hyndman RJ, Benítez JM (2016) Bagging exponential smoothing methods using stl decomposition and box-cox transformation. Int J Forecast 32(2):303–312

Box GE, Jenkins GM, Reinsel GC, Ljung GM (2015) Time series analysis: forecasting and control. Wiley, Oxford

Cesa-Bianchi N, Lugosi G (2003) Potential-based algorithms in on-line prediction and game theory. Mach Learn 51(3):239–261

Cleveland RB, Cleveland WS, McRae JE, Terpenning I (1990) Stl: a seasonal-trend decomposition. J Off Stat 6(1):3–73

Devaine M, Gaillard P, Goude Y, Stoltz G (2013) Forecasting electricity consumption by aggregating specialized experts. Mach Learn 90(2):231–260

Gaillard P, Goude Y (2015) Forecasting electricity consumption by aggregating experts; how to design a good set of experts. In: Poggi JM, Brossat X, Antoniadis A (eds) Modeling and stochastic learning for forecasting in high dimensions. Springer, Berlin, pp 95–115

Gaillard P, Goude Y (2016) opera: Online Prediction by Expert Aggregation. http://pierre.gaillard.me/opera.html, r package version 1.0
Gaillard P, Stoltz G, Van Erven T (2014) A second-order bound with excess losses. In: Conference on learning theory, PMLR, pp 176–196
Gardner ES (1985) Exponential smoothing: the state of the art. J Forecast 4(1):1–28. https://doi.org/10.1002/for.3980040103
Gardner ES (2006) Exponential smoothing: the state of the art—part ii. Int J Forecast 22(4):637–666. https://doi.org/10.1016/j.ijforecast.2006.03.005
Hinton G, Srivastava N, Swersky K (2012) Neural networks for machine learning lecture 6a overview of mini-batch gradient descent. Cited on 14(8)
Hyndman RJ (2020) A brief history of forecasting competitions. Int J Forecast 36(1):7–14. https://doi.org/10.1016/j.ijforecast.2019.03.015
Hyndman RJ, Athanasopoulos G (2018) Forecasting: principles and practice. OTexts
Hyndman RJ, Khandakar Y (2008) Automatic time series forecasting: the forecast package for r. J Stat Softw 27(3):1–22. https://doi.org/10.18637/jss.v027.i03
Hyndman RJ, Koehler AB (2006) Another look at measures of forecast accuracy. Int J Forecast 22(4):679–688. https://doi.org/10.1016/j.ijforecast.2006.03.001
Hyndman R, Kourentzes N (2018) thief: Temporal HIErarchical Forecasting. http://pkg.robjhyndman.com/thief, r package version 0.3
Hyndman RJ, Koehler AB, Snyder RD, Grose S (2002) A state space framework for automatic forecasting using exponential smoothing methods. Int J Forecast 18(3):439–454. https://doi.org/10.1016/S0169-2070(01)00110-8
Hyndman R, Athanasopoulos G, Bergmeir C, Caceres G, Chhay L, O'Hara-Wild M, Petropoulos F, Razbash S, Wang E, Yasmeen F (2018) forecast: Forecasting functions for time series and linear models. https://pkg.robjhyndman.com/forecast/, r package version 8.3
Kingma DP, Ba J (2014) Adam: a method for stochastic optimization. arXiv:1412.6980
Livera AMD, Hyndman RJ, Snyder RD (2011) Forecasting time series with complex seasonal patterns using exponential smoothing. J Am Stat Assoc 106(496):1513–1527. https://doi.org/10.1198/jasa.2011.tm09771
Makridakis S (1993) Accuracy measures: theoretical and practical concerns. Int J Forecast 9(4):527–529. https://doi.org/10.1016/0169-2070(93)90079-3
Makridakis S, Wheelwright S, Hyndman RJ (1998) Forecasting: methods and applications. Wiley, Oxford
Makridakis S, Spiliotis E, Assimakopoulos V (2020) The m4 competition: 100,000 time series and 61 forecasting methods. Int J Forecast 36(1):54–74. https://doi.org/10.1016/j.ijforecast.2019.04.014
Petropoulos F, Hyndman RJ, Bergmeir C (2018) Exploring the sources of uncertainty: why does bagging for time series forecasting work? Eur J Oper Res 268(2):545–554. https://doi.org/10.1016/j.ejor.2018.01.045
Rumelhart DE, Hinton GE, Williams RJ (1986) Learning representations by back-propagating errors. Nature 323(6088):533–536
Team RC et al (2018) R: a language and environment for statistical computing
Waheeb W, Ghazali R (2019) A new genetically optimized tensor product functional link neural network: an application to the daily exchange rate forecasting. Evol Intel 12(4):593–608
Waheeb W, Ghazali R (2020) A novel error-output recurrent neural network model for time series forecasting. Neural Comput Appl 32:9621–9647
Waheeb W, Ghazali R, Herawan T (2016) Ridge polynomial neural network with error feedback for time series forecasting. PLoS One 11(12):1–34. https://doi.org/10.1371/journal.pone.0167248
Waheeb W, Ghazali R, Hussain AJ (2018) Dynamic ridge polynomial neural network with lyapunov function for time series forecasting. Appl Intell 48(7):1721–1738
Waheeb W, Shah H, Jabreel M, Puig D (2020) Bitcoin price forecasting: A comparative study between statistical and machine learning methods. In: 2020 2nd International Conference on Computer and Information Sciences (ICCIS), pp 1–5, https://doi.org/10.1109/ICCIS49240.2020.9257664