Cluster-based Forecasting for Intermittent and Non-intermittent Time Series

Tom van de Looij[1] and Mozhdeh Ariannezhad[2]

[1] University of Amsterdam, Amsterdam, The Netherlands
tom.vandelooij@student.uva.nl
[2] AIRLab, University of Amsterdam, Amsterdam, The Netherlands
m.ariannezhad@uva.nl

Abstract. Producing accurate forecasts is an essential part of successful inventory management for any retail business. Previous research has shown that the clustering of time series data into disjoint clusters can reduce the forecast error, eventually leading to cost savings. A common measure used to cluster time series data is Dynamic Time Warping. While it can handle time series of different length and guarantees to provide the optimal alignment, it is computationally expensive and assumes that one time series is a stretched non-linear version of another time series. For datasets containing intermittent time series, i.e. showing no clear structure, DTW is not the best suited method. In this paper, we propose a new framework that uses Simple Exponential Smoothing (SES) and a Self-Organizing Map (SOM) that is able to improve the clustering performance for clusters containing intermittent and non-intermittent time series. Using LightGBM as the forecasting model, we evaluate our approach on a real-world dataset, and find that the computational time can be reduced substantially compared to DTW when using a combination of SOM and LightGBM for both intermittent and non-intermittent time series while maintaining similar levels of accuracy.

Keywords: Intermittent time series, clustering, forecasting, self-organizing map, hierarchical agglomerative clustering, dynamic time warping

1 Introduction

For any business, sales forecasting is a critical task in order to maintain a correct inventory level, where purchasing too few units of a product leads to a lost sales opportunity and buying too much units results in an overstock. Therefore, producing accurate forecasts is an essential component in maintaining an efficient supply chain [21]. According to [26], intermittent demand may constitute up to 60% of the total stock value. As a result, small improvements in the management of intermittent stock items can contribute to substantial cost savings. Intermittent time series are characterized by multiple non-demand intervals and do not contain enough data to model trend and seasonality [19]. This makes intermittent time series hard to forecast as there are two sources of uncertainty: the sporadic nature of the demand level and the timing of the demand occurrence [17].

Over the last five decades, since the first specialised method for intermittent demand was proposed in [5], there have been very few new forecasting methods specifically designed for intermittent time series compared to fast-moving time series [17]. Previous research has shown that partitioning the dataset into disjoint clusters can reduce the forecast error on fast-moving time series, as data in the same cluster will have similar patterns [9]. A common method to calculate the similarity between two time series is Dynamic Time Warping (DTW) [1]. While it is an accurate measure to cluster misaligned time series that are similar in shape, it assumes that one time series is a stretched non-linear version of another time series and is expensive to compute. Moreover, previous research has only evaluated its performance and impact on the forecasting accuracy on small datasets (<1000 Stock Keeping Units (SKUs)) [20,18]. In a real world business setting, where weekly or even daily forecasts are needed, the use of DTW might be infeasible because of the computational complexity.

In this paper, we introduce a cluster-based forecasting framework that is able to forecast both intermittent and non-intermittent time series. We propose the use of Simple Exponential Smoothing (SES) as a preprocessing step before clustering. The idea behind this approach is that SES can act as a filter that removes noise from the time series and reduces outliers in the Euclidean distance, an often used distance metric in clustering. It is computationally faster to use and it can especially improve the clustering performance for intermittent time series. Smoothed time series are then less penalised by the Euclidean distance when they are misaligned, making it especially useful for intermittent time series.

We summarize our contributions as follows:

- We use a new preprocessing method before clustering to improve the clustering performance that is computationally faster to use than DTW (Section 3).
- We showcase our results on a publicly available dataset. Our results show that our cluster-based approach can achieve similar results in terms of the root mean squared error at a substantial lower computational cost compared to DTW (Section 4).
- We demonstrate that a gradient boosting machine (LightGBM) can benefit from a cluster-based approach with respect to the root mean squared error.

2 Related Work

Clustering Time Series Cluster-based forecasting is a well studied research problem. Dividing time series into clusters can result in much smaller forecasting errors in contrast to a direct prediction [9]. The key insight in a cluster-based forecasting approach is that by partitioning the whole dataset into multiple disjoint clusters, the forecasting models trained on those separate clusters are able to improve the accuracy compared to a forecasting model trained on the whole dataset [13].

In order to cluster time series, Dynamic Time Warping (DTW) is used to as a dissimilarity metric for time series of unequal length [20]. It calculates a warping path between two sequences and is able to provide the optimal alignment. In

contrast to the Euclidean distance metric, DTW can be used to cluster time series that are similar in shape but are out of phase. While DTW can be a good measure to match misaligned time series, it prove to be an infeasible method to use on a larger dataset. To calculate a distance matrix for n time series of length m, the complexity for such a operation is $\mathcal{O}(n^2m^2)$. This makes DTW an unsuitable measure in practice in terms of complexity compared to the Euclidean distance. DTW also assumes that one time series is a stretched non-linear version of another time series. In online retail, where intermittent time series are more prevalent [8], i.e. showing no clear structure, DTW is not the best suited measure to cluster intermittent time series.

Our focus is on efficiently clustering and modelling of a large dataset ($\approx$ 50,000 time series) at the lowest aggregation level containing both fast-moving and intermittent time series. The proposed methods in the above mentioned works are mainly evaluated on small datasets ($<$ 1000 time series) that are forecasted at the highest aggregation level [29,14,20]. In contrast, we propose a method that foregoes the use of DTW as a distance measure but instead uses Simple Exponential Smoothing (SES) as preprocessing method to improve the clustering performance.

Time Series Forecasting Time series forecasting has been an active area of research since the 1970s. The publication of *Time Series Analysis: Forecasting and Control* by Box and Jenkins in 1970 is perceived as an important milestone. It enabled forecast practitioners to apply a systematic approach in time series forecasting [27]. Since then, a wide diversity of methods and algorithms have been used for time series forecasting. According to [15], these methods and algorithms can be divided into two groups. The first group consists of classical statistical approaches such as SES, Auto-Regressive Integrated Moving Average (ARIMA), Theta, and Box-Jenkins. These methods are characterized as linear methods and have a strong capability of modeling trend and seasonality in a time series. The second group consists of machine learning (ML) based methods like Support Vector Regression (SVR), Long Short-Term Memory (LSTM), and tree-based models like XGBoost, LightGBM and Random Forest [15].

More recently, ML based methods are being proposed as an alternative to the classical statistical methods [2]. A good indication on the advances in forecasting theory and practice are the M competitions. The M4 competition, held in 2018, showcased that a combination of statistical models and ML models were able to outperform classical methods in terms of accuracy, highlighting the potential value of ML-based approaches in more accurate forecasting. In the M5 competition, held in 2020, exclusively ML-based methods were the winning solutions. As an example, LightGBM, a gradient-boosting framework developed by Microsoft, is able to process numerous, correlated time series effectively and reduce the forecast error [16]. In contrast to the previous M competitions, the M5 competition was also the first competition to include intermittent demand time series.

In this paper, we use the LightGBM model as our forecasting model and conduct our experiments on the M5 competition dataset.

Forecasting Intermittent Time Series In contrast to modelling fast-moving time series, forecasting intermittent time series is a more difficult task, as there are two sources of uncertainty: the sporadic nature of the demand volume and the sporadic timing of the demand arrivals [17]. The first proposed method to specifically forecast intermittent time series was introduced by Croston [5]. It overcomes the aforementioned difficulties by using separate series for the size of demand, and for the demand frequency. Each series is forecasted separately using Simple Exponential Smoothing (SES) and the final forecast is derived by:

$$\hat{y}_t = \frac{\hat{z}_t}{\hat{p}_t} \tag{1}$$

Where $\hat{z}_t$ and $\hat{p}_t$ are the forecast for the demand sizes and the demand intervals, respectively. Later, [24] proved that the Croston's method was biased and proposed a new method called Syntetos-Boylan Approximation (SBA): a bias-correction approximation [25]. To overcome the problem of the bias being positively correlated with the α_p smoothing parameter, a damping factor was added to Croston's method:

$$\hat{y}_t = (1 - \frac{\alpha_p}{2})\frac{\hat{z}_t}{\hat{p}_t} \tag{2}$$

While there is empirical evidence that the per series optimization can gain improvements with respect to the bias, the values for the smoothing parameter are selected in a adhoc manner in practice. Moreover, even with the recent advance in specialised methods for intermittent forecasting, simpler methods like Moving Average (MA) and SES are often used in practice [19].

In this work, we propose a clustering based approach for forecasting intermittent time series, and make use of SES as a preprocessing step.

3 Methodology

We propose a cluster-based forecasting framework that leverages Simple Exponential Smoothing (SES) as a preprocessing step. By comparing SES versions of the time series to each other, we aim to improve the clustering performance.

3.1 Framework Overview

The complete framework is illustrated in Figure 1. Given a dataset X containing n time series of length m, we apply the SES as mentioned in Section 3.2 on X resulting in X'. X' is then clustered into k disjoint clusters using a clustering algorithm. In this paper, we experiment with two clustering methods. The best performing method is selected as the clustering method in the proposed framework. Finally, a total of k LightGBM models are trained on the original non-smoothed dataset X and the predicted values are obtained.

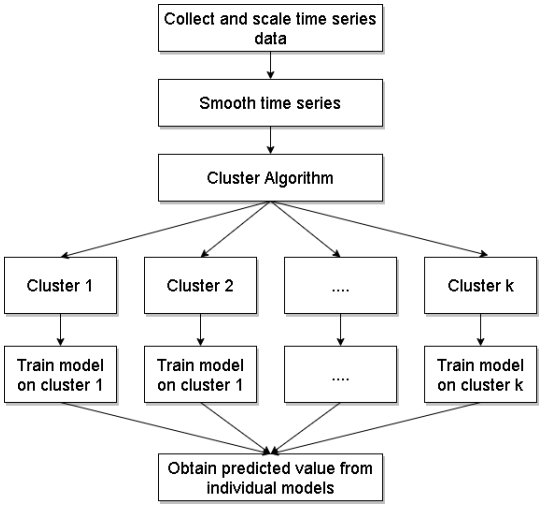

Fig. 1: The proposed framework for cluster-based time series forecasting.

3.2 Framework Details

In this section, we provide details for different components of the proposed framework. First, the smoothing step is highlighted and its effect on the clustering is discussed. We then give an overview of the two types of clustering algorithms used in our experiments. Finally, we discuss the forecasting model used to produce the predictions.

Preprocessing As mentioned in Section 2, the use of DTW as a distance metric on a large dataset is computationally expensive compared to the Euclidean distance. To still be able to cluster time series that are similar in shape, exponential smoothing is used as an additional preprocessing step. The intuition behind this approach is that smoothed time series are less penalised by the Euclidean distance metric than non-smoothed time series. It enables the clustering algorithm to capture the main components of the individual time series [23]. We use SES as the smoothing approach in our framework [7]. SES is defined as follows:

$$s_t = \alpha y_t + (1 - \alpha) s_{t-1}, \tag{3}$$

where α is a smoothing factor between 0 and 1, y_t is the value of the time series at time step t and s_t is the smoothed value. The value of s_t is a simple weighted average of the current observation y_t and the previous smoothed value s_{t-1}.

Hierarchical Agglomerative Clustering We aim to partition a dataset $X = \{x_1, ..., x_n\}$ into a collection of clusters. The first algorithm that is tried in the clustering step of the framework is Hierarchical agglomerative clustering (HAC).

In HAC, records are stored at the leaves as singleton sets and the algorithm proceeds by merging pairs of clusters until the root of the tree is reached, which contains all the elements of X. The distance between any two sub-clusters of X is called the linkage distance and can be denoted as $\Delta(X_i, X_j)$ [4]. The three most common linkage methods are single-link, complete-link, and Ward's. Single-link linkage defines the distance between two clusters as the minimum distance between their members. While it can handle quite complex cluster shapes, single-link linkage only cares about separation and does not take cluster balance or compactness into account. Complete-link linkage defines the distance between two clusters as the maximum distance between their members. Therefore, it is sensitive to outliers [22]. In our experiments, we used the Ward's method. Instead of measuring the distance between clusters directly, it analyses the variance of clusters. To merge $X_i(n_i = |X_i|)$ with $X_j(n_j = |X_j|)$ Ward's method can be defined as:

$$\Delta(X_i, X_j) = \frac{n_i n_j}{n_i + n_j} ||c(X_i) - c(X_j)||^2, \tag{4}$$

where $c(X')$ denotes the centroid of cluster X' [4]. The final output of HAC is a dendogram which defines the hierarchy of the clusters. k clusters are then selected as the final clusters, where k is the cut-off point.

Self-Organizing Map The second type of clustering algorithm used is a Self-Organizing Map (SOM). SOM is a type of artificial neural network (ANN) that is able to transform a high-dimensional input into a two-dimensional representation, called a map [11]. SOM relies on competitive learning that earns activation opportunities through competition between neurons of the output layer as opposed to error-correction learning such as back-propagation with gradient descent [3].

A SOM network consists of i neurons arranged in a 2D grid with a normally randomized weight vector m_i. The architecture allows for lateral interaction between the neurons to activate and inhibit each other. During each training iteration, a training example is fed to the network and its Euclidean distance to all weight vectors is computed. The winning neuron, also called the best-matching unit (BMU) can be expressed as:

$$W(t) = \arg\min_i \{||x(t) - m_i(t)||\}. \tag{5}$$

The update formula for the BMU can then be expressed as:

$$m_i(t+1) = m_i(t) + \alpha(t) * h_{Wi}(t)[x(t) - m_i(t)], \tag{6}$$

where t is the current training iteration and x is the input vector. The amount of movement is constrained by the time-decreasing learning rate α. The learning rate is adjusted over time in order to make substantial changes in the network at the beginning phase. The neighbor neurons near $W(t)$ are denoted by the neighbor kernel h_{Wi}. In the simplest form, it is set to 1 for all neurons close enough to the BMU and 0 for others, but in practice a Gaussian neighborhood is often used [12].

Table 1: Summary statistics for the M5 dataset.

	Demand	Intermittency
Min.	0	0.0000
Median	2.0	0.1942
Max.	4220	0.8992
Mean.	7.9002	0.2386
Std. Deviation	23.6665	0.1942

LightGBM We use LightGBM as our forecasting model. Proved to be one of the best performing methods in the M5 competition [16], LightGBM is a machine learning algorithm that combines two techniques: Gradient-based One-Side Sampling (GOSS) and Exclusive Feature Bundling (EFB) in order to handle large number of data points and a large number of features.

In a gradient boosting tree, the gradient for each data instance provides important information that can be used for data sampling. If a data instance corresponds to a small gradient then the training error for this data instance will also be small, because it is already well trained. By disregarding those data instances, the distribution of the data will change, which will in turn decrease the accuracy of the model. GOSS avoids this problem by retaining all instances with large gradients and uses random sampling on those with small gradients. To counter-act the change of distribution, a constant multiplier is introduced. As high-dimensional data are usually very sparse, this sparsity of the feature space can be used to create an approach to reduce the number of features. It is possible to bundle exclusive features into a single feature because many features are mutually exclusive. This EFB step in LightGBM can significantly speed up the training of the model without decreasing the accuracy [10]. Tweedie's Poisson regression was chosen as the objective function as it deals well with right-skewed data with a large number of zeros [30], which makes is suited for the intermittent dataset.

4 Evaluation

Data The proposed framework is evaluated on the M5 dataset from the M5 competition, ranging from January 2011 until September 2016 and aggregated at the week level. The major objective is to develop a forecasting model that can be used by the supply-chain management. Therefore, the forecasts are on the lowest aggregation level, i.e. products are forecasted at the SKU level. The dataset contains the time series for 30,472 SKUs with the length of 278. Statistical characteristics are shown in Table 1. Demand is expressed as SKU sold per week and the intermittency of a SKU is described as the fraction of zero sales, calculated from the first recorded sale in a time series. SKUs containing more than 90% of zero recorded sales are removed from the M5 dataset, containing too little information to produce any type of reliable forecast.

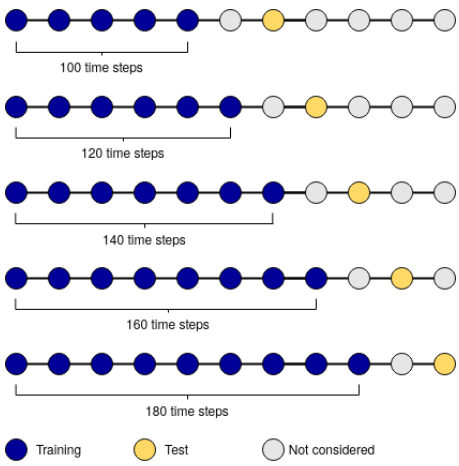

Fig. 2: Forecasting on a rolling window.

Experimental Setup The data was split on the item level via 5-fold time series cross validation. As illustrated in Figure 2, the forecasts are made on a rolling window basis. In each fold, the next 20 time steps are predicted. To prevent leakage of information between the training and the test set, a gap of eight time steps between the training set and test set is used. For the baseline models (Croston's Method and LightGBM) the mean, standard deviation and the median RMSE scores across the five folds are calculated.

To measure the difference in RMSE between the baseline model and the cluster forecasts on a cluster level, the predictions of the SKUs from each fold in each cluster are compared to the predictions of the SKUs in the corresponding fold of the baseline model. The mean, standard deviation, and median RMSE is then calculated to show the difference in RMSE on a cluster level. All experiments are done using an Intel Core i7-8559U CPU clocked at 2.70 GHz.

Evaluating Cluster Performance We use internal validity measures to evaluate the performance of the clustering algorithms. The two validity measures used are the Calinski-Harabasz (CH) score and the Davies-Bouldin (DB) score. The CH score is defined as:

$$CH(C) = \frac{(N - |C|) * \sum_{c_k \in C} |c_k| d(\overline{c_k}, \overline{X})}{(|C| - 1) * \sum_{c_k \in C} \sum_{x_i \in c_k} |c_k| d(x_i, \overline{c_k})} \tag{7}$$

With cluster scatter S denoted as:

$$S(c_k) = \frac{1}{c_k} \sum_{x_i \in c_k} d(x_i, \overline{c_k}), \tag{8}$$

the DB score of a cluster is:

$$DB(C) = \frac{1}{|C|} \sum_{c_k \in C} \max_{c_l \in C n c_k} \left\{ \frac{S(c_k + S(c_l)}{d(\overline{c_k}, \overline{c_l})} \right\}, \tag{9}$$

Where $X = \{x_1, ..., x_n\}$ is the dataset to be clustered, $C = \{c_1, ..., x_k\}$ is the clustering of X into K disjoint clusters, $\overline{c_k}$ is the centroid of the cluster and $\overline{X}$ is the centroid of the dataset [28]. The CH and DB score both measure inter and intra cluster similarities where the CH score should be maximized and the DB score should be minimized.

Evaluating Forecasting Performance Due to the high degree of intermittency in the dataset, the Mean Absolute Error (MAE) and the Mean Absolute Percentage Error (MAPE) cannot be used. The MAE optimizes for the median and when a time series contains many zeros, i.e. intermittent, the median will also be close to zero. With intermittent time series the MAPE is also not well suited because division by zero is very likely to occur.

To evaluate the forecasting performance the Root Mean Square Error (RMSE) is used. The RMSE is defined as:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{n}} \tag{10}$$

Where y_i is the actual value, $\hat{y}_i$ is the predicted value, and n is the total number of observations. Because the RMSE is scale dependent it can not be used to compare the accuracy of the models between two different datasets, however it can be used to compare the improvements in accuracy between the models on the same dataset.

4.1 Results

Time Series Clustering Table 2 illustrates the best performing clustering configurations (number of cluster and smoothing parameter) based on the Calinski-Harabasz (CH) or Davies-Bouldin (DB) score. The parameter $\alpha = \text{N/A}$ indicates that no SES was applied before clustering. The use of SES improved the clustering results for both the HAC and SOM approaches. For the HAC method, the CH and DB score are more in agreement with what the best configuration is. A smoothing parameter of 0.1 and a cluster size of 2 or 3 both result in the highest CH score and the lowest DB score. Figures 3 and 4 show the clustering performance for the smoothing values ranging from 0.1 to 1. The CH score increases as the smoothing parameter increases, indicating that the clusters are better defined. HAC shows a maximum CH score for all ten smoothing values when 2 clusters are chosen. For the SOM the CH first increases as the the number of clusters increase with a maximum CH score between 25 and 36 number of clusters. For both clustering algorithms a smoothing parameter of 0.1 appears to result in the best defined clusters.

Table 2: Best results for different clustering methods and parameters on the M5 dataset.

Clustering method	# Clusters	α	CH Score	# Clusters	α	DB Score
SOM	25	N/A	1410.0694	9	N/A	3.1737
	36	0.1	2215.3234	9	0.1	1.7413
HAC	2	N/A	1163.6122	2	N/A	2.4444
	2	0.1	3229.5529	3	0.1	1.5076

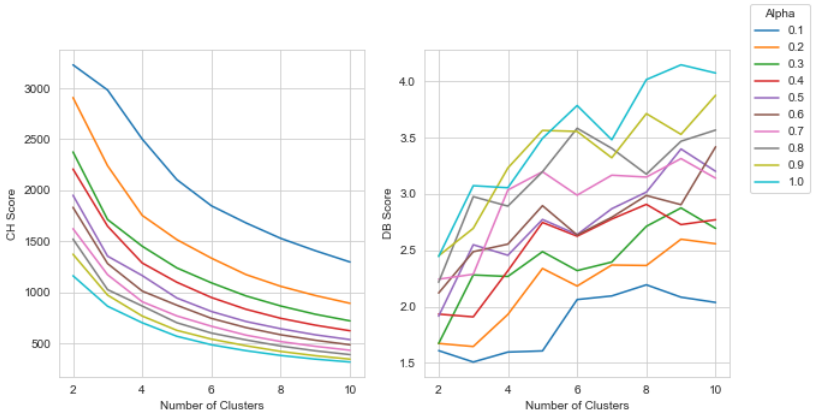

Fig. 3: Calinski-Harabasz and Davies-Bouldin scores for different smoothing parameters (alpha) using HAC.

Analyzing the cluster performance with respect to the DB score, we can see that for both types of clustering algorithms smaller clusters are preferred. The DB score is minimized for a smoothing value 0.1 for both the HAC and SOM method. For HAC the optimal number of clusters is 3 and for the SOM the optimal number of clusters is 9.

Comparison with DTW Table 3 shows the mean and median RMSE scores for a subset of 20,000 SKUs from the M5 dataset. As a non-ML-based method, Croston's Method achieves a mean RMSE of 13.1765. A single LightGBM model is able to achieve a mean RMSE of 8.2276 across five folds and serves as the baseline model to which cluster-based models are compared. The SOM+LightGBM is the best performing model with respect to the mean RMSE, reducing the RMSE by 0,2257 (2,82%). The DTW+HAC+LightGBM model is able to achieve similar performance as the SOM+LightGBM model indicating that it could be a viable alternative. The SES+HAC+LightGBM model is not able to decrease the mean RMSE, performing worse than the baseline model. While SES is able to improve the clustering performance, it is not able to outperform the SOM model

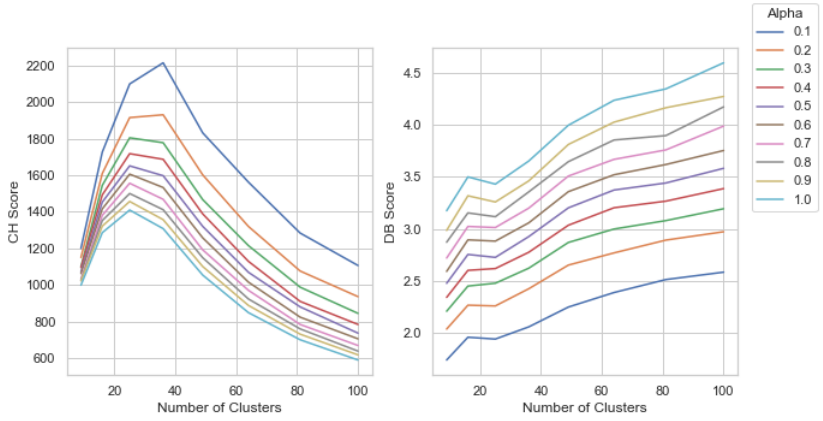

Fig. 4: Calinski-Harabasz and Davies-Bouldin scores for different smoothing parameters (alpha) using SOM.

Table 3: Mean and median RMSE scores across five folds for various models on the M5 dataset. Best performing scores are highlighted in gray.

Model	**# Clusters**	α	**Mean**	**Median**	**CPU Time**
Croston's Method	N/A	N/A	13.1765 (± 0.9298)	12.9772	2min 15s
LightGBM	N/A	N/A	8.2276 (± 0.3470)	8.1604	16min 13s
SOM+LightGBM	9	N/A	8.0019 (± 0.3524)	8.0830	5min 19s
SES+SOM+LightGBM	9	0.1	8.0082 (± 0.3371)	8.1112	6min 3s
SES+HAC+LightGBM	2	0.1	8.3030 (± 0.3304)	8.3267	12min 57s
DTW+HAC+LightGBM	5	N/A	8.0954 (± 0.3378)	8.0368	20h 24min 5s

without SES with respect to the mean RMSE. Comparing CPU run times we observe that while DTW+HAC+LightGBM is able to result in similar forecasting performance, its CPU run time is multiplied by a factor of 246.

Figure 5 illustrates the mean RMSE difference per cluster as a percentage compared to the mean RMSE of the baseline model (top row). The bottom row illustrates the size of each cluster. Bars are coloured according to the mean intermittency in that cluster. Looking at Figure 5, we observe that on a cluster level, in four out of five clusters the use of DTW+HAC were not able to produce meaningful clusters to reduce the mean RMSE. The SOM was able to reduce the mean RMSE on a cluster level for moderate (0.5) to low (0.2) average intermittency levels within a cluster. Cluster 2 and 8, having very few zero recorded sales, did not benefit from a clustered approach with respect to the mean RMSE. In contrast to DTW, the SOM was able to reduce the mean RMSE for more intermittent clusters.

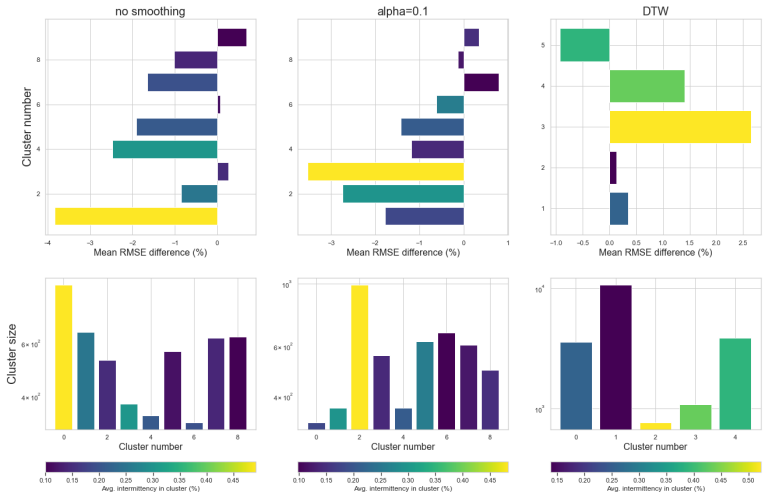

Fig. 5: Top row: mean RMSE difference across 5 folds between the baseline model (LightGBM) and the cluster-based forecasting method (SOM) on the M5 dataset. Bottom row: number of SKUs per cluster.

Figures 6 and 7 illustrate an example forecast for an intermittent and non-intermittent time series. A clear distinction can be made between the different smoothing values. HAC is used to cluster the distance matrix made by DTW.

5 Discussion

Clustering time series dataset into disjoint clusters and training individual models on those clusters can lead to a reduction in the forecasting error [13,9]. A common and accurate method to measure the similarity between two time series is DTW [18], but it comes at a cost: it is a computationally expensive measure to use on large datasets and on intermittent time series, DTW may not be the best suited measure to use, since intermittent time series show no clear structure. Our results show that by applying SES as a preprocessing method, clustering and forecasting performance can be improved in combination with SOM. While not achieving the same level of performance as DTW+HAC, it does come at a substantially lower computational cost. In practice, it is then up to the forecast practitioner to decide whether the focus should lie on speed, for instance in real-time forecasting, or on accuracy when for example weekly forecasts are produced. A limitation of using SES instead of DTW is that during clustering the

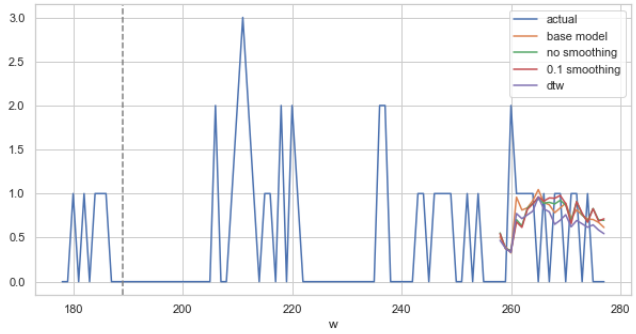

Fig. 6: Example forecast on an intermittent time series from the M5 dataset. Time series are clustered with SOM.

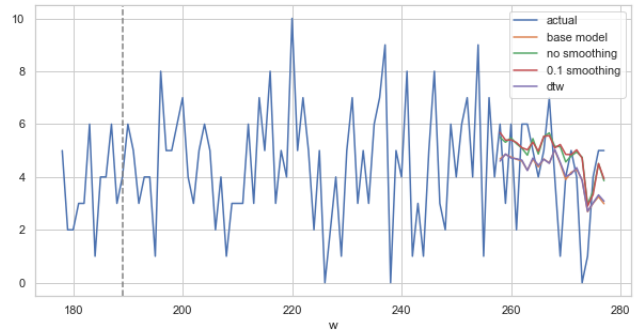

Fig. 7: Example forecast on a non-intermittent time series from the M5 dataset. Time series are clustered with SOM.

Euclidean distance is still used, making it not useful for time series of different length or for time series that are far apart from each other.

On a cluster level, the improvements in accuracy gained for intermittent time series seems to come at a cost of a higher forecasting error for less intermittent time series. This can be explained by the size of the produced clusters. Smaller clusters tend to perform worse than the base model because of insufficient training examples. LightGBM may also not be the best suited model to use on smaller clusters as it is sensitive to overfitting, especially on small datasets. Other models such as XGBoost or Random Forest in combination with recursive feature elimination could be used on the smaller clusters to mitigate the problem of overfitting [6].

The use of HAC, with and without SES, has almost no impact on the forecasting error, where HAC was unable to produce any meaningful clusters. HAC

also suffers from *rich get richer* behaviour, which explains the uneven cluster sizes. This may also explain why on the M5 dataset, SES+HAC+LightGBM was unable to reduce the mean RMSE. A cluster containing almost all the time series does not differ much from the original dataset. Performance from a model trained on that cluster will then be similar to the performance of a single model on the complete dataset.

6 Conclusions and Future Work

Producing forecasts for intermittent and non-intermittent time series is a trade-off between computational complexity and accuracy. We have explored alternatives to Dynamic Time Warping that are less computationally expensive and have shown that a combination of Simple Exponential Smoothing and a Self-Organizing Map is able to reduce the forecast error and provide similar results to Dynamic Time Warping in combination with Hierarchical Agglomerative Clustering. Especially for intermittent time series, the use of clustering has a positive effect on the forecasting accuracy. While the use of SES as a preprocessing step was able to increase cluster performance, its effect on the forecasting accuracy was minimal. Further experimentation should clarify if other versions of Exponential Smoothing are able to decrease the forecasting error. We have also shown that LightGBM can work as a model choice when forecasts are produced for separate clusters. Future work could also explore the options of combining forecasts from a single model and a clustered model to achieve better results for both intermittent and non-intermittent time series.

References

1. Aghabozorgi, S., Seyed Shirkhorshidi, A., Ying Wah, T.: Time-series clustering â€“ a decade review. Information Systems **53**, 16–38 (2015). https://doi.org/https://doi.org/10.1016/j.is.2015.04.007, https://www.sciencedirect.com/science/article/pii/S0306437915000733
2. Ariannezhad, M., Schelter, S., de Rijke, M.: Demand forecasting in the presence of privileged information. In: AALTD: International Workshop on Advanced Analytics and Learning on Temporal Data. pp. 46–62. LNCS 12588, Springer (December 2020)
3. Chen, I.F., Lu, C.J.: Sales forecasting by combining clustering and machine-learning techniques for computer retailing. Neural Computing and Applications **28**(9), 2633–2647 (2017). https://doi.org/10.1007/s00521-016-2215-x
4. Contreras, P., Murtagh, F.: Hierarchical clustering. Handbook of Cluster Analysis (February), 103–124 (2015). https://doi.org/10.1201/b19706
5. Croston, J.D.: Forecasting and stock control for intermittent demands. Operational Research Quarterly (1970-1977) **23**(3), 289–303 (1972), http://www.jstor.org/stable/3007885
6. Darst, B.F., Malecki, K.C., Engelman, C.D.: Using recursive feature elimination in random forest to account for correlated variables in high dimensional data. BMC genetics **19**(1), 1–6 (2018)

7. Holt, C.C.: Forecasting seasonals and trends by exponentially weighted moving averages. International Journal of Forecasting **20**(1), 5–10 (2004). https://doi.org/https://doi.org/10.1016/j.ijforecast.2003.09.015, https://www.sciencedirect.com/science/article/pii/S0169207003001134
8. Jha, A., Ray, S., Seaman, B., Dhillon, I.S.: Clustering to Forecast Sparse Time-Series Data (2015)
9. Kamini, V., Vadlamani, R., Prinzie, A., Van den Poel, D.: Cash demand forecasting in atms by clustering and neural networks. European Journal of Operational Research **232**, 383–392 (01 2014). https://doi.org/10.1016/j.ejor.2013.07.027
10. Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., Liu, T.Y.: Lightgbm: A highly efficient gradient boosting decision tree. In: Advances in neural information processing systems. pp. 3146–3154 (2017)
11. Kohonen, T.: Self-organized formation of topologically correct feature maps **69**, 59–69 (1982)
12. Kohonen, T.: Self-organizing feature maps. In: Self-organization and associative memory, pp. 119–157. Springer (1989)
13. Lu, C.J., Kao, L.J.: A clustering-based sales forecasting scheme by using extreme learning machine and ensembling linkage methods with applications to computer server. Engineering Applications of Artificial Intelligence **55**, 231–238 (2016). https://doi.org/10.1016/j.engappai.2016.06.015, http://dx.doi.org/10.1016/j.engappai.2016.06.015
14. Lu, C.J., Wang, Y.W.: Combining independent component analysis and growing hierarchical self-organizing maps with support vector regression in product demand forecasting. International Journal of Production Economics **128**(2), 603–613 (feb 2010). https://doi.org/10.1016/j.ijpe.2010.07.004
15. Makridakis, S., Spiliotis, E., Assimakopoulos, V.: Statistical and machine learning forecasting methods: Concerns and ways forward. PloS one **13**(3), e0194889 (2018)
16. Makridakis, S., Spiliotis, E., Assimakopoulos, V.: The M5 Accuracy Competition: Results, Findings and Conclusions (October), 1–44 (2020), https://www.researchgate.net/publication/344487258
17. Nikolopoulos, K.: We need to talk about intermittent demand forecasting. European Journal of Operational Research **291**(2), 549–559 (2021). https://doi.org/10.1016/j.ejor.2019.12.046
18. Paparrizos, J., Gravano, L.: Fast and accurate time-series clustering. ACM Transactions on Database Systems **42**(2) (2017). https://doi.org/10.1145/3044711
19. Petropoulos, F., Kourentzes, N.: Forecast combinations for intermittent demand. Journal of the Operational Research Society **66**(6), 914–924 (2015). https://doi.org/10.1057/jors.2014.62
20. Puspita, P.E., Ä°nkaya, T., Akansel, M.: Clustering-based Sales Forecasting in a Forklift Distributor. UluslararasÄ± Muhendislik Arastirma ve Gelistirme Dergisi pp. 1–17 (feb 2019). https://doi.org/10.29137/umagd.473977
21. Seaman, B.: Considerations of a retail forecasting practitioner. International Journal of Forecasting **34**(4), 822–829 (2018). https://doi.org/10.1016/j.ijforecast.2018.03.001, https://doi.org/10.1016/j.ijforecast.2018.03.001
22. Shalizi, C.: Distances between Clustering , Hierarchical Clustering. Data Mining (September), 36–350 (2009), www.stat.cmu.edu/$\sim$cshalizi/350
23. Smyl, S.: A hybrid method of exponential smoothing and recurrent neural networks for time series forecasting. International Journal of Forecasting **36**(1), 75–85 (2020). https://doi.org/10.1016/j.ijforecast.2019.03.017, https://doi.org/10.1016/j.ijforecast.2019.03.017

24. Syntetos, A.A., Boylan, J.E.: On the bias of intermittent demand estimates. International journal of production economics **71**(1-3), 457–466 (2001)
25. Syntetos, A.A., Boylan, J.E.: The accuracy of intermittent demand estimates. International Journal of forecasting **21**(2), 303–314 (2005)
26. Syntetos, A.A., Zied Babai, M., Gardner, E.S.: Forecasting intermittent inventory demands: Simple parametric methods vs. bootstrapping. Journal of Business Research **68**(8), 1746–1752 (2015). https://doi.org/10.1016/j.jbusres.2015.03.034, http://dx.doi.org/10.1016/j.jbusres.2015.03.034
27. Tsay, R.S.: Time series and forecasting: Brief history and future research. Journal of the American Statistical Association **95**(450), 638–643 (2000), http://www.jstor.org/stable/2669408
28. Van Craenendonck, T., Blockeel, H.: Using internal validity measures to compare clustering algorithms. Icml pp. 1–8 (2015)
29. Xu, S., Chan, H.K., Châ€™ng, E., Tan, K.H.: A comparison of forecasting methods for medical device demand using trend-based clustering scheme. Journal of Data, Information and Management **2**(2), 85–94 (feb 2020). https://doi.org/10.1007/s42488-020-00026-y
30. Zhou, H., Yang, Y., Qian, W.: Tweedie gradient boosting for extremely unbalanced zero-inflated data (2019)