HERMES: Hybrid Error-corrector Model with inclusion of External Signals for nonstationary fashion time series

Anonymous authors
Paper under double-blind review

Abstract

Developing models and algorithms to draw causal inference for time series is a long standing statistical problem. It is crucial for many applications, in particular for fashion or retail industries, to make optimal inventory decisions and avoid massive wastes. By tracking thousands of fashion trends on social media with state-of-the-art computer vision approaches, we propose a new model for fashion time series forecasting. Our contribution is twofold. We first provide publicly[1] an appealing fashion dataset gathering 10000 weekly fashion time series. As influence dynamics are the key of emerging trend detection, we associate with each time series an external weak signal representing behaviours of influencers. Secondly, to leverage such a complex and rich dataset, we propose a new hybrid forecasting model[1]. Our approach combines per-time-series parametric models with seasonal components and a global recurrent neural network to include sporadic external signals. This hybrid model provides state-of-the-art results on the proposed fashion dataset, on the weekly time series of the M4 competition (Makridakis et al., 2018), and illustrates the benefit of the contribution of external weak signals.

1 Introduction

Multivariate time series forecasting is a widespread statistical problem with many applications, see for instance Särkkä (2013); Douc et al. (2014); Zucchini et al. (2017) and the numerous references therein. Parametric generative models provide explainable predictions with statistical guarantees based on a precise modeling of the predictive distributions of new data based on a record of past observations. Calibrating these models, for instance using maximum likelihood inference, often requires a fair amount of tuning to design a time series-specific model to provide accurate forecasts and sharp confidence intervals. Depending on the use case, statistical properties of the signal and the available data, many families of models have been proposed for time series. The exponential smoothing model (Brown & Meyer, 1961), the Trigonometric Box-Cox transform, ARMA errors, Trend, and Seasonal components model (TBATS) (Livera et al., 2011), or the ARIMA with the Box-Jenkins approach (Box et al., 2015) are for instance very popular parametric generative models. Hidden Markov models (HMM) are also widespread and presuppose that available observations are defined using missing data describing the dynamical system. This hidden state is assumed to be a Markov chain such that at each time step the received observation is a random function of the corresponding latent data. Although hidden states are modeled as a Markov chain, the observations arising therefrom have a complex statistical structure. In various applications where signals exhibit non-stationarities such as trends and seasonality, classical HMM are not adapted. However, Touron (2017) recently proposed seasonal HMM, assuming that transition probabilities between the states, as well as the emission distributions, are not constant in time but evolve in a periodic manner. Strong consistency results were established in Touron (2019) and Expectation Maximization based numerical experiments were proposed. Although these works provide promising results, HMM are computationally expensive to train and are not yet well studied for seasonal sequences with thousands of components.

In many fields, single or few time series have become thousands of sequences with various statistical properties. In this new context, classical time series specific statistical models show limitations when dealing with

[1] `https://anonymous.4open.science/r/HERMES-703F/`

numerous heterogeneous data. Recurrent neural networks and recent sequence to sequence deep learning architectures offer very appealing numerical alternatives thanks to their capability of leveraging any kind of heterogeneous multivariate data, see for instance Hochreiter & Schmidhuber (1997); Vaswani et al. (2017); Siami-Namini et al. (2018); Li et al. (2019); Lim et al. (2019); Salinas et al. (2020). The DeepAR model proposed in Salinas et al. (2020) provides a global model from many time series based on a multi-layer recurrent neural network with LSTM cells. More recently, applications using the Transformer model have been proposed (Li et al., 2019). The Temporal Fusion Transformers (TFT) approach is a direct alternative to the DeepAR model (Lim et al., 2019). Unfortunately, all these solutions suffer from two main weaknesses. Firstly, many of them are black-boxes as the final forecast usually does not come with a statistical guarantee although a few recent works focused on measuring uncertainty in recurrent neural networks, see Martin et al. (2021). Secondly, without a fine preprocessing and well chosen hyperparameters, these methods may lead to poor results and be outperformed by traditional statistical models, see Makridakis et al. (2018).

In this paper, we consider an emerging time series forecasting application referred to as *fashion trends prediction*. In fashion and retails industries, accurately anticipating consumers' needs is vital and wrong decisions can lead to massive wastes. With the explosion of social network and the recent advances in image recognition, it is possible to translate the visibility of fashion items on social media over time into time series. Consequently, models and algorithms can be trained to accurately anticipate and predict consumer behaviour. In Ma et al. (2020), a dataset is provided using social media pictures and an image recognition framework to detect several clothes: 2000 fashion time series are proposed with a weekly seasonality. However, only 3 years of historical data is available (144 data points) that may not be sufficient for some statistical approaches. In Ma et al. (2020), another dataset is presented gathering 8000 fashion sequences with an historical available data increased to 5 years. Nevertheless, only 120 values are available for each fashion time series and the overall volume remains low for a large part of the sequences resulting in a lot of noise and no clear patterns. In this paper, we propose a new fashion dataset overcoming the weaknesses of the two previous ones. Based on cutting-edge image recognition algorithms (Ren et al., 2015; Chollet, 2017), we built a large fashion dataset containing 10000 weekly sequences of fashion trends on social media with 5 years of historical data from 01-01-2015 to 30-12-2019. This dataset has very appealing properties: all time series have the same length (261 data points), there is no missing value and there is no sparse time series even for niche trends. Concerning fashion dynamics, some of them appear to be really volatile with nonlinear changes of dynamics resulting from the emergence of new tendencies. In this context, understanding early signals of the apparition of a trend is one of the key to accurately forecast the future of the fashion. Consequently, the originality of our dataset comes from the fact that additional external weak signals are introduced. With our fashion expertise, we detected several groups of highly influential fashion users. Analyzing their specific behaviours on social media, we associate with each time series an external weak signal representing the same fashion trends on a sub-category of users. They are called weak signals because they are often alerts or events that are too sparse, or too incomplete to allow on their own an accurate estimation of their impact on the prediction of the target signal. Exploring this new way of representing fashion, we aim at designing a model able to deal with such a large dataset, leveraging complex external weak signals and finally providing the most accurate forecasts.

Recurrent neural networks are appealing to tackle our forecasting problem due to their capability of leveraging external data. Recently, hybrid models combining deep neural network (DNN) architectures with widespread statistical models to deal with seasonality and trends have been proposed, see for instance Zhang (2003); Jianwei et al. (2019); Bandara et al. (2020). The approach providing the most striking results was proposed in Smyl (2020) in the context of the M4 forecasting competition (Makridakis et al., 2020). Given a large dataset, a per-time-series multiplicative exponential smoothing model was introduced to estimate simple but fundamental components for each time series and compute a first prediction. Then a global recurrent neural network was trained on the entire dataset to correct errors of the previous exponential smoothing models.

Following this work, we present in this paper HERMES, a new hybrid recurrent model for time series forecasting with inclusion of external signals. This new architecture is decomposed into two parts: local predictors and a global corrector. First, a per-time-series parametric statistical model is trained on each sequence. Then, a global recurrent neural network is trained to evaluate and correct the forecast weaknesses of the first collection of models. The external weak signals reveal the real potential of the hybrid approach:

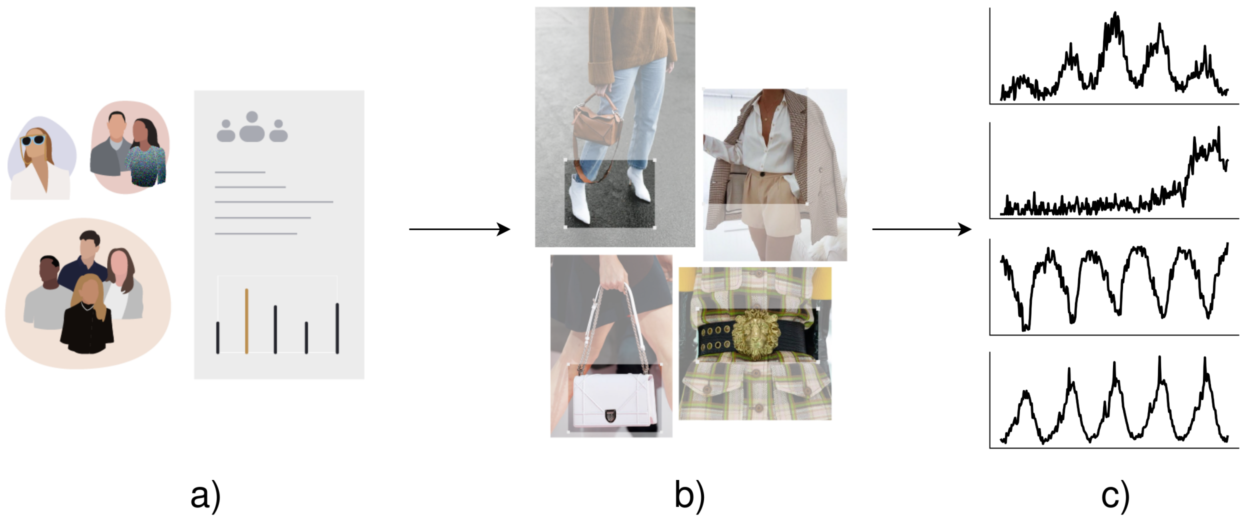

Figure 1: From social media to fashion time series. a) A complete image dataset of 150 millions of pictures is collected from social media users localized on 5 strategic markets. b) A visual recognition pipeline is applied on images. Global fashion items are detected with a collection of fine-grain attributes. c) Results are aggregated by fashion trend over time and normalized in order to remove social media bias.

a global neural network, able to leverage large amounts of heterogeneous data, deal with any kind of external weak signals, learn context and finally correct weaknesses and errors of parametric models.

The paper is organized as follows. Section 2 presents the new fashion dataset provided with this article. Then, the proposed forecasting approach HERMES is presented in Section 3. Section 4 describes the model results and comparisons with several benchmarks on the 2 different use cases: the fashion dataset and the M4 competition weekly dataset. Finally, a general conclusion and some research perspectives are given in Section 5.

2 From social media to fashion time series

2.1 Translate fashion to data

Social media have appeared as an impressive data source to follow the evolution of fashion over the time. Looking at a specific trend, social media can provide where and when this trend had been worn at first and how it spread all over the world. To do it automatically for thousands of trends, the following methodology was introduced. In the first place, a complete image dataset of 150 millions pictures is collected from different social media such as Instagram or Weibo. We targeted 5 strategic markets for the retail industry using posts localisation: the United States, Europe, Japan, Brazil and China. The second step consists in creating a powerful visual recognition framework to be able to detect clothes details on pictures like the type of clothing, the form, the size, the color, the texture, etc. To do so, the following framework is designed.

1. First, an object detection model is trained to detect the position, the size and the general type of possible multiple fashion items on a picture. This localization model is based on the Faster-RCNN architecture introduced in Ren et al. (2015). Starting from weights trained on MS-COCO (Lin et al., 2014), the model is fine-tuned with our data with a standard setup following the original paper.

2. Additionally, several visual recognition models are trained at classifying a rich collection of 350 fashion details. We train one classifier for each category of fashion item: one for pants, another for tops, a third for shoes, etc. These models are all based on the Xception architecture introduced in Chollet (2017). So as to trained them, large amount of social media pictures (between 200k and 800k training images depending on the category) have been manually tagged to constitute meaningful

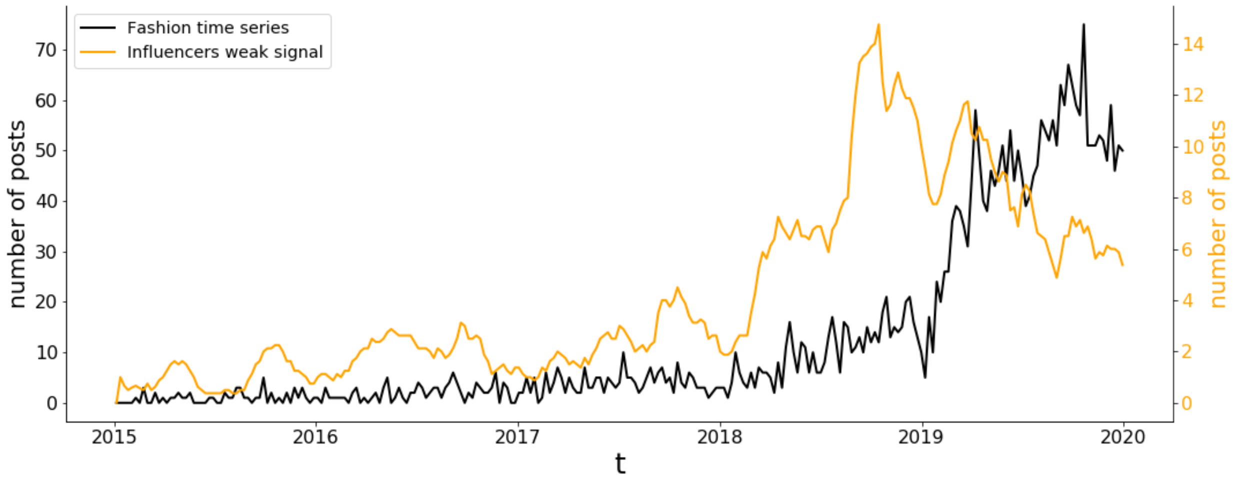

Figure 2: A shoes trend of the fashion dataset. In black the main signal and in orange its associated *fashion-forward* weak signal. The sudden explosion of the influencers signal at the end of 2018 announces the future burst of the trend in the mass market.

training datasets depending on the classification task. Architectures are first initialized with public weights trained on ImageNet (Russakovsky et al., 2014) and then fine tuned on the manually labeled dataset corresponding to their task.

At inference time, we first apply the localisation model which predicts boxes of generic fashion items (tops, pants, shoes, dresses, etc.) for each image. Then, each fashion item is cropped from its full image, resized to the classifiers' input size (299×299 px) and fed into the related classifier: a top will be fed into the model trained on tops, etc. We obtain for each image a set of boxes, associated with a general category and a set of fine-grain attributes describing this object. As a final step, fashion experts aggregate those attributes to define relevant trends for the fashion and retails industry.

The 150 million of social media pictures are analyzed with this visual recognition pipeline. Out of those images, we detected clothes in 96 millions posts making the final dataset used in this paper. We aggregate results by fashion trend definition over the time and thousands of trends are finally translated from social media to time series. We note $y^{c,g,m,i}$ the final raw sequence representing the fashion trend i of the cloth type c for the gender g on market m. At each time t, $y_t^{c,g,m,i}$ represents the number of posted pictures in the market m during the week t where computer vision algorithms detected the fashion trend i of the cloth type c for the gender g. As an illustration, an example of fashion time series is given in Figure 2.

2.2 Removing social media bias

Due to the increasing use of social media and continuous changes of users' behaviours, a normalization step is applied to the raw sequences $y^{c,g,m,i}$ in order to remove bias. Thus, we define the following normalizing signal $\tilde{y}^{c,g,m}$. This signal represents the global sequence of the cloth type c for the gender g on market m (e.g the evolution of the skirts in general for female in Europe). With the R package `stats`, the Seasonal-Trend decomposition using LOESS (Cleveland et al., 1990) is used to remove the seasonal component of $\tilde{y}^{c,g,m}$. The resulting deseasonalized signal is called $\bar{y}^{c,g,m}$. Finally, for any fashion trend i, the following normalized sequence is defined for all $0 \leq t \leq T$:

$$y_t^i = \frac{y_t^{c,g,m,i}}{\bar{y}_t^{c,g,m}} \,, \tag{1}$$

where T denotes the number of available time steps. The time series $y^{c,g,m,i}$ is divided by the deseasonalized signal $\bar{y}^{c,g,m}$ and not $\tilde{y}^{c,g,m}$ in order to avoid removing the seasonality of all the fashion trend sequences. With this normalizing step, most of the social media bias is removed and the final normalized sequences are

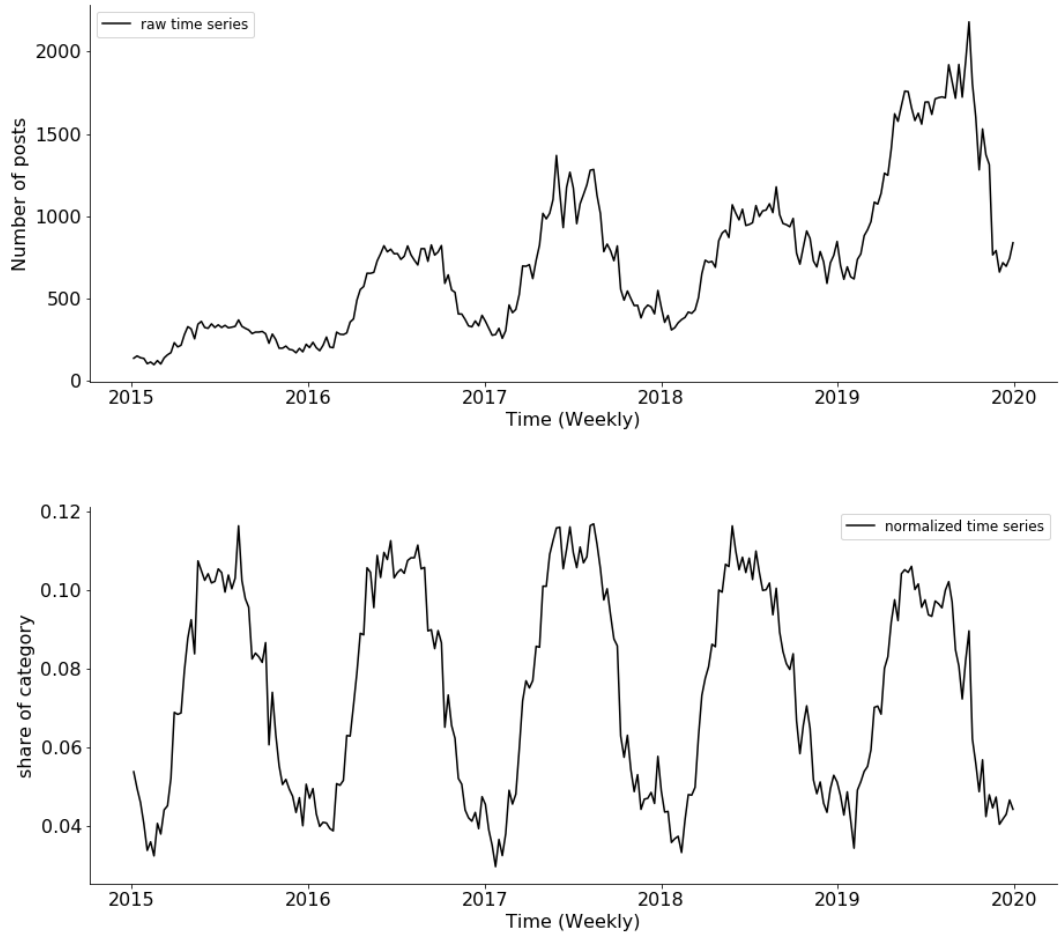

Figure 3: Example of difference between the raw sequence and the normalized one for the Jersey top fashion trend for females in China. In this example, we normalize by the deseasonalized global top fashion trend for females in China. (Top) Time series representing the raw signal of the Jersey top fashion trend for females in China. (Bottom) Time series representing the normalized signal of the Jersey top fashion trend for females in China.

expressed in share of category. As an illustration, an example of the normalization process is displayed in Figure 3. The raw Jersey Top trend for females in China is divided by the deseasonalized global Top trend for females in China.

2.3 Weak signal

In theoretical fashion dynamics (Rogers, 1962), different categories of adopters follow a trend in succession, resulting in several adoption waves. So as to catch the early signal of the emergence of a trend, 6000 social media influencers were selectioned by hand by fashion experts. Aggregating them, a specific "fashion-oriented" panel is created. With the same methodology as for the main panel described in Section 2.1 and Section 2.2, a normalized time series representing each fashion trend on this specific population is created. We named *fashion-forwards* this weak signal. For all fashion sequence $\{y_t^i\}_{1 \leq t \leq T}$, let $\{y_t^{f,i}\}_{1 \leq t \leq T}$ be the normalized sequence representing the behaviours of influencers regarding the fashion trend i. As we want to detect shifts between the main signal and the fashion forward signal, the following input is computed for

Table 1: Fashion time series overview. For each couple geozone/category, the table gives the number of trends (Female/Male).

	Top	Pants	Short	Skirt	Dress	Coat	Shoes	Color	Texture
United States	411/208	149/112	47/22	29/-	20/-	208/151	293/86	38/44	85/81
Europe	409/228	134/114	48/21	28/-	20/-	211/159	303/78	41/42	87/74
Japan	403/218	136/107	49/31	28/-	23/-	185/149	311/78	46/42	92/65
China	424/202	147/114	46/29	27/-	27/-	178/161	310/78	41/47	88/77
Brazil	431/222	134/117	49/27	30/-	28/-	203/152	311/76	48/41	107/84
Total	2078/1078	700/564	239/130	142/-	118/-	985/772	1528/396	214/216	459/381

the hybrid model: for all $t \in \{1, \dots, T\}$ and any fashion trend i,

$$w_t^{f,i} = \frac{y_t^{f,i}}{y_t^{f,i} + y_t^i} \,.$$

where T denotes the number of available time steps. Values close to 0.5 indicate a similar behaviour between the influencers panel and the general panel. For instance, an emerging fashion shoes trend with its *fashion-forwards* weak signal is represented in Figure 2.

2.4 Fashion dataset

With this paper, we provide publicly[1] a sample of 10000 normalized fashion trends for men and women, over 9 different categories and 5 different markets. Each sequence has 261 time steps, from 2015-01-05 to 2019-12-31 with weekly values and no missing values. This collection of 10000 fashion trends was selected in order to represent finely the issues faced by the fashion industry. For instance, some sequences show complex behaviours with sudden changes, referred to as emerging or declining trends. A central point of this work is to accurately detect and forecast such trends. In addition, each fashion time series is linked with its associated normalized fashion forward signal as presented in the section above. An overview of the dataset can be found in Table 1.

3 HERMES: a new hybrid model for time series forecasting

We introduce a new hybrid approach for time series forecasting composed of two parts: a collection of per-time-series parametric models, and a global error-corrector neural network train on all time series. Per-time-series parametric models are used in particular to learn local behaviours and to normalize sequences by removing trends and seasonality. Then, a recurrent neural network driven by the weak signals is trained to correct these per-time-series models.

Consider $N \geqslant 1$ time series. For all $1 \leqslant n \leqslant N$ and $1 \leqslant t \leqslant T$, let y_t^n be the value of the n-th sequence at time t and $\mathbf{y}^n = \{y_t^n\}_{1 \leqslant t \leqslant T}$ be all the values of this sequence. The objective of this paper is to propose a model to forecast all time series in a given time frame $h \in \mathbb{N}$, i.e. we aim at sampling $\{y_{T+1:T+h}^n\}_{1 \leqslant n \leqslant N}$ based on $\{y_{1:T}^n\}_{1 \leqslant n \leqslant N}$.

3.1 Per-time-series predictors

For all $1 \leqslant n \leqslant N$, we note $f^n(.;\theta_{predictor}^n)$ the n-th parametric model of the n-th sequence where $\theta_{predictor}^n$ are unknown parameters. Given the sequences $\{y_{1:T}^n\}_{1 \leqslant n \leqslant N}$ and the estimated parameters $\{\theta_{predictor}^n\}_{1 \leqslant n \leqslant N}$, the time-series-specific forecasts $\{\widehat{y}_{T+1:T+h|T}^{pred,n}\}_{1 \leqslant n \leqslant N}$ are, for all $n \in \{1, \dots, N\}$, for all $i \in \{1, \dots, h\}$,

$$\widehat{y}_{T+i|T}^{pred,n} = f^n(y_{1:T}^n; \theta_{predictor}^n)_i \,. \tag{2}$$

During the M4 competition, the hybrid model of Smyl (2020) was based on a multiplicative exponential smoothing model as the time-series-specific predictor. However, on sporadic time series, this choice leads to

[1] https://anonymous.4open.science/r/HERMES-703F/

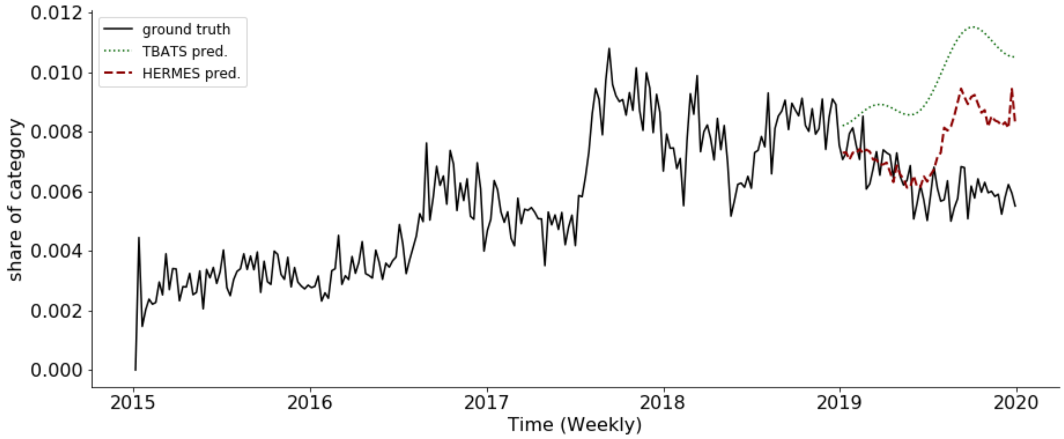

Figure 4: Hermes forecast example on a time series representing the vertical stipes texture fashion trend for females in Brazil. In green the prediction of the TBATS per-time-series predictors. In red the final forecast of our HERMES hybrid model.

poor results and instability. In this paper, a more general framework able to deal with any kind of per-time-series models is provided. In Section 4, two versions of our framework are proposed. The first one is based on an exponential smoothing as a reference similar to the baseline Smyl (2020) and the second one uses a TBATS model (Livera et al., 2011) which provides better results as this parametric model includes Fourier representations with time varying coefficients, and ARMA error correction.

For non stationary time series, huge changes of behaviours are not always predictable using the past of the sequence. In some cases, these changes depend on external variables not considered by univariate parametric models. The difficulty is that the exact influence of external variables on the main signal is mostly unknown. This motivates the introduction of a global RNN trained on all time series and able to consider and leverage external signals.

3.2 Error-corrector recurrent model

The second part of the model is a global RNN, trained on all the N sequences to correct the weaknesses of the first per-time-series parametric models. This task requires a thorough data pre-processing as recurrent neural networks training is highly sensitive to the scale of the data and requires well-designed inputs.

Let $w \in \mathbb{N}$ be the window size, usually this window is proportional to the forecast horizon $w \propto h$. The RNN input is defined as the following normalized, deseasonalized and rescaled sequence $\mathbf{z}_T^n = \{z^n_{T-w+i|T}\}_{1 \leqslant i \leqslant w}$: for all $1 \leqslant n \leqslant N$ and $1 \leqslant i \leqslant w$,

$$z^{n,T}_{T-w+i|T} := \frac{y^n_{T-w+i} - \widehat{y}^{pred,n}_{T+k|T}}{\bar{y}^n_T}\,, \quad \bar{y}^n_T = \frac{1}{w}\sum_{i=1}^{w} y^n_{T-w+i}\,.$$

where $k = i - h\lfloor i/h \rfloor$ with $\lfloor . \rfloor$ the floor function. With the numerator part $y^n_{T-w+i} - \widehat{y}^{pred,n}_{T+k|T}$, the per-time-series prediction is included in the RNN input and all the fundamental patterns already learned by this first predictor are removed from the time series. Then the denominator $\bar{y}^n_T$ is use to rescaled all input at the same level as the time series can have different scales. Another option could have been to divide directly y^n_{T-w+i} by $\widehat{y}^{pred,n}_{T+k|T}$ but with time series hitting 0, this option is not valid. Let $\textsc{rnn}(.;\theta_{corrector})$ be the recurrent neural network model where $\theta_{corrector}$ are unknown parameters. Given the RNN input sequences $\{\mathbf{z}^n_T\}_{1 \leqslant n \leqslant N}$ and the global RNN estimated parameters $\theta_{corrector}$, the error-corrector predictions $\{\widehat{y}^{corr,n}_{T+1:T+h|T}\}_{1 \leqslant n \leqslant N}$ are, for all $n \in \{1, \ldots, N\}$, for all $i \in \{1, \ldots, h\}$,

$$\widehat{y}^{corr,n}_{T+i|T} = \text{RNN}(\mathbf{z}^n_T; \theta_{corrector})_i \cdot \bar{y}^n_T \ .$$

Thus, if no external signals are available, the final hermes forecast is, for all $1 \leqslant n \leqslant N$ and all $i \in \{1, \ldots, h\}$,

$$\begin{aligned} \widehat{y}^{n}_{T+i|T} &= \widehat{y}^{pred,n}_{T+i|T} + \widehat{y}^{corr,n}_{T+i|T} \\ &= f^n(y^n_{1:T}; \theta^n_{predictor})_i + \text{RNN}(\mathbf{z}^n_T; \theta_{corrector})_i \cdot \bar{y}^n_T \,. \end{aligned} \tag{3}$$

3.3 Weak signal

In addition to the N target time series, $K \times N$ external sequences indexed from 0 to T are now considered. For all $1 \leqslant n \leqslant N$, $1 \leqslant k \leqslant K$ and $1 \leqslant t \leqslant T$, let $w^{n,k}_t$ be the value of the k-th external sequence at time t associated with the sequence $\mathbf{y}^n$. Let $\mathbf{w}^n = \{\{w^{n,k}_t\}_{1 \leqslant t \leqslant T}\}_{1 \leqslant k \leqslant K}$ be all the values of the external signals. In addition, let $\mathbf{w}^n_T = \{\{w^{n,k}_{T-w+i}\}_{1 \leqslant i \leqslant w}\}_{1 \leqslant k \leqslant K}$ be only the last w terms of the external sequences. Concatenating $\mathbf{z}^n_T$ and $\mathbf{w}^n_T$, a new input for the RNN is defined:

$$\mathbf{x}^n_T = \{x^n_{T-w+i|T}\}_{1 \leqslant i \leqslant w} = \{z^n_{T-w+i|T}, w^{n,1}_{T-w+i}, ..., w^{n,K}_{T-w+i}\}_{1 \leqslant i \leqslant w} \,.$$

Finally, for all $1 \leqslant n \leqslant N$ and for all $i \in \{1, \ldots, h\}$ the final prediction becomes:

$$\begin{aligned} \widehat{y}^{n}_{T+i|T} &= \widehat{y}^{pred,n}_{T+i|T} + \widehat{y}^{corr,n}_{T+i|T} \\ &= f^n(y^n_{1:T}; \theta^n_{predictor})_i + \text{RNN}(\mathbf{x}^n_T; \theta_{corrector})_i \cdot \bar{y}^n_T \,. \end{aligned} \tag{4}$$

An illustration of the proposed model is displayed in Figure 5 and a first forecast example is given in Figure 4..

4 Experimental results

4.1 Training

The dataset is split into three blocks, *train*, *eval* and *test* sets. The 3 first years are used as the *train* set, the 4th year is kept for the *eval* set and the *test* set is made of the last year. The hybrid model is trained to compute a one-year ahead prediction, h equal to 52, and the window size w is fixed at 104. Using the two first years of the *train* set, a first per-time-series parametric model for each time series is fitted. With the resulting collection of local models, a forecast of the third year is computed for each sequence. Corrector inputs are finally computed and the RNN is trained at correcting this first collection of third-year forecasts. For the *eval* set, per-time-series predictors are fitted a second time using the three first years and forecasts of the fourth year are computed. The *eval* set is used during training to control the learning of the RNN model and prevent overfitting. The per-time-series predictors are fitted a last time for the *test* set using the four first years. The final accuracy measures of all our models are computed on this *test* set. As an illustration, an example of our split is shown in Figure 6.

For the first parametric per-time-series models, existing Python libraries named `statsmodels` and `tbats` are used to estimate the different parameters $\theta^n_{predictor}$. Depending of the choice of local parametric models, two versions of HERMES are proposed. The first one uses as predictors an additive exponential smoothing model as a reference close to Smyl (2020). The second one uses the TBATS model of Livera et al. (2011) and achieves the highest accuracy results on the fashion dataset. The neural network architecture is composed of 3 LSTM layers of shape 50 and a final Dense layer to provide the correct output dimension. A classical Adam optimizer is used with a learning rate and a batch size set using a grid search. The loss function is defined as follows:

$$\ell(y^n_{T+1:T+h}, \widehat{y}^n_{T+1:T+h|T}) = \frac{1}{\bar{y}^n_T} \sum_{i=1}^{h} |y^n_{T+i} - \widehat{y}^n_{T+i|T}| \,.$$

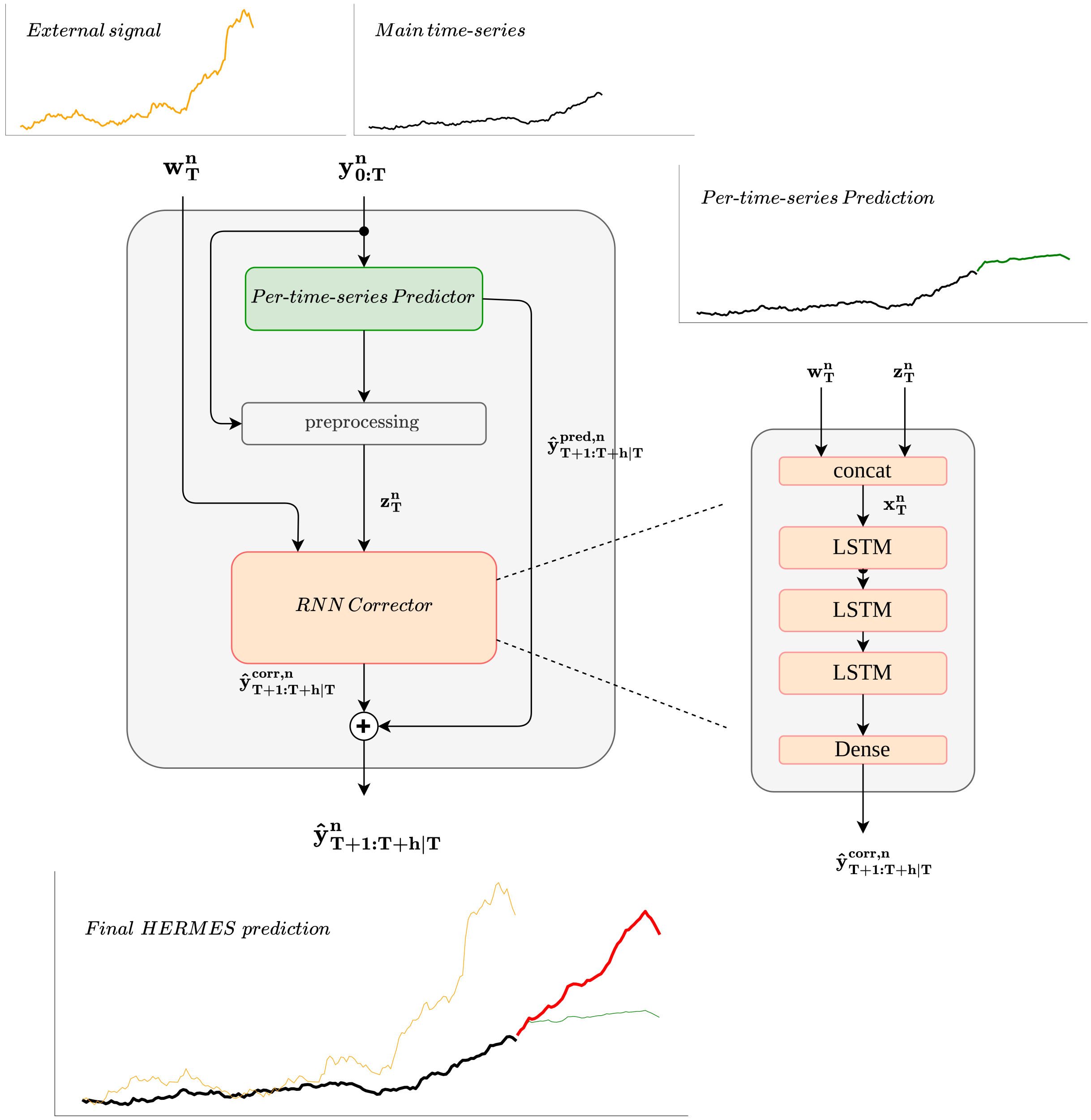

Figure 5: Architecture of the hybrid model with weak signals. The proposed framework can be decomposed in 5 steps: i) provide a time series. ii) (a) fit a first statistical model with the provided time series, (b) compute a first prediction and (c) preprocess the time series for the Global RNN. iii) If available, external signals can be added as part of the RNN input. iv) With a pre-trained RNN, compute a correction of the first statistical prediction. v) Compute the final forecast by adding the first time series prediction and the RNN correction.

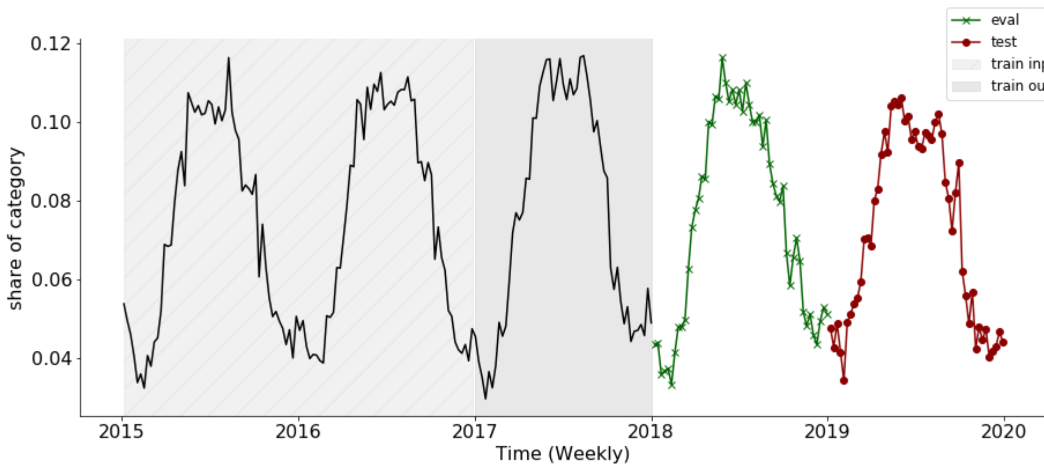

Figure 6: Temporal split for our training process. The three first years define our training set. The fourth year is used as our eval set and the final year is reserved for the test set.

This choice of L_1 loss function is motivated by its robustness to outliers which accounts for some time series in the fashion industry with very specific behaviours. The loss and previous parameters are all set with a complete grid search. See B.1 for additional results concerning the loss function choice and B.2 for a complete grid search example. The code is developed in Python using the Tensorflow library and publicly available[1]. It allows the use of GPU to speed up the training process.

4.2 Benchmarks, hybrid models and Metrics

As benchmarks, several widespread statistical methods and deep learning approaches were selected. Using the R package `forecast` and the Python packages `statsmodels`, `tbats`, for each time series, predictions are computed with the following methods: *snaive*, *ets*, *stlm*, *thetam*, *tbats* and *auto.arima*. The forecast of the *snaive* method is only the repetition of the last past period. The *ets* model is an additive exponential smoothing with a level component and a seasonal component. The *stlm* approach uses a multiplicative decomposition and models the seasonally adjusted time series with an exponential smoothing model. The *Thetam* model decomposes the original signal in θ-lines, predicts each one separately and recomposes them to produce the final forecast and *tbats* uses a trigonometrical seasonality. Finally, *auto.arima* is the R implementation of the ARIMA model with an automatic selection of the best parameters. A complete description and references for these models can be found in Hyndman et al. (2020). As a deep learning approach, a full LSTM (*lstm*) neural network composed of 3 LSTM layers of shape 50 and a final Dense layer of shape 52 is considered. Two versions of HERMES are proposed. They are called respectively *hermes-ets* and *hermes-tbats* according to the per-time-series model choice. Moreover, two versions with the inclusion of the weak signals (ws) are proposed. They are referred to as *hermes-ets-ws* and *hermes-tbats-ws*. In order to provide a fair comparison, a *lstm* with the weak signals named *lstm-ws* is trained.

To compare the different methods, we use the Mean Absolute Scaled Error (MASE) for seasonal time series. As our sequences have completely different scales, from 10^{-5} to 10^{-1}, this metric was chosen to compute a fair error measure, independent of the scale of the sequence and suited for our seasonal fashion time series. The MASE metric is defined as follows, with T the length of the time series, m the seasonal period and h the horizon:

$$\text{MASE} = \frac{T-m}{h} \frac{\sum_{j=1}^{h} |Y_{T+j} - \hat{Y}_{T+j}|}{\sum_{i=1}^{T-m} |Y_i - Y_{i-m}|} \,.$$

[1] `https://anonymous.4open.science/r/HERMES-703F/`

Table 2: Results summary on the 10000ts Fashion dataset. For each metric, the average on all our time series is computed. For approaches using neural networks, 10 models are trained with different seeds. The mean and the standard deviation of the 10 results are displayed.

	MASE ↓		**ACCURACY ↑**	
	mean	*std*	*mean*	*std*
snaive	0.881	-	0.357	-
thetam	0.844	-	0.482	-
arima	0.826	-	0.464	-
ets	0.807	-	0.449	-
stlm	0.770	-	0.482	-
hermes-ets-ws	0.769	0.005	0.501	0.007
hermes-ets	0.758	0.001	0.490	0.006
tbats	0.745	-	0.453	-
lstm-ws	0.728	0.004	0.500	0.008
lstm	0.724	0.003	0.498	0.007
hermes-tbats	0.715	0.002	0.488	0.008
hermes-tbats-ws	**0.712**	0.004	**0.510**	0.005

Detecting emerging and declining trends is a crucial issue for the fashion industry. A correct or incorrect prediction could lead to good returns or massive waste due to overstock or unsold clothes. In addition to the MASE accuracy metric, the different methods are also evaluated on a classification task and especially differences between methods using weak signals or not. In a given year, an increasing trend is defined as a trend that does more than 5% of growth on average with respect to the previous year. In the same way, a decreasing trend is defined as a trend that declines by 5% on average or more. Other trends are classified as flat trends. With this threshold, the proposed fashion dataset is almost balanced on the *test* set: There are 3087 increasing trends, 3342 decreasing trends and 3571 flat trends. To compare the different methods on this classification task, the accuracy metric, defined as the percentage of correct classification, is used.

4.3 Result for the Fashion dataset

10000 Fashion time series global accuracy. For the two metrics and for each model, we compute the average on all sequences in the final year. Results are displayed in Table 2. For methods using neural networks, 10 models are trained with different seeds. The average and the standard deviation of their results are computed and displayed. For the statistical models, TBATS largely dominates the alternatives in terms of MASE. It is one of the main motivations why this model is used on the best HERMES candidate as the predictor model.

Considering the new HERMES approach, *hermes-tbats* and *hermes-tbats-ws* slightly outperform the alternatives in terms of MASE and are stable across the different trainings. Regarding *hermes-ets*, although it is very similar to the baseline Smyl (2020), its accuracy remains low in comparison to the *lstm* benchmark or HERMES using TBATS.

Models using our weak signals perform similarly as without-weak-signals models for the MASE. Interestingly, weak signals significantly improve the accuracy in detecting emerging and declining trends. Figure 7 displays some examples of *hermes-tbats* models and some weaknesses that can be corrected.

10000 Fashion time series classification task. Classification results between the *tbats* model and the hybrid method *hermes-tbats* are given in Table 3, we note an impressive decrease of impactful errors: i.e. forecasting an increase instead of a decrease and vice versa. The *hermes-tbats* model divides by 3 the error rate in comparison to *tbats* with only a slight decrease of the number of correct increase/decrease predictions. However, with our weak signals, we see that *hermes-tbats-ws* is able to catch twice as much as its relative model without weak signals while keeping a relatively low number of impactful errors.

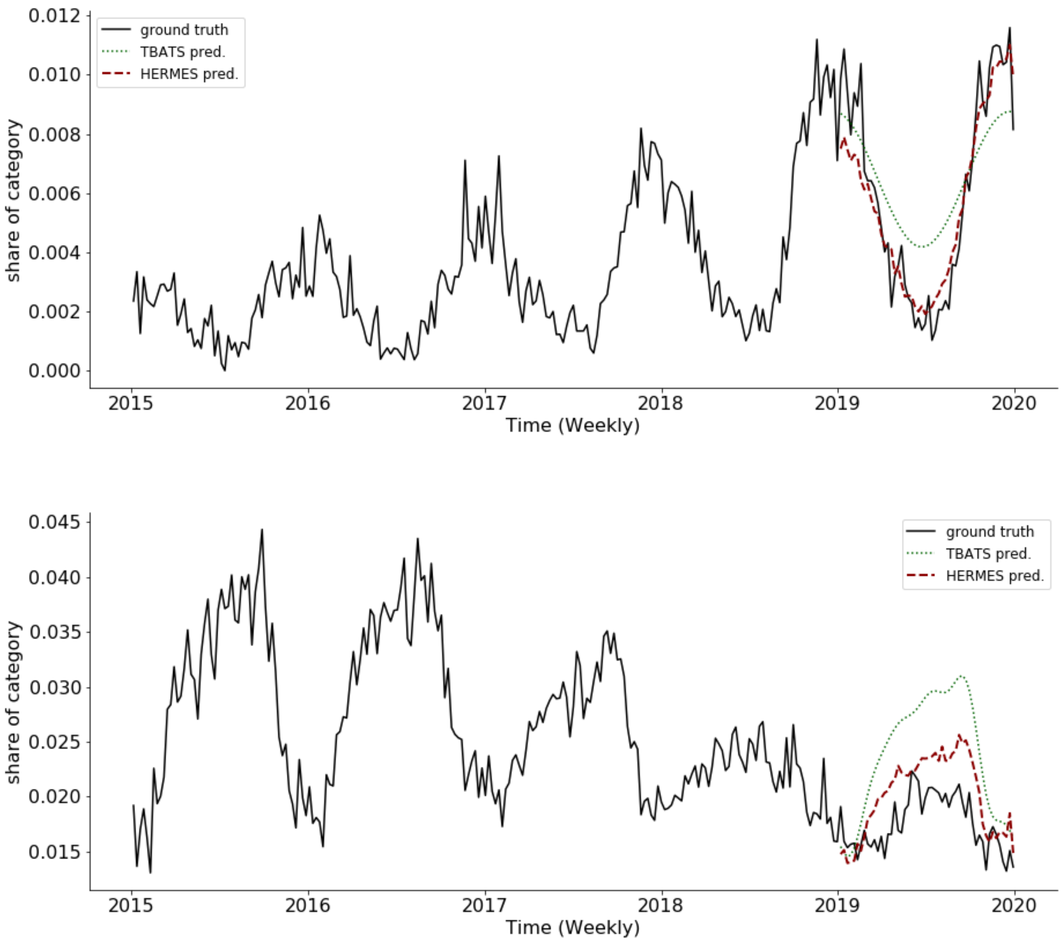

Figure 7: *hermes-tbats* forecast examples. In green the prediction of the per-time-series predictors *tbats*. In red the final forecast of our HERMES hybrid model *hermes-tbats*. (Top) Time series representing a top fashion trend for females in The United States. (Bottom) Time series representing the horizontal stipes texture fashion trend for females in China.

Table 3: *tbats*, *hermes-tbats* and *hermes-tbats-ws* models confusion matrix

	tbats		
	pred-dec	pred-flat	pred-inc
true-dec	902	2113	327
true-flat	351	2920	300
true-inc	300	2078	709

	hermes-tbats		
	pred-dec	pred-flat	pred-inc
true-dec	1261	1960	121
true-flat	549	2823	199
true-inc	214	2004	869

	hermes-tbats-ws		
	pred-dec	pred-flat	pred-inc
true-dec	1956	1245	141
true-flat	1257	2087	227
true-inc	358	1620	1109

Table 4: Results summary on the 1000 time series and 100 time series Fashion dataset. The MASE average on all the time series is computed. For the two approaches using a neural network, 10 models with different seeds are trained. the mean and the standard deviation of the 10 results are displayed.

1000 time series Fashion dataset

	MASE *mean*	*std*
snaive	0.871	-
thetam	0.849	-
arima	0.821	-
ets	0.801	-
stlm	0.765	-
lstm	0.740	0.007
tbats	0.734	-
hermes-tbats	**0.719**	0.002

100 ts Fashion dataset

	MASE *mean*	*std*
snaive	0.876	-
thetam	0.823	-
arima	0.814	-
ets	0.785	-
lstm	0.767	0.045
stlm	0.742	-
tbats	0.745	-
hermes-tbats	**0.739**	0.003

Size of the dataset. In addition to the results on the whole fashion dataset, the robustness of the HERMES model is analyzed when it is trained on smaller datasets. Two experiments are performed on a sub sample of respectively 1000 and 100 randomly selected time series. Results are given in Table 4. The hybrid framework *hermes-tbats* achieves the best performance in terms of global accuracy on both datasets. We can note that the accuracy of the full neural network *lstm* decreases when the dataset size decreases. On the small dataset of 100 time series, a local statistical model like *tbats* or *stlm* largely outperforms the *lstm*. Providing sharp predictions from scratch is a complex task and high-dimensional recurrent neural networks require large amounts of data to do so. By contrast, the HERMES approach can rely on its first statistical part and consequently needs less data to be trained and to obtain interesting performance.

4.4 Result for M4 weekly dataset

We also assessed the performance of HERMES using the M4 weekly dataset (Makridakis et al., 2020). The M4 dataset gathers 359 weekly time series and has 3 main differences compared to the proposed fashion dataset. Firstly, sequences do not have the same length with sequences lying between 93 and 2610 time steps. Secondly, the 359 time series come from different sectors such that finance or Industry. Accordingly, they have very distinct scales and dynamics. Thirdly, compared to the previous fashion application, the time horizon of the prediction is set to 13 for the weekly dataset and no additional external signals are provided.

Training. The M4 dataset is preprocessed as follows. As some sequences are short (93 time steps), they limit the window size w of the RNN. Consequently, 300 time steps are kept for each sequence. shorter sequences are duplicated in order to reach the length of 300 and longer sequences are cropped so as to keep the last 300 time steps. An overview of our train, eval, test set split and the resizing of the shortest sequences is given in Figure 8. Secondly, several M4 weekly time series have a large volume and a high level of variability. Consequently, Equation 2 of the HERMES framework is changed to:

$$\widehat{y}^{pred,n}_{T+i|T} = \exp\left(f^n(log\left(y^n_{1:T}\right); \theta^n_{predictor})_i\right) . \tag{5}$$

This simple modification increases significantly the accuracy of the per-time-series predictors tested on the M4 weekly dataset while reducing the fitting time. As for the fashion dataset, a complete grid search is done on the M4 weekly dataset to fix hyperparameters of the HERMES architecture. The horizon h is set to 13 and the window size w to 104. For the RNN part, the same architecture as described in Section 4.2 is used. The Adam optimizer is used and the MASE is directly used as the loss function. Finally a rolling window is applied on the train set so as to increase the number of examples and improve the training process. The number of slinding windows, the learning rate, the batch size, the RNN architecture and input size are set using a grid search and detailled in B.2.

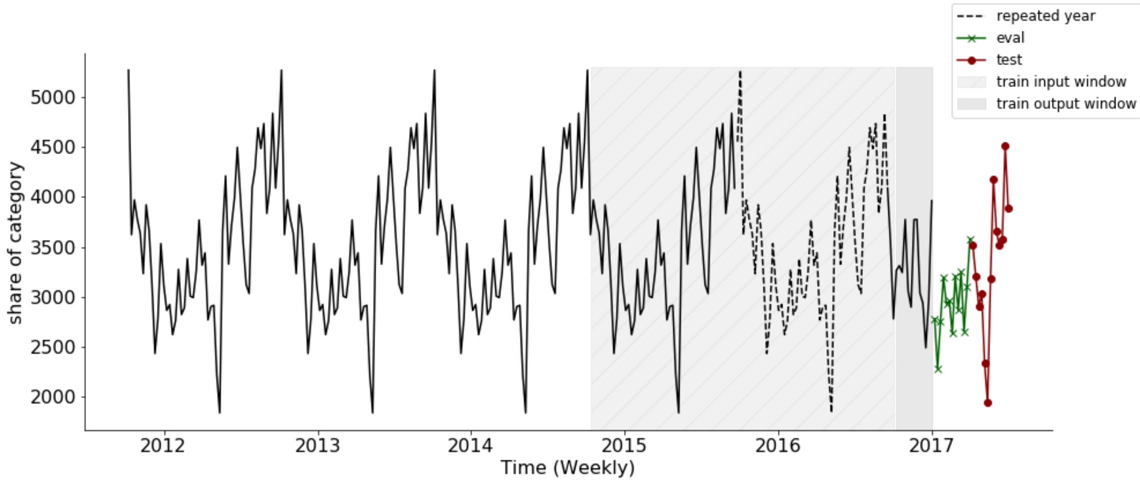

Figure 8: One of the shortest sequences of the M4 weekly dataset (93 time steps). In order to fit its predictor, the last complete year of the train set is duplicated in order to reach a total length of 300 time steps.

Evaluation. The proposed model is evaluated along with a rich collection of benchmarks provided by the M4 competition, encompassing statistical models and neural network approaches. In addition, the hybrid model named *Uber* of S.Smyl is added. For a complete description and references of the benchmark models and the hybrid model *Uber*, see Makridakis et al. (2020) and Smyl (2020). As a HERMES candidate, a version using TBATS is proposed and called *hermes-tbats*. We propose a focus on the top 3 models reaching the highest accuracy on the M4 weekly dataset. These three methods are based on an ensembling and combine various approaches. The first model is presented in Darin & Stellwagen (2020) and called *Darin & Stellwagen*. The second model is introduced in Petropoulos & Svetunkov (2020) and called *Petropoulos & Svetunkov*. Finally, a description of the third model called *Pawlikowski, et al.* can be found in Pawlikowski & Chorowska (2020). An ensembling combining 4 HERMES variations is proposed. It is based on the FFORMA algorithm introduced in Montero-Manso et al. (2020) and called *fforma-hermes*. A complete description of the training process of the proposed ensembling is given in A.3. Following the M4 competition methodology, all the candidates are evaluated according to the MASE, the SMAPE and the OWA measures. A complete definition of these metrics is proposed in Makridakis et al. (2020) and summarized in A.1. See also A.1 for additional information about the M4 weekly dataset.

Results and discussion. The final results for the M4 weekly dataset are displayed in Table 5. The HERMES approach *hermes-tbats* outperforms all the benchmarks. This result is partially induced by the use of TBATS per-time-series predictors which achieve impressively good results on the test set. Regarding the hybrid model proposed by S.Smyl, its accuracy remains low in comparison to *tbats* and *hermes-tbats*. For the ensembling methods, the proposed FFORMA model with 4 HERMES variations *fforma-hermes* reaches the same high level of accuracy as the top 3 methods of the competition on the weekly dataset. The results provided by *hermes-tbats* confirm that the HERMES model is well suited for a large collection of forecasting tasks even difficult ones with small datasets, heterogeneous time series and the absence of additional useful external signals. Secondly, the accuracy gap between the proposed hybrid model and the approach proposed in Smyl (2020) illustrates the importance of a global framework able to leverage any kind of per-time-series predictors depending of the use cases. Finally, our model can be easily included as part of an ensembling method to improve the final robustness and accuracy of the predictions.

5 Conclusion

In this paper, we propose a new hybird model for non stationary time series forecasting. By mixing the performance of local parametric models and a global neural network, *hermes-tbats* clearly outperforms traditional statistical methods and full neural network models on two forecasting tasks. Furthermore, this new

Table 5: Results summary on the m4 weekly dataset. For each metric, the average on all our time series is computed. For approaches using a neural network, 10 models are trained with different seeds. The mean and the standard deviation of the 10 results are displayed.

	SMAPE		**MASE**		**OWA**	
	mean	*std*	*mean*	*std*	*mean*	*std*
MLP	21.349	-	13.568	-	3.608	-
RNN	15.220	-	5.132	-	1.755	-
snaive	9.161	-	2.777	-	1.000	-
SES	9.012	-	2.685	-	0.975	-
Theta	9.093	-	2.637	-	0.971	-
Holt	9.708	-	2.420	-	0.966	-
Com	8.944	-	2.432	-	0.926	-
Damped	8.866	-	2.404	-	0.917	-
Uber Smyl (2020)	7.817	-	2.356	-	0.851	-
tbats	7.409	-	2.204	-	0.801	-
hermes-tbats	**7.383**	0.016	**2.191**	0.010	**0.797**	0.002
Pawlikowski, et al.	6.919	-	2.158	-	0.766	-
Petropoulos & Svetunkov	6.726	-	2.133	-	0.751	-
Darin & Stellwagen	**6.582**	-	2.107	-	0.739	-
fforma-hermes	6.614	-	**2.058**	-	**0.732**	-

model is totally suited to deal with external signals. With a fine pre-processing and a well-designed architecture, the proposed hybrid framework succeeds at leveraging complex extra data and reaches promising accuracy levels. In addition, a fashion dataset gathering a sample of 10000 time series and a collection of weak signals is provided. By making it publicly available, we hope that it will enhance the diversity of datasets for time series forecasting and pave the way for further explorations. As a possible future work, designing new models for the weak signals would improve their inclusion in the HERMES architecture. Focusing on the examples with important changes of behaviours, a fine analysis of the impact of the collection of weak signals is the topic of ongoing works. In the same way, an interesting improvement of the hybrid framework can be to introduce not a single but several neural networks trained at correcting different kinds of weaknesses. A perspective is to add a latent discrete label to select dynamically the regime shifts.

References

Kasun Bandara, Christoph Bergmeir, and Hansika Hewamalage. LSTM-MSNet: Leveraging forecasts on sets of related time series with multiple seasonal patterns. *IEEE transactions on neural networks and learning systems*, 2020.

George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

Robert G. Brown and Richard F. Meyer. The fundamental theorem of exponential smoothing. *Operations Research*, 9(5):673–685, 1961.

François Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1251–1258, 2017.

Robert B Cleveland, William S Cleveland, Jean E McRae, and Irma Terpenning. Stl: A seasonal-trend decomposition. *J. Off. Stat*, 6(1):3–73, 1990.

Sarah Goodrich Darin and Eric Stellwagen. Forecasting the M4 competition weekly data: Forecast Pro's winning approach. *International Journal of Forecasting*, 36(1):135–141, 2020.

Randal Douc, Éric Moulines, and David Stoffer. *Nonlinear time series: theory, methods and applications with R examples*. Chapman and Hall/CRC, 2014.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Rob J Hyndman, George Athanasopoulos, Christoph Bergmeir, Gabriel Caceres, Leanne Chhay, Mitchell O'Hara-Wild, Fotios Petropoulos, Slava Razbash, and Earo Wang. Package 'forecast'. *Online] https://cran. r-project. org/web/packages/forecast/forecast. pdf*, 2020.

E Jianwei, Jimin Ye, and Haihong Jin. A novel hybrid model on the prediction of time series and its application for the gold price analysis and forecasting. *Physica A: Statistical Mechanics and its Applications*, 527:121454, 2019.

Shiyang Li, Xiaoyong Jin, Yao Xuan, Xiyou Zhou, Wenhu Chen, Yu-Xiang Wang, and Xifeng Yan. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. *arXiv preprint arXiv:1907.00235*, 2019.

Bryan Lim, Sercan O Arik, Nicolas Loeff, and Tomas Pfister. Temporal fusion transformers for interpretable multi-horizon time series forecasting. *arXiv preprint arXiv:1912.09363*, 2019.

Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft COCO: Common objects in context, 2014.

Alysha M. De Livera, Rob J. Hyndman, and Ralph D. Snyder. Forecasting time series with complex seasonal patterns using exponential smoothing. *Journal of the American Statistical Association*, 106(496):1513–1527, 2011.

Yunshan Ma, Yujuan Ding, Xun Yang, Lizi Liao, Wai Keung Wong, and Tat-Seng Chua. Knowledge enhanced neural fashion trend forecasting. In *Proceedings of the 2020 International Conference on Multimedia Retrieval*, pp. 82–90, 2020.

Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. The M4 competition: Results, findings, conclusion and way forward. *International Journal of Forecasting*, 34(4):802–808, 2018.

Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. The M4 competition: 100,000 time series and 61 forecasting methods. *International Journal of Forecasting*, 36(1):54–74, 2020.

Alice Martin, Charles Ollion, Florian Strub, Sylvain Le Corff, and Olivier Pietquin. The Monte Carlo Transformer: a stochastic self-attention model for sequence prediction. *arXiv preprint arXiv:2007.08620*, 2021.

Pablo Montero-Manso, George Athanasopoulos, Rob J. Hyndman, and Thiyanga S. Talagala. FFORMA: Feature-based forecast model averaging. *International Journal of Forecasting*, 36(1):86–92, 2020. M4 Competition.

Maciej Pawlikowski and Agata Chorowska. Weighted ensemble of statistical models. *International Journal of Forecasting*, 36(1):93–97, 2020.

Fotios Petropoulos and Ivan Svetunkov. A simple combination of univariate models. *International Journal of Forecasting*, 36(1):110–115, 2020. ISSN 0169-2070.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015.

Everett M Rogers. *Diffusion of innovations*. Simon and Schuster, 1962.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge, 2014.

Table 6: M4 weekly dataset overview. For each category, the number of sequences and the average length are given.

	Nb. of sequences	**Avg. length**	**Min. length**
Demographic	24	1659	1615
Finance	164	1237	260
Industry	6	834	356
Macro	41	1264	522
Micro	112	473	**93**
Other	12	1598	470

David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting*, 36(3):1181–1191, 2020.

Simo Särkkä. *Bayesian filtering and smoothing*. Cambridge university press, 2013.

Sima Siami-Namini, Neda Tavakoli, and Akbar Siami Namin. A comparison of ARIMA and LSTM in forecasting time series. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pp. 1394–1401, 2018.

Slawek Smyl. A hybrid method of exponential smoothing and recurrent neural networks for time series forecasting. *International Journal of Forecasting*, 36(1):75–85, 2020.

Augustin Touron. Modeling rainfalls using a seasonal hidden Markov model, 2017.

Augustin Touron. Consistency of the maximum likelihood estimator in seasonal hidden Markov models. *Statistics and Computing*, 29(5):1055–1075, 2019.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *31st Conference on Neural Information Processing Systems (NeurIPS 2017)*, 2017.

G Peter Zhang. Time series forecasting using a hybrid ARIMA and neural network model. *Neurocomputing*, 50:159–175, 2003.

Walter Zucchini, Iain L MacDonald, and Roland Langrock. *Hidden Markov models for time series: an introduction using R*. CRC press, 2017.

A M4 weekly dataset, Ensembling training and results

A.1 M4 weekly dataset

The M4 weekly dataset is a collection of 359 time series with contrasting behaviours and sizes. An overview of the dataset is given in Table 6 and some examples of sequences are given in Figure 9. This use case is not properly suited for the HERMES approach as the dataset is small and there is no clear link between time series. Moreover, no additional external signals are available that could help the RNN part to correct the first errors of the per-time-series predictors.

A.2 M4 accuracy metrics

The M4 competition proposes 3 metrics to evaluate the different approaches: the mean absolute scaled error (MASE), the symmetric mean absolute percentage error (SMAPE) and the overall weighted average (OWA). MASE and SMAPE are defined as follow:

$$\text{MASE} = \frac{T-m}{h} \frac{\sum_{j=1}^{h} |Y_{T+j} - \hat{Y}_{T+j}|}{\sum_{i=1}^{T-m} |Y_i - Y_{i-m}|}\,, \qquad \text{SMAPE} = \frac{2}{h} \sum_{j=1}^{h} \frac{|Y_{T+j} - \hat{Y}_{T+j}|}{|Y_{T+j}| + |\hat{Y}_{T+j}|}\,,$$

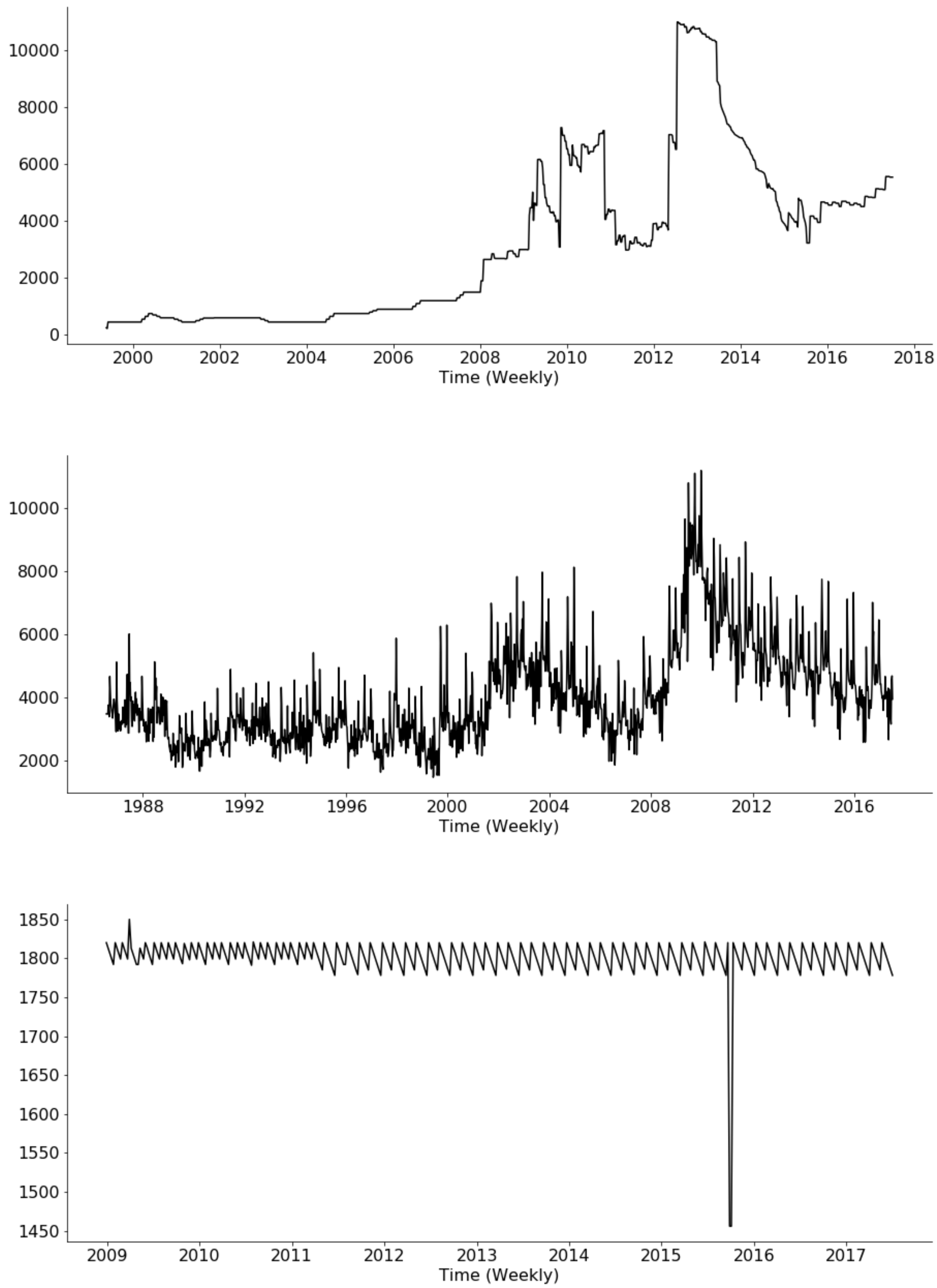

Figure 9: Examples of time series from the M4 weekly dataset. From Top to Bottom : time series called *W10* from the *Other* category, *W20* from the *Macro* category and *W220* from the *Finance* category.

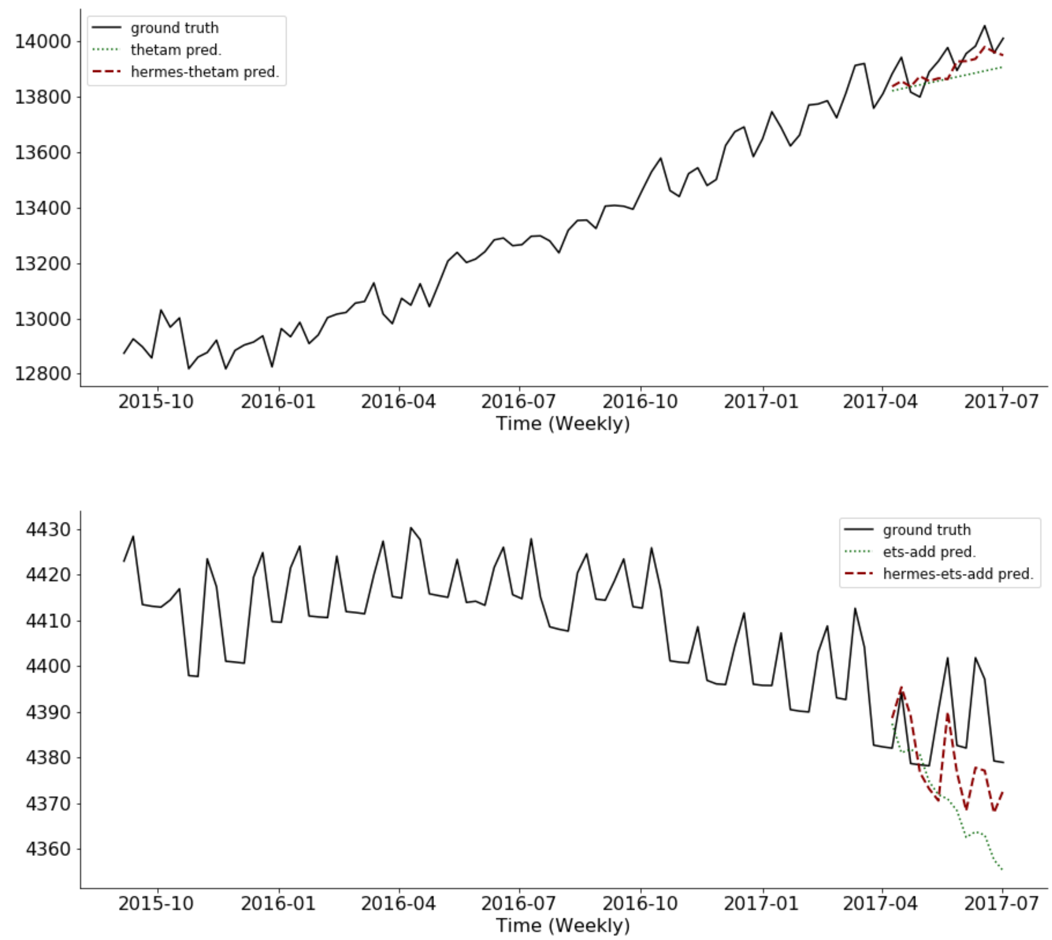

Figure 10: forecast examples of HERMES variations on 2 time series of the M4 weekly dataset. At the top, the *W133* time series is displayed with the prediction of the per-time-series predictor *thetam* (green) and the final forecast of the HERMES hybrid model *hermes-thetam* (red). At the bottom, the *W262* time series is represented with the corresponding prediction of the per-time-series predictors *ets-add* (green) and the HERMES correction of *hermes-ets-add* (red).

Table 7: Results summary on the m4 weekly dataset of the HERMES variations. For each metric, the average on all the time series is computed. For approaches using a neural network, 10 models are trained with different seeds. The mean and the standard deviation of the 10 results are displayed. For the statistical models *ets-add*, *ets-mul* and *thetam*, the Python package `statsmodels` is used. The Python package `tbats` is used for the *tbats* approach.

	SMAPE		**MASE**		**OWA**	
	mean	*std*	*mean*	*std*	*mean*	*std*
ets-mul	8.933	-	2.412	-	0.922	-
hermes-ets-mul	8.889	0.021	2.377	0.016	0.913	0.004
ets-add	8.929	-	2.410	-	0.921	-
hermes-ets-add	8.880	0.022	2.377	0.016	0.913	0.004
thetam	7.609	-	2.377	-	0.843	-
hermes-thetam	7.590	0.012	2.359	0.010	0.839	0.002
tbats	7.409	-	2.204	-	0.801	-
hermes-tbats	**7.383**	0.016	**2.191**	0.010	**0.797**	0.002

where h is the forecast horizon and m the length of the seasonality. The final OWA is computed by following these steps: i) compute the average MASE and SMAPE of a model. ii) Divide the previous results by the MASE and SMAPE computed with the benchmark method *snaïve*. iii) Compute the OWA as the average of the relative MASE and SMAPE obtained is step ii). As an example on the m4 weekly dataset, the method *hermes-tbats* gets a MASE of 7.383 and a SMAPE of 2.191. The benchmark method *snaïve* obtains a MASE of 9.161 and a SMAPE of 2.777. Thus the OWA of *hermes-tbats* is equal to 0.797.

$$OWA_{hermes\text{-}tbats} = \frac{1}{2}(\frac{7.282}{9.161} + \frac{2.191}{2.777}) \approx 0.797$$

A.3 FFORMA ensembling with HERMES variations

In this section, a complete description of the proposed ensembling on the M4 weekly dataset is provided. In a first time, 4 HERMES variations are trained using different per-time-series predictors. The first one called *hermes-tbats* uses TBATS and is presented in Section 4.4. The second version is called *hermes-thetam* and use the Thetam method provided with the Python package `statsmodels`. The two remaining variations use as per-time-series predictors an additive or multiplicative exponential smoothing and are called respectively *hermes-ets-add* and *hermes-ets-mul*. As for Thetam, the Python package `statsmodels` is used to fit the different exponential smoothing models. Concerning the HERMES architecture, for simplicity, hyperparameters described in Section 4.4 are used for each version but a grid search could have be run for each of them. 10 models are trained per version with different seeds and the best one based on the eval set is kept for the ensemble model. In a second time, the FFORMA ensembling introduced in Montero-Manso et al. (2020) is used to combine the 4 HERMES methods. The R package `M4metalearning` containing the FFORMA model is directly used without change of the hyperparameters, imported in Python with the library `Rpy2` and combined with the HERMES code base.

A.4 M4 weekly dataset results

In addition of the results provided is Section 4.4, Table 7 displays the results of all the HERMES variations included in the FFORMA ensembling as well as the accuracy of the per-time-series predictors. In each cases, HERMES approaches always improve the predictors accuracy. These improves can appear slight but are justified regarding the absence of link between time series and the absence of additional useful external signals. Nevertheless, efficient corrections can be obtained on some examples as displayed in Figure 10.

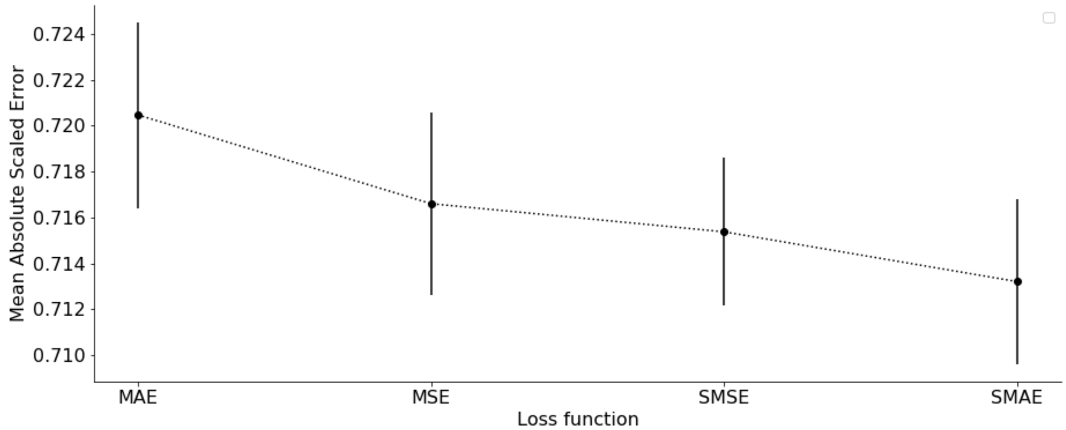

Figure 11: MASE accuray for the *hermes-tbats-ws* model depending on the loss used during the RNN training. For each loss, 10 models with different seeds have been trained. The mean and the standard deviation are represented with a point and a vertical line.

B Training parameters and loss

B.1 Loss grid search on the Fashion Dataset

Using deep learning models in time series forecasting is an appealing way to achieve higher accuracy performance. However, it induces two main issues. First, it requires a large enough dataset to train the model as illustrated in Section 4. Second, a dataset can hide contrasting time series in terms of scale, noise and behaviour. These differences can impact training performance. For the HERMES architecture, some candidate losses were defined for the training: the Mean Absolute Error (MAE), the Mean Square Error (MSE), the Scaled Mean Absolute Error (SMAE) and the Scaled Mean Square Error (SMSE). The loss functions are defined as follows:

$$MAE = \frac{1}{h}\sum_{i=1}^{h} |y^n_{T+i} - \widehat{y}^n_{T+i|T}| \,,$$

$$MSE = \frac{1}{h}\sum_{i=1}^{h} (y^n_{T+i} - \widehat{y}^n_{T+i|T})^2 \,,$$

$$SMAE = \frac{1}{\bar{y}^n_T}\sum_{i=1}^{h} |y^n_{T+i} - \widehat{y}^n_{T+i|T}| \,,$$

$$SMSE = \frac{1}{\bar{y}^n_T}\sum_{i=1}^{h} (y^n_{T+i} - \widehat{y}^n_{T+i|T})^2 \,.$$

For each loss, 10 *hermes-tbats-ws* models have been trained with different seeds and the final mean and standard deviation are given in Figure 11. The final Scaled Mean Absolute Error reaches the lowest MASE and was selected to train all the HERMES models on the Fashion dataset.

B.2 Parameters grid search on the M4 weekly Dataset

In addition to the loss function, the HERMES model also depends on several hyperparameters to set correctly in order to reach satisfactory performance. For instance, an overview of the learning rate, batch size and number of windows per time series grid search for the M4 weekly dataset is shown in Figure 12. For each parameter, a collection of 10 *hermes-tbats* models have been trained with different seeds and the final OWA

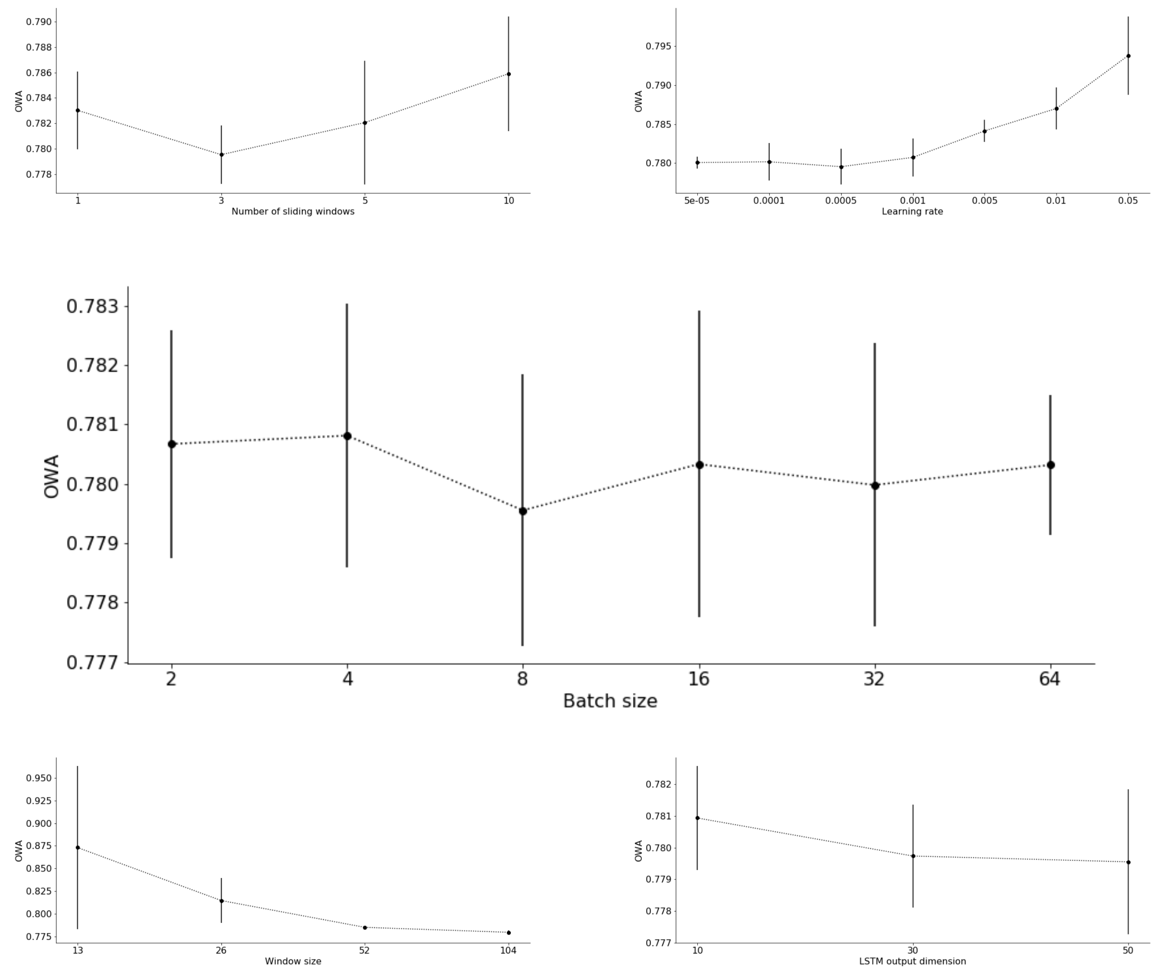

Figure 12: OWA for the *hermes-tbats* model on the eval set of the M4 weekly dataset. 5 hyperparameters used during the RNN training are tested: the number of moving windows per time series (top left), the learning rate (top right), the batch size (middle), the window size for the RNN input (bottom left) and the dimension of the LSTM layers output (bottom right). For each parameter, 10 models with different seeds have been trained. The mean and the standard deviation of the OWA on the eval set are represented with a point and a vertical line.

was calculated. As in the Figure 11, the mean and the standard deviation of each group of 10 trainings is computed. For the final *hermes-tbats* model of the M4 weekly dataset, the following set of parameters was selected: 3 windows per time series were used as the train set, the batch size was set to 8 and the learning rate was fixed to 0.005.