Probabilistic forecasting of seasonal time series Combining clustering and classification for forecasting

Colin Leverger, Thomas Guyet, Simon Malinowski, Vincent Lemaire, Alexis Bondu, Laurence Rozé, Alexandre Termier, Régis Marguerie

Probabilistic forecasting of seasonal time series
Combining clustering and classification for forecasting

Colin Leverger[1,2], Thomas Guyet[3], Simon Malinowski[4], Vincent Lemaire[2], Alexis Bondu[2], Laurence Rozé[5], Alexandre Termier[4], and Régis Marguerie[2]

[1] IRISA
F-35042 Rennes Cedex, France
[2] Orange Labs
[3] Institut Agro/IRISA
[4] Université Rennes 1/Inria/IRISA
[5] INSA/Inria/IRISA

Abstract. In this article, we propose a framework for seasonal time series probabilistic forecasting. It aims at forecasting (in a probabilistic way) the whole next season of a time series, rather than only the next value. Probabilistic forecasting consists in forecasting a probability distribution function for each future position. The proposed framework is implemented combining several machine learning techniques 1) to identify typical seasons and 2) to forecast a probability distribution of the next season. This framework is evaluated using a wide range of real seasonal time series. On the one side, we intensively study the alternative combinations of the algorithms composing our framework (clustering, classification), and on the other side, we evaluate the framework forecasting accuracy. As demonstrated by our experiences, the proposed framework outperforms competing approaches by achieving lower forecasting errors.

Keywords: Time series, Probabilistic forecasting, Seasonality

1 Introduction

Forecasting the evolution of a temporal process is a critical research topic, with many challenging applications. In this work, we focus on time series forecasting and on data-driven forecasting models. A time series is a timestamped sequence of numerical values, and the goal of forecasting is, at a given point of time, to predict the next values of the time series based on previously observed values and possibly on other linked exogenous observations. Data-driven algorithms are used to predict future time series values from past data, with models that are able to adapt automatically to any type of incoming data. The data science challenge is to learn accurate and reliable forecasting models with as few human interventions as possible. Time series forecasting has many applications in medicine (for instance, to forecast blood glucose of a patient [1]), in economy (for instance, to forecast macroeconomic variable changes [2]), in the financial domain (forecasting financial time series [3]), in electricity load [4] or in industry (for instance, to forecast the server load [5,6]).

Time series forecasting algorithms provide information about possible situations in the future, and can be used to anticipate crucial decisions. Taking correct decisions requires anticipation and accurate forecasts. Unfortunately, these objectives are often contradictory. Indeed, the larger the forecasting horizon, the wider the range of expectable situations. In such case, a probabilistic forecasting algorithm is a powerful decision support tool, because it handles the uncertainty of the predictions. Probabilistic or density forecasting is a class of forecasting that provides intervals or probability distributions as outcomes of the forecasting. It is claimed in [7] that, in recent years, probabilistic forecasts have become widely used. For instance, fan charts [8], highest density regions [9] or functional data analysis [10] enable to forecast ranges for possible values of future data.

We are particularly interested in time series that have some periodic regularities in their values. This kind of time series is said to be seasonal. For instance, time series related to human activities or natural phenomena are often seasonal, because they often exhibit daily regularities (also known as the circadian cycle). Knowing that a time series is seasonal is a valuable information that can help for forecasting. More specifically, learning the seasonal structures can help to generate longer-term predictions as it provides information about several seasons ahead.

Furthermore, seasonality of a time series gives a natural midterm forecasting horizon. Classical forecasting models (*e.g.*, SARIMA [11]) predict the future values of a given time series stepwise. The predicted values are used by further steps. At each step, there is then a risk of the error to be accumulated due to the recursive nature of the forecasts. The prediction of a whole season at once aims at spreading the forecasting error all along the season. Thus, we expect to forecast more accurately the salient part of a season that may lie in the middle of the season. More practically, the prediction of a whole season at once allows applications where such prediction is required to plan actions (*e.g.*, to plan electricity production a day ahead, it is necessary to predict the consumption for the next 24 hours).

A second limitation of usual seasonal forecasting methods is the assumption that the seasons have the same shape, *i.e.*, the values evolve in the same way over the season. The differences are with each other are due to noise and an additive constant. Nevertheless, most of the real seasonal time series often contain more than just one periodic pattern. For instance, daily connections to a given website exhibit different patterns for a weekday or for a Sunday for instance. This kind of structure cannot be well captured by classical forecasting methods.

In this article, we propose a generic framework called P-F2C (which stands for "Probabilistic Forecasting with Clustering and Classification") for seasonal time series forecasting. This approach extends the F2C framework [6] (which stands for "Forecasting with Clustering and Classification"). P-F2C predicts future values for a complete season ahead at once, and this in a probabilistic manner. The P-F2C predictions may be used for supporting decision-making about the next season, handling the uncertainty in the future through the probabilistic presentation of the result.

2 Probabilistic seasonal time series forecasting

In this section, we introduce the notations and the problem of seasonal time series forecasting.

2.1 Seasonal time series

A time series Y is an ordered sequence of values $y_{0:n} = y_0, \ldots, y_{n-1}$, where $\forall i \in [0, n-1]$, $y_i \in \mathbb{R}$ (univariate time series). n denotes the length of the observed time series.

Y is said to be (ideally) *seasonal* with season length s if there exists $\mathcal{S} = \{S^1, \ldots, S^p\}$ a finite collection of p sub-series (of length s) called *typical seasons* such that

$$\forall i \in [0, m-1]\,,\; y_{(s\times i):s\times(i+1)} = \sum_{j=1}^{p} \sigma_{i,j} S^j + \boldsymbol{\varepsilon}_i \tag{1}$$

where m is the number of seasons in the time series, $\boldsymbol{\varepsilon}_i \in \mathbb{R}^s$ represents a white noise and $\sum_j \sigma_{i,j} = 1$ for all j. In other words, it means that for a seasonal time series Y, every season in Y is a weighted linear combination of typical seasons. Intuitively, this modelling of a typical season corresponds to additive measurements (*e.g.*, consumption or traffic) for which the observed measure at time t is the sum of individual behaviours. In this case, a typical season corresponds to a typical behaviour of individuals, and the $\sigma_{,j}$ represents the proportion of individuals of type j contributing to the observed measure.

In the following $\boldsymbol{y}_i = y_{(s\times i):s\times(i+1)} \in \mathbb{R}^s$ denotes the i-th season of Y.

2.2 Seasonal probabilistic forecasting

Let $Y = y_0, \ldots, y_{n-1}$ be a seasonal time series, an s be its season length. Note that the season length of a time series (s) is estimated using Fisher's g-statistics [12]. Without loss of generality, we assume that the length of a time series is a multiple of the season length, *i.e.*, $n = m \times s$. m denotes the number of seasons in the observed time series. The goal of seasonal probabilistic forecasting is to estimate

$$\Pr(y^*_{n:n+s} \mid y_{(n-\gamma\times s):n}) = \Pr(\boldsymbol{y}^*_m \mid \boldsymbol{y}_{(m-\gamma):m}) \tag{2}$$

where $\boldsymbol{y}^*_m = y^*_{n:n+s}$ are the forecasts of the s next values (next season) of the observed time series, and $\boldsymbol{y}_{(m-\gamma):m} = y_{(n-\gamma\times s):n}$ are the observed values of the last γ seasons. γ is a parameter given by the user.

We now propose an equivalent formulation of this problem considering our hypothesis on seasonal time series and we denote $\mathcal{S} = \{S^1, \ldots, S^p\}$ the set of p typical seasons. Thus, Equation 2 can be rewritten as follows:

$$\Pr(\boldsymbol{y}^*_m \mid \boldsymbol{y}_{(m-\gamma):m}) = \sum_{S\in\mathcal{S}} \Pr(\boldsymbol{y}^*_m \mid S).\Pr(S \mid \boldsymbol{y}_{(m-\gamma):m}) \tag{3}$$

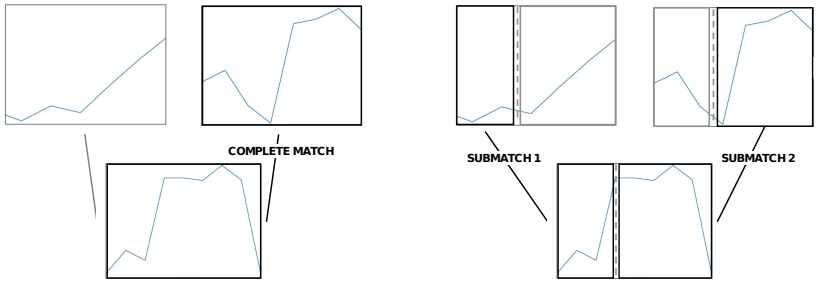

Fig. 1: Difference between clustering (on the left), which matches the entire time series, with coclustering (on the right), which is able to match subintervals of the time series of various other time series.

where $\Pr(\boldsymbol{y}_m^* \mid S)$ is the probability of having $\boldsymbol{y}_m^*$ given that the type of the next season and $\Pr(S \mid \boldsymbol{y}_{(m-\gamma):m})$ is the probability that the next season is of type S given past observations.

The problem formulation given by Eq. 3 turns the difficult problem of Eq. 2 into two well-known tasks in time series analysis:

- estimating the first term, $\Pr(\boldsymbol{y}_m^* \mid S)$ leads to a problem of time series clustering. The problem is to both define the typical seasons, $\mathcal{S}$, and to have the distributions of the season values. A clustering of the seasons $(\boldsymbol{y}_i)_{i=0:m}$ of the observed time series identifies the typical seasons (clusters) and gives the required empirical distributions $\hat{\Pr}(\boldsymbol{y}, S)$.
- estimating the second $\Pr(S \mid \boldsymbol{y}_{m-\gamma:m})$ is a probabilistic time series classification problem. This distribution can be empirically learnt from the past observations $(\boldsymbol{y}_{i-\gamma:i}, S_{i+1}^*)_{i=\gamma:m}$ where S_i^* denotes the empirical type of the i-th season obtained from the clustering assignment above.

This problem formulation and remarks sketch the principles of a probabilistic seasonal time series forecasting. P-F2C is an implementation of these principles with a specific time series clustering.

3 The P-F2C forecaster

P-F2C is composed of a clusterer that models the latent typical seasons and a classifier that predicts the next season type given the recent data. The forecaster is fit on the historical data of a time series. Then, the forecaster can be applied on the time series to predict the next season(s).

P-F2C clusterer is based on a probabilistic co-clustering model that is presented in the next section. In Section 3.2, we present how to use classical classifiers to predict the next seasons.

3.1 Coclustering of time series: a probabilistic model

Coclustering is a particular type of unsupervised algorithm which differs from regular clustering approaches by creating co-clusters. The objective of coclustering approaches consists in simultaneously partitioning the lines and the columns of an input data table. Thus, a co-cluster is defined as a set of examples belonging to both a group of rows and a group of columns. In [13], Boullé proposed an extension of co-clustering to tri-clustering in order to cluster time series. In this approach, a time series with an identifier C is seen as a set of couples (T, V), where T is a timestamp and V a value of a measurement. Thus, the whole set of time series is a large set of points represented by triples (C, T, V). The tri-clustering approach handles the three variables (C is categorical and T, V are numerical) to create homogeneous groups. A co-cluster gathers time series (group of identifiers) that have similar values during a certain interval of time. Contrary to the classical clustering approaches (*e.g.,* KMeans, K-shape, GAK) [14] that are based on the entire time series, the coclustering approach uses a local criterion. This difference is illustrated in Figure 1: A distance based clustering (on the left) evaluates the distance between whole time series, in the co-clustering approaches, the distance is based on subintervals of the seasons. This enables to identify which parts of the season are the most discriminant. Besides, tri-clustering is robust to missing values in time series.

The tri-clustering approach of Boullé is based on the MODL framework [10]. The MODL framework makes a constant piecewise assumption to estimate the joint distribution $\Pr(C, T, V)$ by jointly discretising the variables T, V and grouping the time series identifiers of the variable C. The resulting model consists of the Cartesian product[6] of the three partitions of the variables C, T, V. This model can be represented as a 3D grid (see Figure 2, on the left). In this 3D grid, if one considers a given group of time series (*i.e.*, a given group of C), the model provides a bivariate discretisation which estimates $\Pr(T, V \mid C) = \frac{\Pr(C,T,V)}{\Pr(C)}$ as a 2D grid (see Figure 2, on the right). This 2D grid gives the probability to have a given range of values during a given interval of time. Therefore knowing that a time series belongs to a given cluster the corresponding 2D grid may then be used for crafting forecasts (see next section).

In the MODL approach, finding the most probable tri-clustering model is turning into a model selection problem. To do so, a Bayesian approach called Maximum A Posteriori (MAP) is used to select the most probable model given the data. Details about how this 3D grid model is learned may be found in [13,15]. The main idea could be summarised as finding the grid which maximises the contrast compared to a grid based on the assumption that T, V and C are independent (*i.e.*, $\Pr(V, T, C)$ compared to $\Pr(V)\Pr(T)\Pr(C)$). Therefore the estimation of this MAP model outputs: (i) ν intervals of values $V_i = [v_i^l, v_i^u]$ for $i = 1, \ldots, \nu$, (ii) τ intervals of times $T_i = [t_i^l, t_i^u]$ for $i = 1, \ldots, \tau$, (iii) groups of time series. These groups of time series corresponds to the typical seasons,

[6] The Cartesian product of the three partitions is used as a constant piecewise estimator – *i.e.*, a 3D histogram.

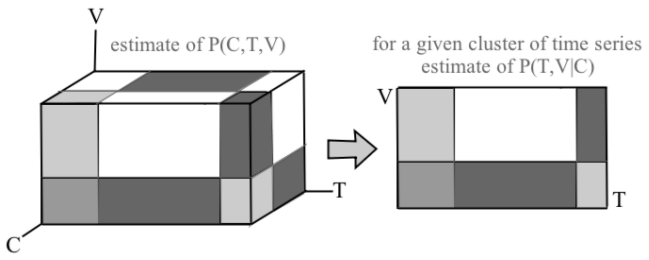

Fig. 2: Illustration of a trivariate coclustering model where a slice referred to forecasting "grid" is extracted.

denoted $\mathcal{S}$ in the above model. $|\mathcal{S}|$ is the number of clusters at the finer level that is optimal in the sense of the MODL framework.

In the time series forecasting approach proposed in this paper, the right number of (tri-)clusters is optimised regarding to the forecasting task. More precisely, this number is optimised according to the performance of the model at prediction time, using the validation ensemble. This value could differ from $|\mathcal{S}|$. Therefore the MODL coclustering approach allows applying a hierarchical clustering to the finer level to have a coarse level with a lower number of clusters called C^*, $C^* < |\mathcal{S}|$. A grid search selects the C^* value based on the forecast accuracy on the valid dataset.

Let us now come back to the formalisation of probabilistic time series forecasting: $\hat{\Pr}(\boldsymbol{y}_m^* \mid S)$ is estimated by the MODL model from the conditional probabilities $\Pr(V, T \mid C = S)$ where S denotes one of the time series groups, *i.e.* a typical season. In practice, the grid is used to estimate the distribution of values at each time point of a season. With MODL, the distribution is a piecewise constant function.

3.2 Predict the next type of seasons

The problem is here to estimate empirically $\Pr(S_{i+1} \mid \boldsymbol{y}_{(i-\gamma):i})$ the probability of having a type of season $S_{i+1} \in \mathcal{S}$ for the $(i+1)$-th season given the observations over the γ past seasons. We consider two different sets of features to represent the γ previous seasons. The first approach consists in having only the time series values $\boldsymbol{y}_{(i-\gamma):i}$ as features. The second approach uses the time series values and the types of the previous seasons as features.

Then, the next season prediction problem consists in learning a probabilistic classifier (Naive-Bayes classifier, logistic regression, decision tree or random forests) or a time series classifiers (TSForest [16], Rocket [17]). Note that time series classifiers can use only the time series values.

3.3 Select the best parameters (Portfolio)

The P-F2C forecaster is parameterised by the number of seasons in the past (γ) used for learning next season type, a maximum number of typical seasons to

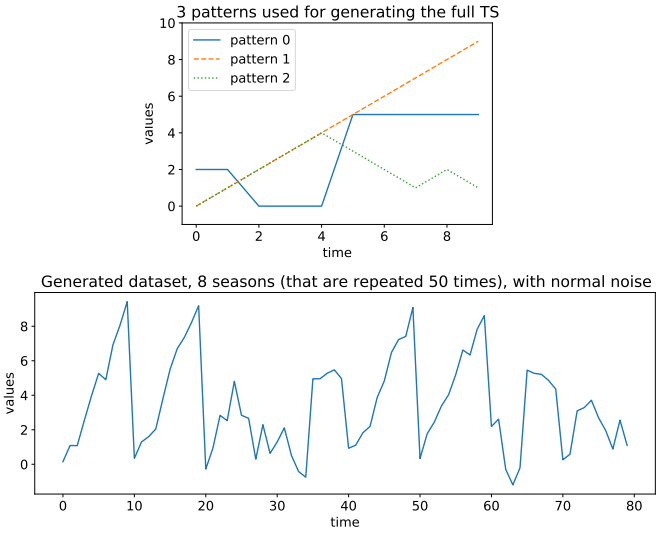

Fig. 3: At the top: typical seasons of length 10 used for generating the time series, at the bottom: examples of generated time series with white noise (7 seasons).

detect in a non-supervised way, and the type of classifier. The γ parameter is introduced in the problem definition and its choice is left to the user who specifies what is the forecasting task. On the other hand, the other parameters may be difficult to be set by the user, and we do not think that one of the classifiers will outperform the others for all the time series. For these reasons, the portfolio approach (denoted PP-F2C) implements a grid search for the best parameters by splitting the dataset into a training (75%) and a validation dataset (25%) to identify the best value of the parameters. Once the best values have been set, the clusterer and the classifier are fitted on the entire dataset.

4 Illustration on a synthetic dataset

This section shows results with synthetic data. The goal is to illustrate the probabilistic grid used in P-F2C method, and to give intuitions behind probabilistic forecasting that are provided by P-F2C. We compare the output of P-F2C against the output of DeepAR [18], a state-of-the-art probabilistic time series forecaster.

4.1 The data generated

Generating data is a good strategy for checking assumptions before launching experiments at scale. Indeed, the shape of the generated data is often simpler, and completely controlled. Experiments may be executed with various parameters, to plot understandable results and to validate basic expectations.

The seasonal data generated for this section follows some well-established seasonal sequences. Three different time series patterns are defined for three different latent types of season of length 10. In the Figure 3, one type of season (s_1 in orange) with always increasing values is observed, one type of season (s_2 in green) with two peaks is observed, etc. Those three different types of season are then repeated 50 times in a defined order ($s_1, s_1, s_0, s_2, s_1, s_1, s_2, s_0$, as observed in Figure 3, on the right, which shows the entire sequence that is being repeated), and noise is added to the final time series to make the forecasting process less straightforward.

4.2 Grid probabilistic forecasts

Once trained, we apply the P-F2C forecaster at the end of the time series illustrated in Figure 3 on the right. Knowing the sequence of patterns, we can guess that a season of type s_2 is coming ahead. Indeed, the last three patterns seems to follow the sequence $[s_1, s_1, s_0]$.

The Figure 4 shows two examples of forecasts with different values of γ.

The real values of the predicted time series are in blue (noisy version of the s_2 pattern). The probabilistic forecasts are shown in a red overlay. It is a set of rectangles that visualise the homogeneous regions that have been identified by MODL coclustering. The darker the red, the more probable next season ahead lay in this (T, V) interval.

The Figure 4 on the left is the forecast obtained with $\gamma = 1$. It illustrates a probabilistic forecast with a lot of uncertainty. Indeed, light red cells are observed in the figure where the data are predicted to lay (with a low probability). In this case, the classifier is unable to predict accurately the next type of season. With $\gamma = 1$ the classifier has only the information of the preceding season (of type s_0). In this case, the forecaster encountered two types of season after a s_0 season: s_1 or s_2 with the same probability. Then, the predicted grid is a mixture of the two types of grids. For the first half of the season, the forecast is confident in predicting the linear increase of the value (darker red cells), but for the second half, the forecast suggests two possible behaviours: continue the linear increase (s_1) or a decrease (s_2). Note that the grids of all typical seasons share the same squaring. MODL necessarily creates the same cuttings of a dimension (V or T) along the others (C).

The Figure 4 on the middle is the forecast obtained with $\gamma = 3$. It illustrates a good probabilistic forecast. The real values (in blue) often appear in the red boxes where the red is very dark. It means that the season type was both well described by MODL and well predicted by the classifier. In this case, a larger memory of the forecaster disentangles the two possible choices it had above. After a $[s_1, s_1, s_0]$, the forecaster always observed seasons of type s_2. Thus, the grid of this pattern is predicted.

It is worth noting that, for $\gamma = 1$, the use of the MODL probabilistic grid suggests two distinct possible evolution of the time series, but there is an uncertainty on which evolution will actually occur. In the classical probabilistic

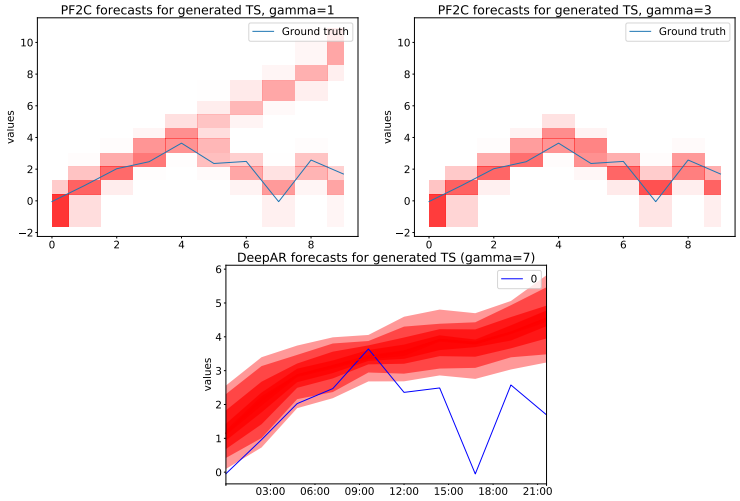

Fig. 4: One season ahead grid forecasts for the generated time series with $\gamma = 1$ at the top left and $\gamma = 3$ at the top right, and DeepAR at the bottom.

forecasts, probabilities are distributed around a mean time series. This is illustrated on the Figure 4 on the right with DeepAR using the 7 seasons in the past to predict the next season. On the second half of the season, the predicted probabilistic distribution suggests a behaviour in between s_1 and s_2 with a larger uncertainty. Such model makes confusion between uncertainty of behaviour and imprecise forecast. In the case of seasonal time series with different types of season, the mean time series has no meaning for an analyst.

5 Experiments

This section presents experiments to assess the accuracy of P-F2C. We start by introducing the experimental settings, then we investigate some parameters of our model and finally we present the result of an intensive comparison of P-F2C to competitors.

5.1 Experimental protocol

The framework has been developed in Python 3.5. The MODL coclustering is performed by the Khiops tool [19]. The classification algorithms are borrowed from the *sklearn* library [20].

In our experiments, we used 36 datasets[7], from various sources and nature: technical devices, human activities, electricity consumption, natural processes,

[7] Datasets details can be downloaded here: `https://tinyurl.com/4kffdwhc` (temporary link). It includes the sources of time series.

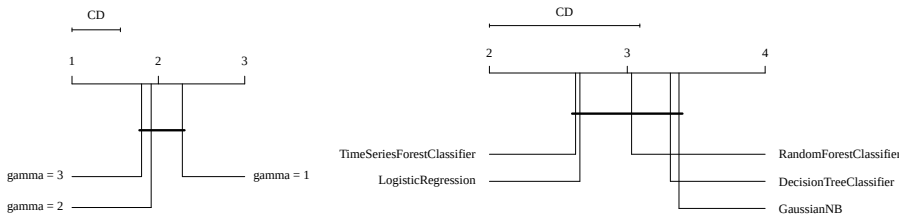

Fig. 5: Critical diagrams used to find the best parameters for the P-F2C implementation.

etc. All these datasets have been selected because seasonality was identified and validated with a Fisher g-test [12]. Each time series is normalised using a z-normalisation prior to data splitting, in order to have comparable results. For the experiments, 90% of the time series are used to train the forecaster (this train test is internally split in training and valid datasets) and 10% of the original time series are used to evaluate the accuracy.

P-F2C and PP-F2C are compared with classical deterministic time-series forecasters (AR, ARIMA, SARIMA, HoltWinters), with LSTM [21], Prophet [22] and with the F2C method [6] which uses the principles as P-F2C but with K-means clustering algorithm and random forest classifiers to learn the structure in the season sequence. P-F2C being a probabilistic methodology, we also compare it with DeepAR [18].

We use Mean Absolute Error (MAE) and Continuous Ranked Probability Score (CRPS) to compare the forecasts to the real time series. The MAE is dedicated to deterministic forecasts while CRPS is to probabilistic ones. It is worth noting that the CRPS is analogous to MAE for deterministic forecasts. Therefore, comparing MAE measure for deterministic forecasts against CRPS values for probabilistic forecasts is technically sound [23]. The CRPS is used for DeepAR and P-F2C. All the other approaches forecast crisp time series and their accuracy is evaluated through MAE. For each experiment, we illustrate the results with critical difference diagrams. A critical difference diagram represents the mean rank of the methods that have been obtained on the set of the 36 times series. The lower the better. In addition, the representation shows horizontal bars that group some methods. In a same group, the methods are not statistically different according to the Nemenyi test.

5.2 Parameters sensitivity

In this section, an analysis of the alternative settings of the P-F2C methodology is conducted. We investigate the effect of two choices: the choice of the γ value, *i.e.*, the number of seasons to consider in the history; and the choice of the classifier to predict the next type of season in case we do not use the portfolio optimisation.

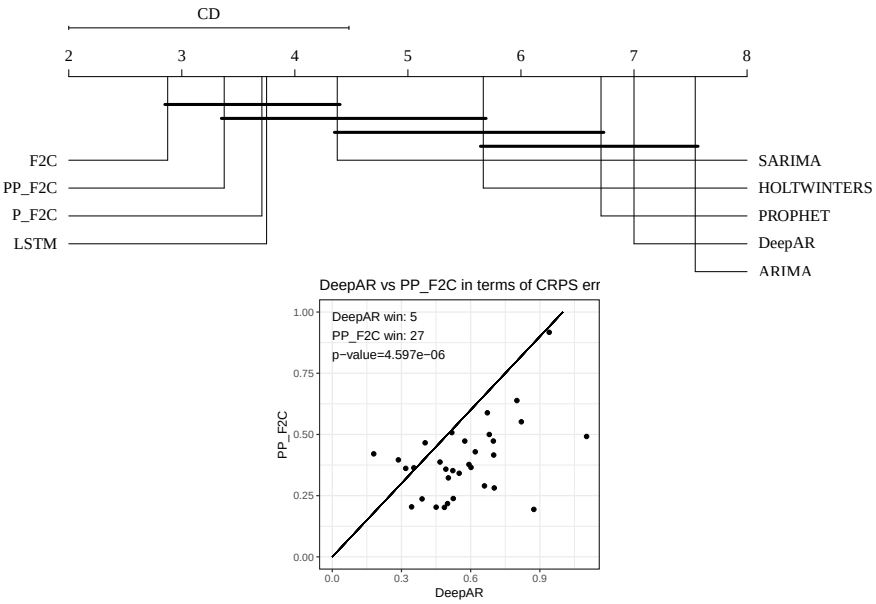

Fig. 6: At the top: Critical diagram of the comparison between different prediction approaches (acronyms of method are detailed in the text). At the bottom: Win-Tie-Loose graph between PP-F2C and DeepAR.

Figure 5 on the left shows a critical diagram that compares the ranking of P-F2C with different values of γ (1, 2 or 3). For this experiment, the classifier is the RandomForestClassifier (and we had the same results with the other classifiers). We notice that the larger γ, the lower the error. Indeed, as seen in Section 4, larger γ improves the accuracy of the forecast of the next season type. Nonetheless, we observed that for some time series, lower γ may be better. We explain this counter-intuitive results by the small length of some of the time series. In the cases, the number of seasons in the training set is too small to fit the numerous parameters of a classifier with $\gamma \times s$ features.

Figure 5 on the right shows a critical diagram that compares the classifiers used to predict the next type of season. It shows that time series forest classifier [16] is on average in first position. This classifier has been designed specifically for time series classification, it explains why it outperforms the other approaches. Nonetheless, the differences with Logistic Regression and Random Forest are not statistically significant. Their capability to use extra-information, such as the type of seasons, may be an interesting advantage to improve performances.

5.3 P-F2C and PP-F2C vs opponents

The critical diagram of Figure 6 compares the performances of the methods. P-F2C denotes our approach configured with the best parameters on average

found in Section 5.2. PP-F2C denotes P-F2C that is optimised on the valid test for each dataset (portfolio). It shows that rank-wise, the seasonal forecaster F2C, P-F2C and PP-F2C are performing better than the others. We can first notice that the portfolio actually improve the performances of P-F2C. Nonetheless, the non-probabilistic approach outperform PP-F2C.

We also notice that F2C outperforms PP-F2C. Even if a PP-F2C forecast fits the time series (see Figure 4), the piece-wise approximation generates a spread of the probabilistic distribution that penalises the CRPS. Nonetheless, it is worth noting that the rank difference with F2C is not statistically significant, and that probabilistic forecast convey meaningful information to trust the forecasts.

Then, we compared PP-F2C with another probabilistic forecaster, *i.e.* DeepAR. The critical diagram of Figure 6 shows that PP-F2C outperforms DeepAR significantly ($p < 10^{-6}$). The win/tie/lose graph on the right shows how many times PF2C won against DeepAR (points below the diagonal) and the relative values of CRPS. The point positions illustrate that PP-F2C outperforms DeepAR significantly on most of the datasets.

6 Conclusion

P-F2C is a probabilistic forecaster for seasonal time series. It assumes that seasons are a mixture of typical seasons to transform the forecasting problem into both a clustering and a classification of time series. The P-F2C applies parameterless coclustering approach that generates grid forecasts, each typical grid being a typical seasonal behaviour. In addition we proposed PP-F2C that adjust P-F2C parameters for each time series. PP-F2C outperforms on average the competitors except F2C on various seasonal time series. F2C is based on the same principle as PP-F2C but is not probabilistic and parameterless. Nonetheless, we illustrated the interest of probabilistic grid forecasting to give information about uncertain distinct mean behaviours. Indeed, the probabilistic grid mixture is more interpretable than combining probabilistic distribution around a mean.

References

1. Liu, C., Vehí, J., Avari, P., Reddy, M., Oliver, N., Georgiou, P., Herrero, P.: Long-term glucose forecasting using a physiological model and deconvolution of the continuous glucose monitoring signal. Sensors **19**(19) (2019) 4338
2. Li, J., Chen, W.: Forecasting macroeconomic time series: Lasso-based approaches and their forecast combinations with dynamic factor models. International Journal of Forecasting **30**(4) (2014) 996–1015
3. Tay, F., Cao, L.: Application of support vector machines in financial time series forecasting. Omega **29**(4) (2001) 309–317
4. Laurinec, P., Lóderer, M., Lucká, M., Rozinajová, V.: Density-based unsupervised ensemble learning methods for time series forecasting of aggregated or clustered electricity consumption. Journal of Intelligent Information Systems **53**(2) (2019) 219–239

5. Bodìk, P.: Automating Datacenter Operations Using Machine Learning. PhD thesis, UC Berkeley (2010)
6. Leverger, C., Malinowski, S., Guyet, T., Lemaire, V., Bondu, A., Termier, A.: Toward a framework for seasonal time series forecasting using clustering. In: Proceedings of the International Conference on Intelligent Data Engineering and Automated Learning. (2019) 328–340
7. De Gooijer, J., Hyndman, R.: 25 years of time series forecasting. International journal of forecasting **22**(3) (2006) 443–473
8. Wallis, K.F.: Asymmetric density forecasts of inflation and the bank of england's fan chart. National Institute Economic Review **167**(1) (1999) 106–112
9. Hyndman, R.: Highest-density forecast regions for nonlinear and non-normal time series models. Journal of Forecasting **14**(5) (1995) 431–441
10. Boullé, M.: Data grid models for preparation and modeling in supervised learning. Hands-On Pattern Recognition: Challenges in Machine Learning **1** (2011) 99–130
11. Kareem, Y., Majeed, A.R.: Monthly peak-load demand forecasting for sulaimany governorate using SARIMA. In: Proceedings of the International Conference on Transmission & Distribution Conference and Exposition. (2006) 1–5
12. Wichert, S., Fokianos, K., Strimmer, K.: Identifying periodically expressed transcripts in microarray time series data. Bioinformatics **20**(1) (2004) 5–20
13. Boullé, M.: Functional data clustering via piecewise constant nonparametric density estimation. Pattern Recognition **45**(12) (2012) 4389–4401
14. Paparrizos, J., Gravano, L.: Fast and accurate time-series clustering. ACM Transactions on Database Systems (TODS) **42**(2) (2017) 1–49
15. Bondu, A., Boullé, M., Cornuéjols, A.: Symbolic representation of time series: A hierarchical coclustering formalization. In: International Workshop on Advanced Analysis and Learning on Temporal Data, Springer (2015) 3–16
16. Deng, H., Runger, G., Tuv, E., Vladimir, M.: A time series forest for classification and feature extraction. Information Sciences **239** (2013) 142–153
17. Dempster, A., Petitjean, F., Webb, G.I.: Rocket: Exceptionally fast and accurate time series classification using random convolutional kernels. arXiv:1910.13051 (2019)
18. Salinas, D., Flunkert, V., Gasthaus, J., Januschowski, T.: Deepar: Probabilistic forecasting with autoregressive recurrent networks. International Journal of Forecasting **36**(3) (2020) 1181–1191
19. Boullé, M.: Khiops: Outil d'apprentissage supervisé automatique pour la fouille de grandes bases de données multi-tables. In: Actes de la conférence Extraction et Gestion des Connaissances. (2016) 505–510
20. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al.: Scikit-learn: Machine learning in python. Journal of machine Learning research **12** (2011) 2825–2830
21. Gers, F.A., Schmidhuber, J., Cummins, F.: Learning to forget: Continual prediction with LSTM. In: Proceedings of the 9th International Conference on Artificial Neural Networks (ICANN). (1999) 850–855
22. Taylor, S., Letham, B.: Forecasting at scale. The American Statistician **72**(1) (2018) 37–45
23. Hersbach, H.: Decomposition of the continuous ranked probability score for ensemble prediction systems. Weather and Forecasting **15**(5) (2000) 559–570