Single-hidden layer neural networks for forecasting intermittent demand

F. Lolli[a*], R. Gamberini[a], A. Regattieri[b], E. Balugani[a], T. Gatos[b], S. Gucci[b]

[a] *Department of Sciences and Methods for Engineering, University of Modena and Reggio Emilia, Via Amendola 2 – Padiglione Morselli, 42100 Reggio Emilia, Italy*

[b] *Department of Industrial Engineering, University of Bologna, Viale Risorgimento 2, 40136 Bologna, Italy*

[*] Corresponding author: Ph.D. Francesco Lolli

E-mail: francesco.lolli@unimore.it

Tel.: +39 0522 522635

Single-hidden layer neural networks for forecasting intermittent demand

Abstract

Managing intermittent demand is a vital task in several industrial contexts, and good forecasting ability is a fundamental prerequisite for an efficient inventory control system in stochastic environments. In recent years, research has been conducted on single-hidden layer feedforward neural networks, with promising results. In particular, back-propagation has been adopted as a gradient descent-based algorithm for training networks. However, when managing a large number of items, it is not feasible to optimize networks at item level, due to the effort required for tuning the parameters during the training stage. A simpler and faster learning algorithm, called the extreme learning machine, has been therefore proposed in the literature to address this issue, but it has never been tried for forecasting intermittent demand. On the one hand, an extensive comparison of single-hidden layer networks trained by back-propagation is required to improve our understanding of them as predictors of intermittent demand. On the other hand, it is also worth testing extreme learning machines in this context, because of their lower computational complexity and good generalisation ability.

In this paper, neural networks trained by back-propagation and extreme learning machines are compared with benchmark neural networks, as well as standard forecasting methods for intermittent demand on real-time series, by combining different input patterns and architectures. A statistical analysis is then conducted to validate the best performance through different aggregation levels. Finally, some insights for practitioners are presented to improve the potential of neural networks for implementation in real environments.

Keywords: Intermittent Demand; Forecasting; Artificial Neural Networks; Extreme Learning Machines.

1. Introduction

Forecasting intermittent demand is a highly topical concern which arises in several real environments, such as spare parts, start-up productions and multi-echelon supply-chains. For instance, spare parts are typically consumed intermittently and have high unit purchase costs, which exacerbates the well-known trade-off in inventory control theory between holding and back-ordering costs. In such a stochastic contest, a prerequisite for effective inventory management is the adoption of accurate forecasting methods. However, research in forecasting such items has been limited (Syntetos et al., 2015).

The bias of the simple exponential smoothing as predictor of intermittent demand has led to the investigation of new methods, pioneered by Croston (1972). In order to obtain the estimator for the expected value of the demand per period, the guideline for the Croston-type approaches is to model intermittent demand as the result of two independent stochastic processes involving two stochastic variables, i.e. the positive demand and the demand occurrence. Readers can refer to Boylan and Syntetos (2010) for a review of intermittent demand forecasting, and Teunter et al. (2011) for a more recent forecasting procedure.

Conversely, the artificial neural networks (ANNs) have attracted the attention of a wide range of practitioners and researchers in recent years, due to their ability of learning a non-linear function from samples without the need for any distribution assumptions. In particular, single-hidden layer feedforward neural networks have been already investigated for forecasting intermittent demand in a few research projects (see Carmo and Rodrigues, 2004; Gutierrez et al., 2008; Nasiri Pour et al., 2008; Mukhopadhyay et al., 2012; and Kourentzes, 2013). All of these ANNs belong to the family of multi-layered perceptron networks, the most widely used networks in the demand forecasting field (Bishop, 1995). In particular, the use of a single-hidden layer for forecasting is according to the guidelines provided by Xiang et al. (2005). All of these ANNs, with the exception of Nasiri Pour et al. (2008), adopt feedforward architectures; this suggests that the experimental research into the use of ANNs for forecasting intermittent demand should also

be extended to time-delay and recurrent architectures, since they have been already applied to other fields successfully.
Moreover, only gradient descent-based algorithms, with back-propagation conventionally used, have been applied for training the aforementioned networks. However, these algorithms have several drawbacks that limit their applicability to real settings. The main ones are: i) slow convergence to the minimum of the error function through time-consuming iterative tuning of the hidden parameters (i.e. input and output weights and bias); ii) risk of being trapped in local minima; iii) need to set the learning parameters (e.g. learning rate and momentum); iv) need to establish the number of training epochs, as well as the proper stopping criterion against overfitting. In order to overcome these drawbacks, a faster learning algorithm proposed by Huang et al. (2006), i.e. the extreme learning machine, has more recently attracted the attention of researchers, with promising results in several fields. Readers can refer to the review of Huang et al. (2015). Nevertheless, extreme learning machines have never been applied for forecasting intermittent demand, even if they deserve attention, especially for their advantages for implementation in real settings.
In this paper, a new input pattern was adopted with the aim of helping the network to learn the temporal behaviour of the time series in terms of zero/non-zero demand. Three network architectures with this new input pattern were tested on 24 real time series, all trained both by back-propagation and by an extreme learning machine. The forecasting accuracy of these ANNs was compared to that achieved by other ANNs (i.e. Gutierrez et al., 2008; Mukhopadhyay et al., 2012) and that of two well-known estimators of intermittent demand (Croston, 1972; Syntetos and Boylan, 2005). This comparison was then enriched by adopting two different accuracy metrics on different time horizons. Such a detailed comparison aims at bridging the gap between theory and practice of ANNs in the field of intermittent demand. In fact, the potential for implementation of ANNs in real environments can only increase by providing useful guidelines about their design and training for practitioners. Finally, a statistical analysis of the networks' performance was conducted, for robust validation of the results.
The work is organised as follows: Section 2 contains a review of the main published contributions related with the presented research; in Section 3 benchmark forecasting methods are briefly explained; in Section 4 single-hidden layer neural networks are enounced, along with the tested architectures; Section 5 contains the approaches used for training the networks, i.e. back-propagation and extreme learning machine; in Section 6 the experienced networks are described, and the notation adopted for synthetically indicating them is presented; Section 7 clarifies the comparison approach; Section 8 refers to the experimental analysis with the parameter settings; Section 9 contains the statistical analysis performed on the results, with some suggestions for practitioners; and Section 10 concludes the paper with some potential directions for the future research agenda.

2. Research background

Several papers focus both on comparing ANNs to conventional forecasting approaches and introducing modifications to ANNs to achieve significant performance improvements. In order to offer a clear and concise classification of the main contributions of ANN applications to the forecasting field, this section reports the main advantages of ANNs compared to other time series forecasting approaches. It then reports on the specific application of ANN modelling to forecasting intermittent demand patterns.
Sharda and Patil (1992) conducted a forecasting M-competition (Makridakis et al., 1982) between ANNs and the Box-Jenkins ARIMA models (Auto Regressive Integrated Moving Average), achieving similar results for 75 time series. Taking 14 of the series dealt with by Sharda and Patil (1992), and Tang and Fishwick (1993) concluded that ANNs outperformed the Box-Jenkins ARIMA models for time series with a short memory or with greater irregularity, while for long memory series, both approaches achieved roughly the same performance. Kang (1991) achieved similar results through a more systematic study of 50 M-competition time series. He concluded that the best ANN is always better than the Box-Jenkins model. Moreover, ANNs performed better as the forecasting horizon increased and they needed less data to perform as well as

ARIMA. These results have been subsequently corroborated by several authors (Caire et al., 1992; Lachtermacher and Fuller, 1995; Kohzadi et al., 1996; Ho et al., 2002; Zhang and Qi, 2005; Hamzaçebi et al., 2009; West and Dellana, 2011).

In comparison with a wide variety of traditional forecasting approaches (i.e. exponential smoothing, regression-based and multivariate modelling), beneficial effects related to the use of ANNs were described in Chakraborty et al. (1992) and in Hill et al. (1994). In particular, in a later study, Hill et al. (1996) evaluated the results achieved by ANNs in 1001 time series in an M-competition, obtaining forecasts significantly better in terms of the mean absolute percentage errors (MAPEs) from ANNs. Denton (1995), after generating several datasets, observed that in ideal situations ANNs are as good as regression, while in less ideal situations ANNs perform better. Abdel-Aal (2008) emphasised the superiority of ANNs compared to multivariate modelling, which requires data for the exogenous factors influencing demand which may not be readily available.

Intermittent demand forecasting represents a specific research field for the peculiarities of the demand generation process, which has prompted several authors to address the competition between different approaches. Readers can refer to: Regattieri et al. (2005) on the comparison of forecasting approaches belonging to the categories of exponential smoothing and moving average; Gamberini et al. (2010) and Lolli et al. (2011, 2014a; 2014b) on the adoption of SARIMA models for intermittent profiles with trend and seasonality components; and Teunter and Duncan (2009), Teunter and Sani (2009), Wallström and Segerstedt (2010), and Babai et al. (2014) on the comparison of several variants of the original Croston's method.

Despite the large amount of papers that refer to the application of ANNs for time series forecasting, only a few works focus specifically on the application of single-hidden layer ANN modelling to forecasting intermittent demand profiles. Gutierrez et al. (2008) emphasise that traditional time series forecasting methods sometimes fail to capture non-linear patterns in data, and are therefore not effective in the case of intermittent demand patterns; artificial neural network modelling is thus a logical choice to overcome this limitation. They defined a network architecture characterized by two input neurons, then compared their forecasting accuracy with that obtained from the best-performing forecasting methods of those specifically used for intermittent demand patterns, i.e. single exponential smoothing, Croston's method (CR) and the Syntetos-Boylan Approximation (SBA) introduced by Syntetos and Boylan (2005), through detailed empirical analysis. Nasiri Pour et al. (2008) compared the performance of several ANNs (Gutierrez' network, generalized regression neural network-GRNN, recurrent neural network-RNN and a new hybrid network) with those obtained by the application of the SBA on 30 time series. Their new hybrid network outperformed the other networks. Mukhopadhyay et al. (2012) modified one of the two neurons of the input layer proposed by Gutierrez et al. (2008), applied dataset partitioning, then compared 24 time series of the aforementioned ANNs, using an optimized weighted moving average method. Other benchmark methods were applied, i.e. CR, SBA and SES, and the new ANN showed promising results. Kourentzes (2013) proposed two ANNs named NN-Dual and NN-Rate, representing bivariate models, allowing interaction between the demand and the inter-demand intervals. The comparison was performed taking CR method into account, and several of its variants. It was also considered in terms of inventory metrics, concluding that the ANNs had poor forecasting performance but high inventory cost savings.

With the exception of the recurrent architecture tested by Nasiri Pour et al. (2008), all the aforementioned contributions adopt feedforward architectures. This also indicates the opportunity of investigating time-delay neural networks, along with feedforward and recurrent networks. Moreover, the learning algorithms adopted in these contributions are gradient descent-based, whose drawbacks have been already highlighted. Huang et al. (2006) demonstrated that optimal output weights can be analytically obtained through a one-shot algorithm after the input weights and the bias of the hidden neurons are generated randomly before learning. This represents the core of the extreme learning machines. Actually, such a theoretical finding strongly simplifies the implementation of ANNs in real settings, because of the much lower computational efforts required. Extreme learning machines have been already applied to forecasting time series in several fields

(e.g. Sun et al., 2008; Lian et al., 2014; Mohammadi et al., 2015; Heinermann and Kramer, 2016; Ertugrul, 2016), but they have never been tested before for forecasting intermittent demand, despite the fact that these demand profiles are highly critical to predict, as well as relevant in many real industrial settings.
On the one hand, the main novelty of this contribution is the adoption of extreme learning machines in this context. On the other hand, it represents an extension of previous work in terms of architecture, with the aim of improving our understanding of the behaviour of neural networks as a predictor of intermittent demand, and thus encouraging practitioners to implement them accurately in real environments.

3. Intermittent demand estimators: CR and SBA

Croston (1972) proposed a method that considers both the demand size and the inter-arrival time between demands as the stochastic variables of a Bernoulli process; these variables are assumed to have constant means and variances and to be mutually independent. CR is based on two simple exponential smoothing, only when demand occurs, both of the demand size at the end of the review time period t (z_t), and of the interval between non-null demands (p_t). Thus, the equations are as follows:

$$z_t = \alpha d_t + (1-\alpha) z_{t-1} \quad (1)$$
$$p_t = \beta\, g_t + (1-\beta) p_{t-1} \quad (2)$$

where g_t is the actual value of the time between consecutive transactions at the instant t and α and β are the smoothing parameters.
If no demand occurs, then the smoothed estimates remain unchanged. If demand does occur, then the estimates are updated. If demand occurs in every time period, Croston's estimator is identical to a simple exponential smoothing for each time period.
The expected value of the demand, used as forecast for demand per period (D_{t+1}) at the end of time period *t*, is given by:

$$D_{t+1} = \frac{z_t}{p_t} \quad (3)$$

An error in CR's mathematical derivation of expected intermittent demand size was reported by Syntetos and Boylan (2001), who proposed a revision as an approximate correction of CR: the SBA method.
Several variations were applied to CR after its introduction in 1972, with SBA considered by several authors as one of the most effective for high level of intermittence.
The forecast demand per period (D_{t+1}) at the end of time period t for the SBA is given by:

$$D_{t+1} = (1 - \frac{\beta}{2}) \frac{z_t}{p_t} \quad (4)$$

Note that Eq.(3) is similar to Eq.(4), except for the presence of the corrective factor $(1 - \frac{\beta}{2})$. In fact, both z_t and p_t have the same meaning as those in Eq.(3).

4. Single-hidden layer neural networks

Three different single-hidden layer architectures have been adopted, i.e. feedforward, time-delay, and recurrent, which are set out in Sections 4.1, 4.2, and 4.3 respectively.

4.1 Feedforward neural networks

Given a set of T samples $\{(\mathbf{x_t}, \mathbf{d_t})|t = 1, \dots, T\}$, with $\mathbf{x_t} = (x_{t1}, x_{t2}, \dots, x_{tn}) \in \mathbb{R}^{\boldsymbol{n}}$ and $\mathbf{d_t} = (d_{t1}, d_{t2}, \dots, d_{tm})^{\mathrm{T}} \in \mathbb{R}^{\boldsymbol{m}}$ corresponding respectively to the input and the target vectors for the supervised training, a single-hidden layer feedforward neural network with N hidden neurons and activation function g is represented as follows:

$$G_N(\mathbf{x_t}) = \sum_{i=1}^{N} \boldsymbol{\beta_i} g(\mathbf{w_i} \cdot \mathbf{x_t} + b_i) \quad \text{with } t = 1, \dots, T \tag{5}$$

where $\mathbf{w_i} = (w_{i1}, w_{i2}, \dots, w_{in})^{\mathrm{T}}$ is the input weight vector connecting the input neurons with the i-th hidden neuron, $\boldsymbol{\beta_i} = (\beta_{i1}, \beta_{i2}, \dots, \beta_{im})^{\mathrm{T}}$ is the output weight vector connecting the hidden to the output neurons, and b_i is the bias associated with the i-th hidden neuron.
Approximating the sample with zero errors is equivalent to find $\mathbf{w_i}$, $\boldsymbol{\beta_i}$, and b_i such that:

$$\|G_N(\mathbf{x_t}) - \mathbf{d_t}\| = 0 \quad \text{with } t = 1, \dots, T \tag{6}$$

It is possible to reformulate these T equations by introducing the so called hidden layer output matrix $\mathbf{H}$ (Huang and Babri, 1998; Huang, 2003), where the i-th column contains all the outputs of the i-th hidden neuron fed by all the T samples:

$$\mathbf{H}\boldsymbol{\beta} = \mathbf{D} \tag{7}$$

where:

$$\mathbf{H} = \begin{bmatrix} g(\mathbf{w_1} \cdot \mathbf{x_1} + b_1) & \dots & g(\mathbf{w_N} \cdot \mathbf{x_1} + b_N) \\ \dots & \dots & \dots \\ g(\mathbf{w_1} \cdot \mathbf{x_T} + b_1) & \dots & g(\mathbf{w_N} \cdot \mathbf{x_T} + b_N) \end{bmatrix}_{T \times N} \tag{8}$$

$$\boldsymbol{\beta} = \begin{bmatrix} \boldsymbol{\beta_1} \\ \dots \\ \boldsymbol{\beta_N} \end{bmatrix}_{N \times m} \quad \text{and} \quad \mathbf{D} = \begin{bmatrix} \mathbf{d_1} \\ \dots \\ \mathbf{d_T} \end{bmatrix}_{T \times m} \tag{9}$$

Figure 1 shows the standard feedforward architecture.

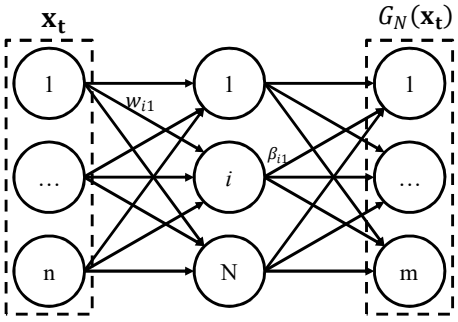

Figure 1. The feedforward neural network

4.2 Time-delay neural networks

The time-delay architecture (see Figure 2) differs from the feedforward one since each input t not only includes the feedforward network input $\mathbf{x_t}$ but also some of the preceding samples. The number of these preceding inputs is called taps. Thus the time-delay neural network input t is:

$$\hat{\mathbf{x}}_\mathbf{t} = \left(\mathbf{x_t}, \mathbf{x_{t-1}}, \ldots, \mathbf{x_{t-taps}}\right) \tag{10}$$

This changes both the dimension of each input vector, $\hat{\mathbf{x}}_\mathbf{t} \in \mathbb{R}^{n\cdot(taps+\ )}$, and of each input weight vector $\mathbf{w_i} \in \mathbb{R}^{n\cdot(taps+\ )}$. The rest of the architecture remains the same.

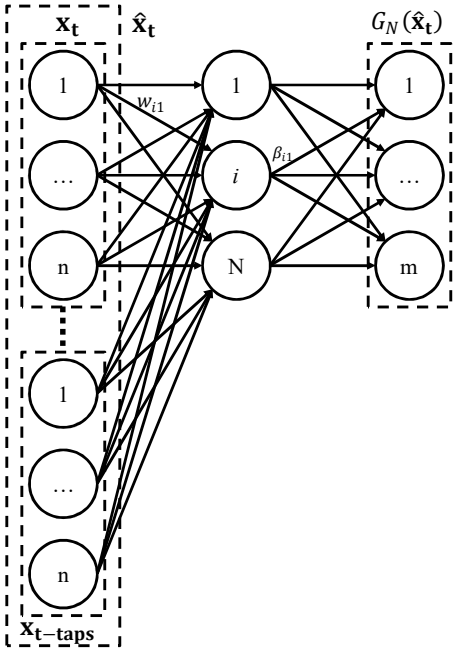

Figure 2. The time-delay neural network.

4.3 Recurrent neural networks

Recurrent architecture (see Figure 3) contains a context layer which is fully connected to the hidden layer and recurrently enriches the training stage using the previous output signals. The recurrent architecture can be modelled and used as a feedforward one by changing the network input. Each input t not only includes the sample $\mathbf{x_t}$ but also hidden layer's output, produced for the previous input, defined as:

$$\hat{\mathbf{g}}_\mathbf{t} = \left(g(\mathbf{w_1} \cdot \hat{\mathbf{x}}_\mathbf{t} + b_i), g(\mathbf{w_2} \cdot \hat{\mathbf{x}}_\mathbf{t} + b_i), \ldots, g(\mathbf{w_N} \cdot \hat{\mathbf{x}}_\mathbf{t} + b_i)\right) \tag{11}$$

Thus the time-delay neural network input for the period t is:

$$\hat{\mathbf{x}}_\mathbf{t} = (\mathbf{x_t}, \hat{\mathbf{g}}_\mathbf{t-1}) \tag{12}$$

This architecture was proposed by Elman and Zipser (1988) to provide dynamic memory to the network. It was also tested by Nasiri Pour et al. (2008), with promising results for forecasting intermittent demand patterns. If back-propagation is adopted for training the network, a further parameter named *time constant* has to be set, with a value of less than one; it allows back-propagation of the previous output signals with a contribution that decays slowly toward zero during each cycle. Hence, the context layer contains its own embedded 'memory' mechanism. In the extreme learning machines training there is no *time constant*. Each input weight vector $\boldsymbol{w_i}$ is set before the training takes place and all the output weight vectors $\boldsymbol{\beta_i}$ are affected by the $\hat{\boldsymbol{x}}_\boldsymbol{t}$ time-delay neural network inputs only.

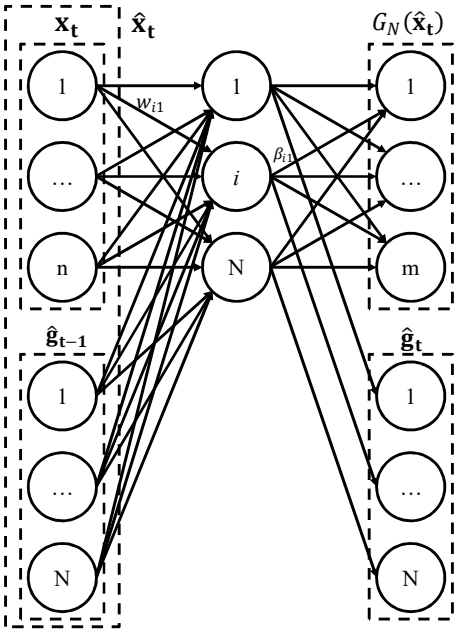

Figure 3. The recurrent neural network.

5. Learning approaches

Training the network aims at finding $\mathbf{w}_i^*$, $\boldsymbol{\beta}_i^*$, and b_i^* for each i-th hidden neuron, minimising the error $\|G_N(\mathbf{x_t}) - \mathbf{d_t}\|$ over the samples. From Eq.(7):

$$\|\mathbf{H}(\mathbf{w}_1^*, \dots, \mathbf{w}_N^*, \boldsymbol{\beta}_1^*, \dots, \boldsymbol{\beta}_N^*, b_1^*, \dots, b_N^*)\boldsymbol{\beta}^* - \mathbf{D}\| = \min_{\mathbf{w_i}, \boldsymbol{\beta_i}, b_i} \|\mathbf{H}(\mathbf{w_1}, \dots, \mathbf{w_N}, \boldsymbol{\beta_1}, \dots, \boldsymbol{\beta_N}, b_1, \dots, b_N)\boldsymbol{\beta} - \mathbf{D}\| \quad (13)$$

From Eq.(5), this is equivalent to minimising the total error E made by the network in approximating the T samples:

$$E = \sum_{t=1}^{T} \left(\sum_{i=1}^{N} \boldsymbol{\beta_i} g(\mathbf{w_i} \cdot \mathbf{x_t} + b_i) - \mathbf{d_t} \right)^2 \quad (14)$$

Eq.(14) represents a least squares problem, where a batch quadratic error has to be minimised, and can be solved through different algorithms. As introduced in Section 1, two learning algorithms are compared here. The first one (Section 5.1) is the well-known back-propagation algorithm (Rumelhart et al., 1986) and belongs to the family of gradient-based algorithms. The latter is more recent (Huang et al., 2004; Huang et al., 2006) and is called extreme learning machines (Section 5.2).

5.1 Back-propagation

This is a first order gradient method, where a backward pass starts by computing the error gradient on $G_N(\mathbf{x_t})$ (Eq.(5)), and then propagates derivatives from the output to the input layer by using the chain rule in order to assess the error gradient with respect to $\mathbf{W} = (\mathbf{w_1}, \dots, \mathbf{w_N}, \boldsymbol{\beta_1}, \dots, \boldsymbol{\beta_N}, b_1, \dots, b_N)$. The error surface is thus explored by following the opposite of the error gradient in order to establish a global minimum; this constitutes the training stage over a predefined number of iterations named epochs.
Two different learning modes can be used during the training stage. In the first type, called batch learning, the whole sample set is presented to the network in each epoch e. Hence, the error must be minimised across all the training samples in the opposite direction of the resulting gradient. Eq.(14) represents the batch error across the samples, whose gradient must be evaluated. In the second one, called online or example-by-

example learning, $\mathbf{W}$ is changed after the presentation of each sample from 1 to T. That is to say, the error gradient is first calculated using the first sample, and then the second one, and so on until T. This can potentially lead to instability whenever they change offset due to the randomized position of the subsequent training samples in the error space, and computation effort may be wasted on irrelevant data. Conversely, online learning can be helpful to escape from local minima. Online learning is the only type that can be applied to real-time problems, i.e. when data arrive in real time.
The standard gradient descent formula adopts the learning rate η, indicating how far the components of $\mathbf{W}$ change on the error surface over the epochs, in terms of the percentage of gradient arrow length. Given a generic epoch e and the expression of the back-propagated error E, the vector $\mathbf{W}^{e}$ referring to the epoch e is given by:

$$\mathbf{W}^{e} = \mathbf{W}^{e-1} - \eta \frac{\partial E(\mathbf{W})}{\partial \mathbf{W}} \tag{15}$$

For example, a learning rate equal to one causes the movements on the error surface to be exactly equal to the gradient. A learning rate value which is too high may cause entrapment of the learning in a local minimum, due to the oscillation phenomenon. Conversely, a low value leads to a slow learning stage.
A common way to improve the gradient descent is to combine the gradient descent (Eq.(15)) with a momentum μ (reader can refer to Rumelhart et al., 1986; Qian 1999). This factor determines the inertia of the weight change through the application of exponential smoothing, with the aim of establishing the relative importance of past weight changes for the current one, analogous to a common smoothing parameter.
The learning rate and momentum are therefore applied to the weight change formula for the current epoch e as follows:

$$\Delta\mathbf{W}^{e} = \mu\Delta\mathbf{W}^{e-1} - \eta \frac{\partial E(\mathbf{W})}{\partial \mathbf{W}} \tag{16}$$

With:

$$\Delta\mathbf{W}^{e} = \mathbf{W}^{e} - \mathbf{W}^{e-1} \tag{17}$$

Note that a null momentum means the past weight change will not be considered.
Several works deal with the choice of η and μ, providing useful criteria for selecting η and μ (e.g. $\eta + \mu = 1$) but do not guarantee their optimality, regardless of the specific problem type faced by the ANNs. Gutierrez et al. (2008) adopt $\eta = 0.1$ and $\mu = 0.9$, as well as Mukhopadhyay et al. (2012), while Nasiri Pour et al. (2012) do not state these values. Kourentzes (2013) applies the Levenberg-Marquardt algorithm instead of the gradient descent (for a description of this algorithm, see Fun and Hagan, 1996).
ANNs trained by means of the gradient descent of the back-propagated error suffer from the weaknesses already emphasized in Section 1. Among these, they are susceptible to the overfitting like all the machine learning tools, which compromises their generalizing ability through reduced adaptability to new data. There may be several potential causes of the overfitting phenomenon to be investigated (e.g. high number of neurons in the hidden layers, high number of setting parameters, limited training data, etc...), including the excessive length of the training stage in terms of the number of epochs. Hence a wide set of stopping criteria have been introduced in the literature, with the aim of avoiding the overfitting phenomenon, as well as the subsequent worsening of predictive performance for new data. Fixing the number of learning epochs, choosing a threshold value for the error during the training, applying cross-validation and early stopping are some of the possible strategies to avoid overfitting. Readers can refer to Prechelt (1998), Nguyen et al. (2005), Chan et al. (2006) and Piotrowski and Napiorkowski (2013) for a comprehensive review of the overfitting phenomenon, along with the strategies for avoiding it. With regard to the ANNs used to forecast intermittent demand, Gutierrez et al. (2008) state that the training should stop whenever the error is

minimized, as do Mukhopadhyay et al. (2012). However, it is worth emphasising that the best stopping criterion cannot be generalized, regardless of the ANN or the data being handled. This is supported by the empirical evidence reported in the experimental section.

5.2 Extreme learning machines

Two theorems constitute the theoretical foundation of extreme learning machines, whose proofs are reported in Huang et al. (2006).

The first states that a for a single-hidden layer feedforward neural network where $N = T$, that is to say with a number of hidden neurons equal to the number of samples, and an activation function g that is infinitely differentiable in any interval, the hidden layer output matrix $\mathbf{H}$ (see Eq.(8)) is invertible, and $\|\mathbf{H\beta} - \mathbf{D}\| = 0$ for any $\mathbf{w_i}$ and b_i randomly chosen according to any continuous probability distribution. It follows that a single-hidden layer feedforward neural network with these properties satisfies the universal approximation capability (Huang et al., 2015), and can approximate the T samples with zero error. Moreover, given the random initialisation of $\mathbf{w_i}$ and b_i, the only free parameters for learning are thus the weights between the hidden layer and the output layer, which are the components of the vector $\boldsymbol{\beta}$ (see Eq.(9)). The implementation of extreme learning machines is therefore greatly simplified because the iterative tuning of the hidden neurons is no longer required.

Given an activation function g infinitely differentiable in any interval and for any $\mathbf{w_i}$ and b_i randomly chosen according to any continuous probability distribution, and according to the first theorem, the second one states that there always exists a number of hidden neurons $N \leq T$ such that $\|\mathbf{H\beta} - \mathbf{D}\| < \varepsilon$, for any small positive value $\varepsilon > 0$. The upper bound of the hidden nodes is therefore T. If learning error is allowed, it follows that the only parameter to set is the number of hidden neurons. This implies the overfitting risk for extreme learning machines as well, but only due to the number of hidden neurons. Conversely, the gradient method may lead to overfitting, even due to an excessive tuning of the hidden neurons during the training stage.

Since $N \ll T$ in many real applications to forecasting, which means that the errors are allowed in the training stage, the random assignment of $\mathbf{w_i}$ and b_i makes $\mathbf{H\beta} = \mathbf{D}$ a linear system, and Eq.(13) is equivalent to finding $\boldsymbol{\beta}^*$ as the least-squares solution of:

$$\|\mathbf{H}(\mathbf{w_1}, \dots, \mathbf{w_N}, \mathrm{b}_1, \dots, \mathrm{b}_\mathrm{N})\boldsymbol{\beta}^* - \mathbf{D}\| = \min_{\beta_i} \|\mathbf{H}(\mathbf{w_1}, \dots, \mathbf{w_N}, \mathrm{b}_1, \dots, \mathrm{b}_\mathrm{N})\boldsymbol{\beta} - \mathbf{D}\| \tag{18}$$

Actually, if $N < T$ then $\mathbf{H}$ is a non-square matrix. A well-known algebraic theorem (readers can refer to Serre, 2002) states that the smallest norm solution of Eq.(18) is:

$$\boldsymbol{\beta}^* = \mathbf{H}^\dagger \mathbf{D} \tag{19}$$

where $\mathbf{H}^\dagger$ is the *Moore-Penrose generalised inverse* of $\mathbf{H}$ (Penrose and Todd, 1955)**.** Singular value decomposition is one of several methods used to calculate $\mathbf{H}^\dagger$, e.g. orthogonalization and iterative. Huang et al. (2006) underlined that singular value decomposition can always be used for the $\mathbf{H}^\dagger$ calculation in an extreme learning machines scenario. Said calculation can take place even if the matrix $\mathbf{H}$ contains linearly dependent columns.

6. The experienced networks

By using the notation adopted previously, a general forecasting problem dealing with neural networks consists of assessing the input pattern $\mathbf{x_t} = (x_{t1}, x_{t2}, \dots, x_{tn}) \in \mathbb{R}^{\boldsymbol{n}}$**,** with sample $t = 1, \dots, T$, to correlate with the corresponding T mono-dimensional target $d_t \in \mathbb{R}$**,** i.e. the T past demand values.

In this work, a three-dimensional input $\mathbf{x_t}$, with $t = 1, \dots, T$, is represented by the pattern $\mathbf{x_t} = (d_{t-1}, \tau_t, \delta_t)$, where:

- d_{t-1}: the demand at time period $t - 1$;
- τ_{t-1}: "the number of periods separating the last two non-zero demand transactions at the end of the immediately preceding period" (Gutierrez et al., 2008);
- δ_{t-1}: "the cumulative number of successive periods with zero demand" (Mukhopadhyay et al., 2012).

The second and the third input neurons were respectively introduced by Gutierrez et al. (2008) and Mukhopadhyay et al. (2012) into their two-dimensional input patterns $\mathbf{x_t} = (d_{t-1}, \tau_{t-1})$ and $\mathbf{x_t} = (d_{t-1}, \delta_{t-1})$, but they have never been used simultaneously. However, data from both of these input neurons could jointly improve the training on the temporal behaviour of the time series in terms of zero/non-zero demand. In fact, this is the main issue to be addressed with regard to intermittent demand patterns. Since their interpretation can be misunderstood, Table 1 clarifies the input pattern with a numeric example.

d_{t-1}	τ_{t-1}	δ_{t-1}
1	0	0
5	0	0
7	0	0
0	0	1
0	0	2
12	2	0
1	0	0

Table 1. An example of three-dimensional input patterns.

This input pattern is adopted into the three architectures, i.e. feedforward, time-delay, and recurrent, respectively set out in Sections 3.1, 3.2, and 3.3. The resulting networks are trained by the back-propagation gradient-descent algorithm, both in online and batch modalities, and by extreme learning machines. The extreme learning machines training is performed only in the batch mode.

6.1 Notation

In order to uniquely identify a network, a compact notation consisting of a string of fields separated by "_" is introduced, which refer to the architecture, the learning approach, and the learning mode. These fields are indicated with the acronyms below, with their meanings written in brackets:

- Architecture: FF (*Feedforward neural network*); TD (*Time-delay neural network*); R (*Recurrent neural network*).
- Learning approach: BP (*Back-propagation*); E (*Extreme learning machines*).
- Learning mode: B (*Batch*); O (*Online*).

For instance, a network TD_BP_O has a time-delay architecture and is trained using back-propagation in the online modality.
Since the experimental comparison involves the networks of Gutierrez et al. (2008) and Mukhopadhyay et al. (2012) as benchmarks, they are respectively indicated as "GUT" and "MUK", assuming that these networks are feedforward with two input neurons. However, GUT and MUK have been trained by back-propagation as in the original contributions, both in batch and in online modalities (GUT_BP_B, GUT_BP_O, MUK_BP_B, and MUK_BP_O) and by extreme learning machines (GUT_E and MUK_E).
CR and SBA differ only in terms of the smoothing coefficients, which are fixed at the same value (i.e. $\alpha = \beta$), and are followed by the value of the smoothing coefficient, e.g. CR_0.2.

7. The comparison approach

The time series were first divided into a training and a test set. The former contains the first T samples on which the networks are trained, while the latter is composed of the remaining S observations on which the comparison is performed. Hence, the comparison involved the demands ($d_{T+1}, d_{T+2}, \dots, d_{T+S}$) and the forecasts ($D_{T+1}, D_{T+2}, \dots, D_{T+S}$). The partition of the time series into a training and a subsequent test set cannot be driven by any general rule, due to the sensitivity of ANN performance to a multitude of correlated features affecting their generalizing ability (e.g. network architecture, time series length, training parameters, etc.). With regard to the ANNs compared in this work, Gutierrez et al. (2008) adopted a partition of 65-35 % of the time series for the training and the test set respectively, while Mukhopadhyay et al. (2012) tested three different partitions, i.e. 80-20, 65-35 and 50-50 %. However, in order to make the results as comparable as possible, the 65-35 % partition was adopted for all the approaches.
The Croston method-CR (Croston, 1972) and the Syntetos-Boylan approximation-SBA (Syntetos and Boylan, 2005) have also been taken as benchmarks. Given a predefined set of values for the smoothing coefficients, they were optimised by minimising the mean square error (MSE) on the training set for each time series. The choice of using such an accuracy measure for setting the smoothing coefficients is analogous to the neural networks learning approach (see Eq.(7)).
The results reported by Makridakis and Hibon (2000) referring to the well-known M3-Competition, as well as by others focusing specifically on forecasting intermittent demand (e.g. Teunter and Duncan, 2009; Teunter and Sani, 2009) suggest using different accuracy measures for the comparison, because a single measure could not be entirely informative on the different dimensions of the error. Readers can refer to Wallström and Segerstedt (2010) for a review of different accuracy measures for intermittent demand. In general, two categories of measures may be identified, which are respectively scale and non-scale dependent. Since it is intended here to propose statistical tests on these measures from a set of time series, only non-scale dependent measures must be used. In particular, two measures were taken into account.
The first is the common $MAPE$ (Mean Absolute Percentage Error), which is expressed as follows:

$$MAPE = \frac{\sum_{t=T+1}^{T+S} |d_t - D_t|}{\sum_{t=T+1}^{T+S} d_t} \tag{20}$$

This measure provides the relationship between the average absolute error and the average demand on a certain time horizon, so that it can be considered a valid choice for time series with different mean levels. In fact, $MAPE$ is given by the ratio of the metrics MAD (Mean Absolute Deviation) and A (Average Demand):

$$MAD = \frac{\sum_{t=T+1}^{T+S} |d_t - D_t|}{S} \tag{21}$$

$$A = \frac{\sum_{t=T+1}^{T+S} d_t}{S} \tag{22}$$

$MAPE$ can also be applied to intermittent demand with at least one strictly positive observation during the time horizon, because d_t does not appear in any denominator.
The second accuracy measure, i.e. ME/A, refers to the non-scale dependent systematic error (bias) committed, and thus allows us to determine whether a forecasting approach on average underestimates or overestimates the level of demand. This is expressed as the ratio between ME (Mean Error) and A (Eq.22), where ME is as follows:

$$ME = \frac{\sum_{t=T+1}^{T+S} (d_t - D_t)}{S} \tag{23}$$

Hence:

$$ME/A = \frac{\sum_{t=T+1}^{T+S}(d_t - D_t)}{\sum_{t=T+1}^{T+S} d_t} \quad (24)$$

Actually, a positive or a negative ME/A means respectively that the method underestimates or overestimates the level of demand.

A further well-known guideline reported in Makridakis and Hibon (2000) suggests evaluating the performance of forecasting methods for different time horizons. Here, the accuracy of the forecasting method was evaluated for one, three and five periods ahead. In the case of three and five periods ahead in particular, both periodic and rolling updating were tested. Periodic updating grouped the single periods into aggregated periods. The size of each aggregated period was determined by the time horizon considered (three or five) and a single period could belong to an aggregated period only. Each accuracy evaluation was performed knowing the last aggregated period outcome divided by the number of single periods used to produce it. The rolling updating, on the contrary, worked with a moving window pattern. Each single period could belong to more aggregate periods and the central point of the aggregate period under consideration would move ahead one single period at a time. The distinction between periodic and rolling updates only makes sense for time horizons greater than one time period. Moreover, the number of accuracy evaluations applied for rolling updating is higher than for periodic updating. For instance, a test set of twelve time periods with a forecasting horizon of three periods allows four periodic and ten rolling accuracy evaluations.

To sum up, following the notation reported in Section 6.1, three new ANNs , i.e. FF, TD, and R, and two existing ANNs, i.e. GUT (Gutierrez et al., 2008) and MUK (Mukhopadhyay et al., 2012), have been trained by BP (Rumelhart et al., 1986), in modalities O and B, and ELM (Huang et al., 2006), and compared with optimised CR (Croston, 1972) and SBA (Syntetos and Boylan, 2005) for three time horizons (one, three and five periods ahead), through periodic and rolling updates, in terms of $MAPE$ and ME/A.

8. Experimental analysis

After data collection with the relevant descriptive statistics (Section 8.1), the parameter settings are provided. (Section 8.2).

8.1. Data collection

Twenty-four weekly intermittent time series were collected from the spare-part dataset of an industry operating in the automotive sector. In order to highlight the erraticness and intermittence levels, CV and ADI were computed using the definitions provided in Willemain et al. (1994). Specifically, CV represents the coefficient of variation of non-zero demands, while ADI represents the average number of time periods between two successive non-zero demands. Alternatively, using the definitions given in Syntetos and Boylan (2001), CV^2, i.e. the squared version of CV, can be computed. CV and ADI thus represent the measures of demand size erraticness and intermittency respectively (see rows ADI and CV, or rows ADI and CV^2, in Table 2). Moreover, CV and ADI are not correlated here, and the time series do not show any seasonality or trend. CV^2 and ADI have been used by Syntetos et al. (2005) for categorising demand patterns on the basis of their theoretical thresholds (0.49 and 1.32, respectively), which have been derived as the values identifying the regions where SBA outperforms CR in terms of the theoretical mean square error. Figure 4 plots the twenty-four time series in a CV^2-ADI diagram, where the number of items and the categories of demand patterns are reported into each quadrant.

Moreover, the mean values of the time series are reported in Table 2 in the 'mean' rows, along with the length of the time series.

Series	1	2	3	4	5	6	7	8
Length	75	76	69	73	76	75	61	65
ADI	1.19	1.23	1.61	1.35	1.38	1.34	1.3	1.3
CV	0.71	0.57	0.44	0.57	0.62	0.96	0.63	0.65
CV^2	0.5	0.32	0.19	0.32	0.38	0.92	0.4	0.42
Mean	22.23	3.07	0.87	3.89	2.61	12.01	13.34	4.23
Series	9	10	11	12	13	14	15	16
Length	84	73	84	74	72	74	83	82
ADI	1.91	1.92	1.87	1.85	1.5	2.85	1.38	1.41
CV	0.65	0.44	0.68	0.57	0.52	0.33	0.52	0.74
CV^2	0.42	0.19	0.46	0.32	0.27	0.11	0.27	0.55
Mean	4.92	0.9	1.62	2.85	28.44	0.42	2.18	124.22
Series	17	18	19	20	21	22	23	24
Length	76	74	68	414	83	81	83	81
ADI	1.52	1.18	3.78	1.87	1.36	1.35	1.22	1.27
CV	0.70	0.83	0.42	0.77	0.87	0.57	0.60	0.57
CV^2	0.49	0.69	0.18	0.6	0.76	0.32	0.36	0.32
Mean	6	3.74	1.24	14.02	12.55	128.68	104.25	30.98

Table 2. The time series data collected.

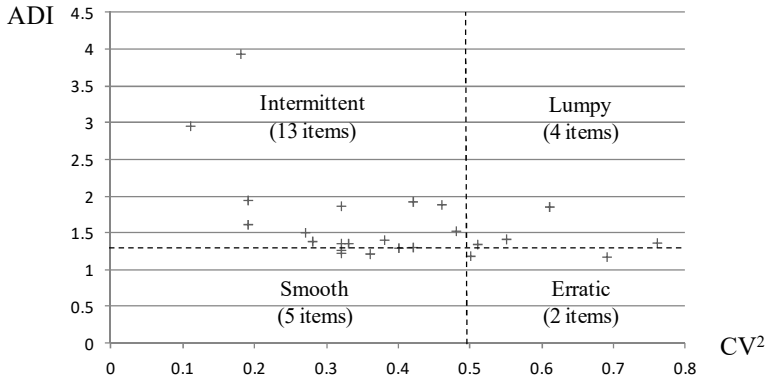

Figure 4. The CV^2-ADI categorisation for the time series.

8.2 Parameter settings

All the experimentation was performed using Joone®, NeuroSolutions®, and Matlab® for extreme learning machines. Table 3 summarises all the parameter settings.
The GUT and MUK networks were set with their original architectures and training parameters. Thus, three hidden neurons were used, with the same learning rate and momentum, i.e. $\varepsilon = 0.1$ and $\mu = 0.9$ respectively. The overfitting phenomenon has to be necessarily monitored when adopting BP as the learning approach, so the stopping criterion should be established on a case by case basis. GUT and MUK involved training the ANNs during subsequent steps of 15000 epochs, as in their original experimentations. A similar stopping criterion, i.e. subsequent steps of epochs, was also adopted for FF_BP and R_BP, but the number of epochs was set to 10000 in these cases. In fact, when the ANNs show good learning ability in the training stage due to the additional input neuron, the overfitting phenomenon can be caused by forcing the training through an excessive number of epochs. Moreover, given the option of continuously monitoring the MSE trend, the training stage was stopped as soon as the minimum error was reached without any further significant

decrease. This strategy appears, in the authors' opinion, to be the most cautionary, as well as the least time-consuming one. For TD_BP_B and TD_BP_O, their better learning ability was empirically revealed during the training stage, and thus a threshold value of MSE can be adopted. In particular, four thresholds of MSE (0.005, 0.01, 0.017 and 0.026) were tested during the training stage. The threshold $MSE = 0.017$ appeared to be the most significant since, after that level the MSE, improvements became harder to achieve. In these conditions, allowing a smaller MSE could have led to overfitting phenomena.
With regard to CR and SBA, $\alpha = \beta$ in the range 0.05-0.2 with 0.05 according to the literature in this field (e.g. Babai et al., 2014).

	α, β	Hidden Neurons	Momentum	Learning Rate	Stopping Criterion	Partition	Taps	Time Constant
CR	0.05/0.1/ 0.15/0.2	-	-	-	-	65-35	-	-
SBA	0.05/0.1/ 0.15/0.2	-	-	-	-	65-35	-	-
GUT_BP_B	-	3	0.9	0.1	Steps of 15000 epochs	65-35	-	-
GUT_BP_O	-	3	0.9	0.1	Steps of 15000 epochs	65-35	-	-
GUT_E	-	3	-	-	-	65-35	-	-
MUK_BP_B	-	3	0.9	0.1	Steps of 15000 epochs	65-35	-	-
MUK_BP_O	-	3	0.9	0.1	Steps of 15000 epochs	65-35	-	-
MUK_E	-	3	-	-	-	65-35	-	-
FF_BP_B	-	3	0.9	0.1	Steps of 10000 epochs	65-35	-	-
FF_BP_O	-	3	0.9	0.1	Steps of 10000 epochs	65-35	-	-
FF_E	-	3	-	-	-	65-35	-	-
TD_BP_B	-	3	0.9	0.1	Threshold value of MSE	65-35	3	-
TD_BP_O	-	3	0.9	0.1	Threshold value of MSE	65-35	3	-
TD_E	-	3	-	-	-	65-35	3	-
R_BP_B	-	3	0.9	0.1	Steps of 10000 epochs	65-35	-	0.8
R_BP_O	-	3	0.9	0.1	Steps of 10000 epochs	65-35	-	0.8
R_E	-	3	-	-	-	65-35	-	-

Table 3. The parameter settings.

9. Statistical analysis

Two different statistical analyses have been performed. The first one (Section 9.1) refers to the comparison between the methods for different aggregation levels. The second one (Section 9.2) aims at investigating the relationship between the features of the time series (ADI and CV) and the accuracy of the forecasting methods under comparison. This deep statistical analysis allowed us to develop some useful suggestions for practitioners (Section 9.3).

9.1. Comparison between methods

In this section, a statistical test is presented for each accuracy measure ($MAPE$ and ME/A) obtained in the test set. Each test is repeated for two aggregation levels, and the latter of these contains two comparisons. The first aggregation level refers to an overhaul analysis, that is the best performance obtained by the CR and SBA methods (CR_0.05, …, CR_0.2, SBA_0.05, …, SBA_0.2) is compared with the best one obtained using the back-propagation learning approach (GUT_BP_B, GUT_BP_O, MUK_BP_B,..., R_BP_O) and the best one achieved with the extreme learning approach (GUT_E, MUK_E,..., R_E). Following the second aggregation level, the first comparison is an in-depth analysis of the back-propagation methods; this means that the best performing feedforward (FF_BP_B and FF_BP_O), time-delay (TD_BP_B and TD_BP_O), recurrent (R_BP_B and R_BP_O), GUT (GUT_BP_B, GUT_BP_O) and MUK (MUK_BP_B, MUK_BP_O) networks are compared with each other. The second comparison is among the extreme learning machines, and is obtained by comparing FF_E, TD_E, R_E, GUT_E and MUK_E with each other. These six tests are performed for each time horizon (one, three and five periods ahead) and update methodology (periodic and rolling updates). This leads to a total of thirty tests. Each test follows the same pattern, which will be detailed in this section. The significant results will be presented in a table and then analysed.

The first objective of the presented tests is to prove whether the average accuracy measure, obtained for one method, is significantly different from the others. Since each test compares more than two methodologies, a t-test cannot be used to address this problem. Instead an ANOVA test should be performed. Nevertheless, the methodology differs slightly in this case. In fact each method, i.e. treatment, is measured based on the same set of items, while the general ANOVA procedure requires each item to be evaluated using one treatment only. This obstacle can be overcome using a variant of the test called Repeated Measures ANOVA (rANOVA). The variance of the subjects analysed, calculated as the sum of squared distances between each subject performance mean and the grand mean, is subtracted from the variance between. As the measures are repeated, this sample–dependent variance can be measured and accounted for in this way, thereby increasing the effectiveness of the test. Moreover, the standard ANOVA procedure assumes that the variance does not change among the treatments. If this assumption holds and the test rejects the null hypothesis, then the rejection must be caused by a difference among the performance means. The rANOVA relies on a similar assumption, which is the sphericity assumption. The difference variance of a pair of treatments is calculated as the sum of the squared difference of each item performance, divided by its degrees of freedom. The Mauchly's Test is performed on these data to accept or reject ($p<=0.05$) the null hypothesis that these difference variances are non-significantly different. In the examples presented, the sphericity hypothesis is often violated. This would increase the chances of incurring type I errors, and thus rejecting more correct null hypotheses than expected. In these cases, the Greenhouse-Geisser correction factor is applied to the degrees of freedom used during the test ratio calculation.

Rejecting the null hypothesis means that the effectiveness for each method is significantly different. In this case, a different test is required to address the second objective of the analysis, that is to find which methods are different. For each pair of treatments, the Tukey's range test is performed as a mean values comparison. This particular test relies on a studentized range distribution and is able to manage the family-wise error rates arising in multiple-comparison scenarios. To be considered significant, a comparison must reject ($p<=0.05$) the null hypothesis, stating that the mean values considered are not different.

Table 4 contains the $MAPE$ analysis results.

Time horizon	Aggregation level	rANOVA p-value	Better performer	Worse performer	Tukey's p-value
One period ahead	Overhaul analysis	0.008	BP $MAPE$: 0.76	E $MAPE$: 0.79	0.044
			E $MAPE$: 0.79	CR-SBA $MAPE$: 0.81	0.045
			BP $MAPE$: 0.76	CR-SBA $MAPE$: 0.81	0.024

	Back-propagation comparison	0.37			
	Extreme learning comparison	0.37			
Three periods ahead, periodic	Overhaul analysis	0	BP $MAPE$: 0.46	CR-SBA $MAPE$: 0.51	0.002
			E $MAPE$: 0.48	CR-SBA $MAPE$: 0.51	0.037
	Back-propagation comparison	0.022	Non-identifiable		
	Extreme learning comparison	0.29			
Three periods ahead, rolling	Overhaul analysis	0.003	BP $MAPE$: 0.47	CR-SBA $MAPE$: 0.51	0.012
	Back-propagation comparison	0.014	Non-identifiable		
	Extreme learning comparison	0.09			
Five periods ahead, periodic	Overhaul analysis	0.026	BP $MAPE$: 0.37	CR-SBA $MAPE$: 0.42	0.047
	Back-propagation comparison	0.029	Non-identifiable		
	Extreme learning comparison	0.451			
Five periods ahead, rolling	Overhaul analysis	0.052			
	Back-propagation comparison	0.003	MUK $MAPE$: 0.40	TD $MAPE$: 0.51	0.036
	Extreme learning comparison	0.279			

Table 4. Statistical tests on $MAPE$ for different aggregation levels.

On the overhaul analysis level and regardless of the time horizon and update methodology, the data show a meaningful performance gap between traditional methodologies and the neural networks trained with the back-propagation algorithm. This behaviour is not followed by the same networks trained with the extreme learning, which present a smaller performance gap compared to traditional methods. This gap is significant only in the first analysis, while an increase of the time horizon means it is not meaningful, as in the last three examples. This performance difference reduction is also observable in the behaviour of networks trained by back-propagation, compared with those trained by extreme learning. In the first comparison, the difference between these training methodologies is significant, in favour of the back-propagation algorithm. This distinction is lost when the time horizon is increased. From three periods ahead, only the difference between the back-propagation and CR-SBA methods is significant. The rolling update methodology acts in the same way, compared to the periodic one. In the three-periods scenario, the gap between BP and CR-SBA drops from 0.05 to 0.04, changing the update methodology. The same happens in the five-periods case, where in the rolling case, for the first time, the overhaul analysis becomes non-significant. On a lower aggregation level, the networks trained with the extreme learning do not present meaningful performance differences. The same networks trained with the back-propagation algorithm often present significant performance gaps, but unfortunately the second level test is often unable to identify with which methods the performances differ. A possible reason for this issue is provided by the last comparison, in which MUK outperforms FF. Table 5 contains the ME/A analysis results.

Time horizon	Aggregation level	rANOVA p-value	Better performer	Worse performer	Tukey's p-value
One period ahead	Overhaul analysis	0.676			
		0.018	MUK	FF	0.057

	Back-propagation comparison		ME/A: 0	ME/A: -0.1	
			MUK ME/A:	R ME/A: -0.16	0.055
	Extreme learning comparison	0.48			
Three periods ahead, periodic	Overhaul analysis	0.536			
	Back-propagation comparison	0.022	MUK ME/A: -0.03	FF ME/A: -0.14	0.045
			MUK ME/A: -0.03	R ME/A: -0.22	0.054
	Extreme learning comparison	0.466			
Three periods ahead, rolling	Overhaul analysis	0.924			
	Back-propagation comparison	0.014	MUK ME/A: -0.01	FF ME/A: -0.12	0.042
			MUK ME/A: -0.01	R ME/A: -0.19	0.046
	Extreme learning comparison	0.28			
Five periods ahead, periodic	Overhaul analysis	0.507			
	Back-propagation comparison	0.028	MUK ME/A: -0.02	FF ME/A: -0.11	0.021
	Extreme learning comparison	0.346			
Five periods ahead, rolling	Overhaul analysis	0.733			
	Back-propagation comparison	0.016	MUK ME/A: -0.03	FF ME/A: -0.13	0.022
			MUK ME/A: -0.03	R ME/A: -0.22	0.053
	Extreme learning comparison	0.256			

Table 5. Statistical tests on ME/A for different aggregation levels.

Both the overhaul analysis and the extreme learning comparison are strongly non-significant. Using a different learning approach does not provide different performances in terms of the relative systematic error. In the same way, if the neural network is trained with the extreme learning methodology, the architecture choice does not provide meaningful performance improvements. The only difference is among the networks trained with back-propagation. This difference is always highlighted in the rANOVA results, but it is not always easy to identify the gap source by applying the Tukey's test. Some rejected Tukey's tests show p-values very close to 0.05 with a meaningful pattern, and thus have been shown. The most robust result in the back-propagation analysis is the performance difference between the MUK and the FF, the former outperforming the latter. This finding is present across all the time horizons and update methodologies, being non-significant only in the one period horizon. The least robust result in the back-propagation analysis is the effectiveness gap between the MUK and R methods, the former probably being more effective. This relation is more doubtful since it is proved only in the three-period rolling scenario. All the others cases show p-values only close to the significant amount 0.05, while in the five-period-ahead time horizon with periodic update methodology, this last result is absent, as the p-value found (0.068) is quite far from the objective value.

From table 4, it appears that the $MAPE$ achieved in the overhaul analysis decreases while the time horizon increases. For example, the BP performance moves from 0.76 in the one-period-ahead horizon to 0.46 in the three-period-ahead periodic scenario and 0.47 in the rolling one. The reduction is also significant in the five-period-ahead periodic horizon, with a $MAPE$ of 0.37. The corresponding rolling case reaches a performance

of 0.40. The first test is nevertheless unable to prove this phenomenon, as each p-value is calculated within a time horizon. A third test has therefore been carried out, comparing the best $MAPE$ performance obtained by each time horizon and update methodology. A fourth specular test has also been run, comparing the ME/A performance across the same time horizons and update methodologies. Both these tests contain a rANOVA analysis and a subsequent Tukey's test, performed if the first analysis provides significant results.
Table 6 contains the $MAPE$ analysis results.

Comparison	r-ANOVA p-value	Better performer	Worse performer	Tukey's p-value
Between time horizons	0	Five periods ahead, rolling $MAPE$: 0.35	Three periods ahead, periodic $MAPE$: 0.45	0.001
		Five periods ahead, rolling $MAPE$: 0.35	Three periods ahead, rolling $MAPE$: 0.46	0
		Five periods ahead, rolling $MAPE$: 0.35	One period ahead $MAPE$: 0.77	0
		Five periods ahead, periodic $MAPE$: 0.35	Three periods ahead, periodic $MAPE$: 0.45	0.001
		Five periods ahead, periodic $MAPE$: 0.35	Three periods ahead, rolling $MAPE$: 0.46	0
		Five periods ahead, periodic $MAPE$: 0.35	One period ahead $MAPE$: 0.77	0
		Three periods ahead, periodic $MAPE$: 0.45	One period ahead $MAPE$: 0.77	0
		Three periods ahead, rolling $MAPE$: 0.46	One period ahead $MAPE$: 0.77	0

Table 6. Statistical tests on $MAPE$ through different time horizons.

The results shown prove the inverse link between time horizons and $MAPE$ performances. The rANOVA analysis is strongly significant and so are the singular Tukey's pairwise tests between different time horizons. Conversely, a strong lack of significance has been found when comparing different update methodologies within the same time horizon. The comparison between five periods ahead periodic and rolling achieves a p-value close to 1. The corresponding match between three periods ahead periodic and rolling achieves a p-value of 0.88. This lack of significance further confirms the link between time horizon and $MAPE$ performances, unaffected by the selected update methodologies.
Table 7 contains the ME/A analysis results.

Comparison	r-ANOVA p-value	Better performer	Worse performer	Tukey's p-value
Between time horizons	0.431			

Table 7. Statistical tests on ME/A through different time horizons.

This last test shows no connection between time horizons and ME/A performances. Since the rANOVA p-value shows a strong lack of significance, there is no need to perform a Tukey's test.

9.2. Regression analysis

As described in Section 8.1, the data used in this experimental analysis are characterized by the ADI and CV^2 parameters, presented in Table 2. It may be significant to compare the performances achieved in each scenario with the characteristics of the series achieving said performances in terms of $MAPE$ and ME/A. When the performance measure focused on is the $MAPE$, a direct relation can be seen by plotting it against the ADI characteristic. Conversely, this behaviour is absent when the $MAPE$ is plotted against CV^2. The second performance measure, ME/A, shows no significant patterns when plotted against ADI or CV^2. The $MAPE$ behaviour is believed to be linear, and this can be proved by calculating the simple regression line. After the line parameters have been calculated, an ANOVA test can be performed, comparing the variance unexplained by the regression with the total variance. The underlying null hypothesis states that the linear relationship does not explain the $MAPE$ behaviour any better than the average $MAPE$ value across the series. The null hypothesis is discarded if the p-value achieved is equal to or lower than 0.05. In this case, the linear model explains the performance behaviour significantly better than the average $MAPE$ value. Regression lines are calculated on three levels of analysis. The first level is the time horizon and the update methodology. Five lines are generated using the best result available in that time horizon by adopting that update methodology for each series. Each line is treated separately, meaning that five different ANOVA tests are performed in order to identify which structures are meaningfully linear. The second level of analysis draws a line for the back-propagation and the extreme learning machines. These lines are calculated within the time horizon frame, leading to a total of fifteen lines. As in the previous analysis, each line is managed independently. This level corresponds to the overhaul analysis performed in section 9.1. The third analysis level operates on non-grouped neural network data. Each neural network method is evaluated, with its time frame, learning algorithm and structure. This leads to fifty lines and corresponding tests. In the table below, the results for this last scenarios are provided in an aggregate form, presenting only the minimum and maximum values registered.
Table 8 contains the regression line parameters and the ANOVA test p-values.

Level of analysis	Time horizon	Learning methodology	Single method	Intercept	Slope	ANOVA p-value
Time horizons	One period ahead			-0.014	0.518	0.000
	Three periods ahead, periodic			-0.034	0.293	0.005
	Three periods ahead, rolling			-0.063	0.326	0.000
	Five periods ahead, periodic			-0.043	0.247	0.030
	Five periods ahead, rolling			-0.088	0.276	0.001
Overhaul analysis	One period ahead	Back-propagation		-0.026	0.530	0.000
		Extreme learning		-0.038	0.558	0.000
	Three periods ahead, periodic	Back-propagation		-0.049	0.312	0.007
		Extreme learning		-0.046	0.325	0.009
	Three periods ahead, rolling	Back-propagation		-0.101	0.359	0.000
		Extreme learning		-0.074	0.354	0.000
	Five periods ahead, periodic	Back-propagation		-0.033	0.257	0.029
		Extreme learning		-0.083	0.296	0.016
	Five periods ahead, rolling	Back-propagation		-0.124	0.309	0.001
		Extreme learning		-0.137	0.329	0.003
Single methods	One period ahead	Back-propagation	Min value	-0.124	0.432	0.000
			Max value	0.241	0.653	0.048
		Extreme learning	Min value	-0.187	0.583	0.000

			Max value	-0.035	0.690	0.000
	Three periods ahead, periodic	Back-propagation	Min value	-0.202	0.277	0.027
			Max value	0.041	0.532	0.050
		Extreme learning	Min value	-0.133	0.295	0.006
			Max value	0.027	0.429	0.030
	Three periods ahead, rolling	Back-propagation	Min value	-0.295	0.328	0.001
			Max value	-0.008	0.598	0.010
		Extreme learning	Min value	-0.159	0.341	0.001
			Max value	-0.029	0.452	0.002
	Five periods ahead, periodic	Back-propagation	Min value	-0.261	0.271	0.017
			Max value	0.011	0.523	0.091
		Extreme learning	Min value	-0.184	0.284	0.007
			Max value	-0.036	0.399	0.038
	Five periods ahead, rolling	Back-propagation	Min value	-0.446	0.287	0.003
			Max value	-0.048	0.638	0.020
		Extreme learning	Min value	-0.288	0.332	0.001
			Max value	-0.099	0.470	0.010

Table 8. Regression line parameters for $MAPE$ and ADI with the corresponding statistical tests.

In the table above, the only non-significant regression line is the one applied to the GUT_BP_B in the five-periods-ahead time horizon (p-value: 0.091). All the other regression lines can explain the $MAPE$ behaviour significantly better than the average $MAPE$. This strongly confirms the linear model as a reliable approximation of the $MAPE$ behaviour, regardless of the methodology applied and the aggregation level considered.
Figure 5 below shows, as an example, the one-period-ahead Overhaul analysis.

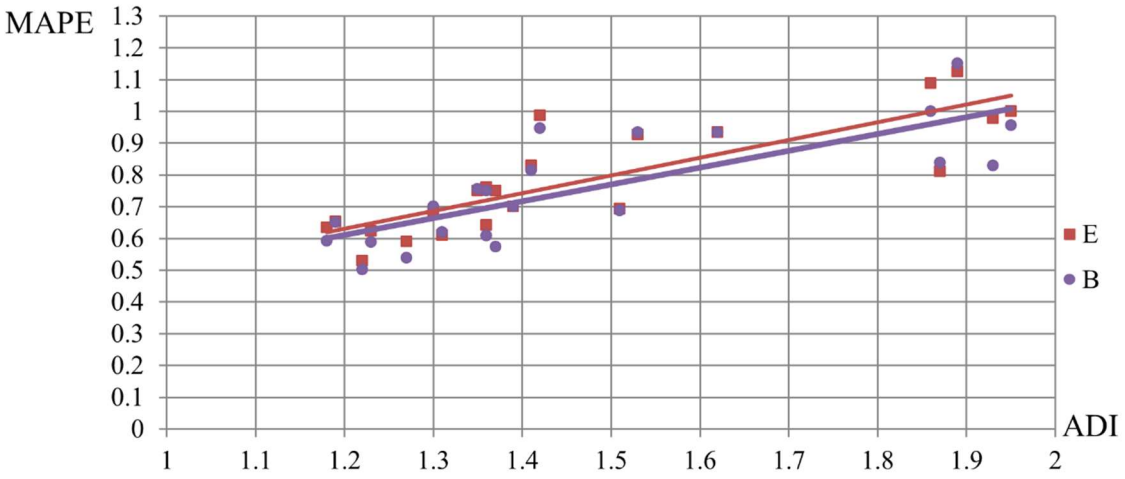

Figure 5. Regression lines for $MAPE$ and ADI, in the one-period-ahead Overhaul analysis.

9.3 Suggestions for practitioners

To sum up the deep statistical analysis performed, and derive indications that are useful for improving the applicability of neural networks to real settings, the following points should be emphasised:

- Neural networks are harder to implement than the simple CR and SBA methods but they do achieve significantly better performance in terms of $MAPE$.
- The choice of a longer time horizon can significantly decrease the $MAPE$ values, regardless of the training method or architecture chosen. This directly impacts the forecasting effectiveness, also from an inventory management point of view.
- The extreme learning machine training method is easier and faster to implement, but may produce less effective results. If possible, a back-propagation algorithm is preferable.
- If the back-propagation training methodology is chosen, we recommend implementing the MUK network. It achieves better results in terms of ME/A and could also lead to better performance in terms of $MAPE$.
- The forecasting performance in terms of $MAPE$ is directly and linearly related to the ADI. The more erratic the time series is, the worse the forecasting performance becomes. This happens regardless of the methodology applied and can significantly affect inventory management. On the contrary, the systematic error committed by the networks is not linearly related to the ADI. This implies that the forecast-oriented categorisation of demand patterns has to be realised only in terms of ADI when single-hidden layer neural networks are adopted.

10. Conclusions and extensions

Accurate forecasts are of fundamental importance for an efficient inventory control system. This is evident in the case of intermittent demand, especially when the purchase and backorder costs are high. Spare parts management is one example of this, where expensive items requiring high availability in order not to appear in backorders typically show intermittent profiles. Single-hidden layer neural networks represent a promising approach, whose main strength is their ability to generalise non-linear functions without the need for distribution assumptions. However, few papers have been devoted to investigating their forecasting accuracy for intermittent demand, and they have not been widely implemented in real industrial environments due to the intensive computational efforts required by the gradient descent-based algorithms for training the networks. Back-propagation belongs to this family of algorithms, and has been typically adopted into this research field.

In order to improve the understanding of these predictors in the field of intermittent demand forecasting, this paper provides a comparison between standard methods, i.e. CR (Croston, 1972) and SBA (Syntetos and Boylan, 2005), and a set of ANNs with different input patterns, architectures (feedforward, time-delay, and recurrent), and trained by two different learning approaches (back-propagation and extreme learning machines), on twenty-four real time series. Two accuracy metrics, updated on both a periodic and a rolling base, were applied for three forecasting horizons (i.e. 1, 3 and 5 periods ahead). Thus, the comparison of the aforementioned methods was designed to be as complete as possible in order to investigate the main aspects affecting forecasting accuracy and provide some useful guidelines for practitioners. In particular, the extreme learning machines have been tested for the first time as predictors of intermittent demand because of their lean implementation in comparison with the back-propagation.

The statistical analysis has revealed the overall superior performance of the back-propagation in terms of $MAPE$, while the systematic error (i.e. bias) does not change significantly. Among the networks trained by back-propagation, the two-input network proposed by Mukhopadhyay et al. (2012) has shown the lower bias over all the time horizons under comparison, while significant differences were not identified in terms of $MAPE$. Moreover, the forecasting accuracy of all the methods in terms of $MAPE$ has almost doubled by augmenting the forecasting horizon from one to three periods ahead. This result is meaningful from an inventory point of view, when setting the review interval of periodic review inventory systems.

Finally, only ADI has shown a linear effect on $MAPE$ with a satisfactory significance level.

Although the back-propagation revealed better performances, and thus should be suggested to practitioners, the price of this increased performances is paid by the higher computational efforts during the training stage. Conversely, extreme learning machines are easier to implement. Investigating the possibility of improving their accuracy as predictors of the intermittent demand may be interesting for further research in the future. Some possible directions may include the investigation on the impact of the distribution form for generating the hidden layer parameters, and the optimisation of the number of hidden nodes by using a threefold partition of time series (i.e. sets of training, optimisation of nodes, and test). Furthermore, given the easy implementation of the extreme learning machines, they could be experienced both on real and on pseudo-randomly generated large dataset.

References

Abdel-Aal, R.E., 2008. Univariate modeling and forecasting of monthly energy demand time series using abductive and neural networks. Comput. Ind. Eng. 54, 903–917. doi:10.1016/j.cie.2007.10.020

Babai, M.Z., Syntetos, A., Teunter, R., 2014. Intermittent demand forecasting: An empirical study on accuracy and the risk of obsolescence. Int. J. Prod. Econ. 157, 212–219. doi:10.1016/j.ijpe.2014.08.019

Bishop, C.M., 1995. Neural Networks for Pattern Recognition, Journal of the American Statistical Association. doi:10.2307/2965437

Boylan, J.E., Syntetos, A.A., 2010. Spare parts management: A review of forecasting research and extensions. IMA J. Manag. Math. 21, 227–237. doi:10.1093/imaman/dpp016

Caire, P., Hatabian, G., Muller, C., 1992. Progress in forecasting by neural networks. [Proceedings 1992] IJCNN Int. J. Conf. Neural Networks 2, 540–545. doi:10.1109/IJCNN.1992.226932

Carmo, J.L., Rodrigues, A.J., 2004. Adaptive forecasting of irregular demand processes. Eng. Appl. Artif. Intell. 17, 137–143. doi:10.1016/j.engappai.2004.01.001

Chakraborty, K., Mehrotra, K., Mohan, C.K., Ranka, S., 1992. Forecasting the behavior of multivariate time series using neural networks. Neural Networks 5, 961–970. doi:10.1016/S0893-6080(05)80092-9

Chan, Z.S.H., Ngan, H.W., Rad, A.B., David, A.K., Kasabov, N., 2006. Short-term ANN load forecasting from limited data using generalization learning strategies. Neurocomputing 70, 409–419. doi:10.1016/j.neucom.2005.12.131

Croston, J.D., 1972. Forecasting and Stock Control for Intermittent Demands. J. Oper. Res. Soc. doi:10.1057/jors.1972.50

Denton, J.W., 1995. How good are neural networks for causal forecasting? J. Bus. Forecasting 14, 17–20.

Elman, J.L., Zipser, D., 1988. Learning the hidden structure of speech. J. Acoust. Soc. Am. 83, 1615–1626. doi:10.1121/1.395916

Ertugrul, Ömer Faruk, 2016. Forecasting electricity load by a novel recurrent extreme learning machines approach. Int. J. Electr. Power Energy Syst. 78, 429–435. doi:10.1016/j.ijepes.2015.12.006

Fun, M.-H., Hagan,M.T., 1996. Levenberg–Marquardt training for modular networks. IEEE Int. Conf. on Neural Networks 1, 468–473. doi:10.1109/ICNN.1996.548938

Gamberini, R., Lolli, F., Rimini, B., Sgarbossa, F., 2010. Forecasting of sporadic demand patterns with seasonality and trend components: An empirical comparison between holt-winters and (s)ARIMA methods. Math. Probl. Eng. 2010. Article ID 579010, 14 pages. http://dx.doi.org/10.1155/2010/579010

Gutierrez, R.S., Solis, A.O., Mukhopadhyay, S., 2008. Lumpy demand forecasting using neural networks. Int. J. Prod. Econ. 111, 409–420. doi:10.1016/j.ijpe.2007.01.007

Hamzaçebi, C., Akay, D., Kutay, F., 2009. Comparison of direct and iterative artificial neural network forecast approaches in multi-periodic time series forecasting. Expert Syst. Appl. 36, 3839–3844.

doi:10.1016/j.eswa.2008.02.042

Heinermann, J., Kramer, O., 2016. Machine learning ensembles for wind power prediction. Renew. Energy 89, 671–679. doi:10.1016/j.renene.2015.11.073

Hill, T., Marquez, L., O'Connor, M., Remus, W., 1994. Artificial neural network models for forecasting and decision making. Int. J. Forecast. 10, 5-15. doi:10.1016/0169-2070(94)90045-0

Hill, T., O'Connor, M., Remus, W., 1996. Neural Network Models for Time Series Forecasts. Manage. Sci. 42, 1082–1092. doi:10.1287/mnsc.42.7.1082

Ho, S.L., Xie, M., Goh, T.N., 2002. A comparative study of neural network and Box-Jenkins ARIMA modeling in time series prediction. Comput. Ind. Eng. 42, 371–375. doi:10.1016/S0360-8352(02)00036-0

Huang, G.-B, 2003. Learning capability and storage capacity of two-hidden-layer feedforward networks. IEEE Trans. Neural Networks 14, 274–281. doi:10.1109/TNN.2003.809401

Huang, G.-B, Babri, H.A., 1998. Upper bounds on the number of hidden neurons in feedforward networks with arbitrary bounded nonlinear activation functions. IEEE Trans. Neural Networks 9, 224–229. doi:10.1109/72.655045

Huang, G., Huang, G.-B, Song, S., You, K., 2015. Trends in extreme learning machines: A review. Neural Networks 61, 32–48. doi:10.1016/j.neunet.2014.10.001

Huang, G.-B, Zhu, Q., Siew, C., 2004. Extreme Learning Machine : A New Learning Scheme of Feedforward Neural Networks. IEEE Int. Jt. Conf. Neural Networks 2, 985–990. doi:10.1109/IJCNN.2004.1380068

Huang, G.-B., Zhu, Q., Siew, C., Ã, G.H., Zhu, Q., Siew, C., Huang, G.-B., Zhu, Q., Siew, C., 2006. Extreme learning machine: Theory and applications. Neurocomputing 70, 489–501. doi:10.1016/j.neucom.2005.12.126

Kang, S., 1991. An investigation of the use of feedforward neural networks for forecasting, Ph.D. Dissertation, Kent State.

Kohzadi, N., Boyd, M.S., Kermanshahi, B., Kaastra, I., 1996. A comparison of artificial neural network and time series models for forecasting commodity prices. Neurocomputing 10, 169–181. doi:10.1016/0925-2312(95)00020-8

Kourentzes, N., 2013. Intermittent demand forecasts with neural networks. Int. J. Prod. Econ. 143, 198–206. doi:10.1016/j.ijpe.2013.01.009

Lachtermacher, G., Fuller, J.D., 1995. Back propagation in time-series forecasting. J. Forecast. 14, 381–393. doi:10.1002/for.3980140405

Lian, C., Zeng, Z., Yao, W., Tang, H., 2014. Ensemble of extreme learning machine for landslide displacement prediction based on time series analysis. Neural Comput. Appl. 24, 99–107. doi:10.1007/s00521-013-1446-3

Lolli, F., Gamberini, R., Regattieri, A., Rimini, B., 2014a. Application of tramo-seats automatic procedure for forecasting intermittent demand patterns, in: OPT-I 2014 - 1st International Conference on Engineering and Applied Sciences Optimization. National Technical University of Athens, Kos Island - Greece, pp. 1435–1445.

Lolli, F., Gamberini, R., Regattieri, A., Rimini, B., Grassi, A., Belluti, P., 2011. Application of tramo-seats automatic procedure for forecasting sporadic and irregular demand patterns with seasonality, in: HMS 2011–13th International Conference on Harbor, Maritime and Multimodal Logistics Modeling and Simulation, Held at the International Mediterranean and Latin American Modeling Multiconference, I3M 2011, Rome, Italy, pp. 214–220, 2014a

Lolli, F., Gamberini, R., Regattieri, A., Rimini, B., Morsiani, M., 2014b. On the liaison between forecasting and periodic inventory control : application of sarima models to intermittent demand profiles, in: OPT-I 2014 - 1st International Conference on Engineering and Applied Sciences Optimization. National

Technical University of Athens, Kos Island - Greece, pp. 1426–1434.

Makridakis, S., Andersen, A., Carbone, R., Fildes, R., Hibon, M., Lewandowski, R., Newton, J., Parzen, E., Winkler, R., 1982. The accuracy of extrapolation (time series) methods: Results of a forecasting competition. J. Forecast. 1, 111–153. doi:10.1002/for.3980010202

Makridakis, S., Hibon, M., 2000. The M3-Competition: results, conclusions and implications. Int. J. Forecast. doi:10.1016/S0169-2070(00)00057-1

Mohammadi, K., Shamshirband, S., Yee, P.L., Petković, D., Zamani, M., Ch, S., 2015. Predicting the wind power density based upon extreme learning machine. Energy 86, 232–239. doi:10.1016/j.energy.2015.03.111

Mukhopadhyay, S., Solis, A.O., Gutierrez, R.S., 2012. The accuracy of non-traditional versus traditional methods of forecasting lumpy demand. J. Forecast. 31, 721–735. doi:10.1002/for.1242

Nasiri Pour, A., Rostami Tabar, B., Rahimzadeh, A., 2008. A hybrid neural network and traditional approach for forecasting lumpy demand. World Academy of Science, Engineering and Technology 40, 384–390.

Nguyen, M.H., Abbass, H.A., McKay, R.I., 2005. Stopping criteria for ensemble of evolutionary artificial neural networks. Appl. Soft Comput. J. 6, 100–107. doi:10.1016/j.asoc.2004.12.005

Penrose, R., Todd, J. A., 1955. A generalized inverse for matrices. Math. Proc. Cambridge Philos. Soc. 51, 406–413. doi:10.1017/S0305004100030401

Piotrowski, A.P., Napiorkowski, J.J., 2013. A comparison of methods to avoid overfitting in neural networks training in the case of catchment runoff modelling. J. Hydrol. 476, 97–111. doi:10.1016/j.jhydrol.2012.10.019

Prechelt, L., 1998. Automatic early stopping using cross validation: Quantifying the criteria. Neural Networks 11, 761–767. doi:10.1016/S0893-6080(98)00010-0

Qian, N., 1999. On the momentum term in gradient descent learning algorithms. Neural Networks 12, 145–151. doi:10.1016/S0893-6080(98)00116-6

Regattieri, A., Gamberi, M., Gamberini, R., Manzini, R., 2005. Managing lumpy demand for aircraft spare parts. J. Air Transp. Manag. 11, 426–431. doi:10.1016/j.jairtraman.2005.06.003

Rumelhart, D.E., Hinton, G.E., Williams, R.J., 1986. Learning representations by back-propagating errors. Nature 323, 533–536. doi:10.1038/323533a0

Serre, D., 2002. Matrices: Theory and applications, Springer. doi:10.1007/978-1-4419-7683-3

Sharda, R., Patil, R., 1992. Connectionist approach to time series prediction: an empirical test. J. Intell. Manuf. 3, 317–323. doi:10.1007/BF01577272

Sun, Z.-L., Choi, T.-M., Au, K.-F., Yu, Y., 2008. Sales forecasting using extreme learning machine with applications in fashion retailing. Decis. Support Syst. 46, 411–419. doi:10.1016/j.dss.2008.07.009

Syntetos, A.A., Zied Babai, M., Gardner, E.S., 2015. Forecasting intermittent inventory demands: simple parametric methods vs. bootstrapping. J. Bus. Res. 68, 1746–1752. doi:10.1016/j.jbusres.2015.03.034

Syntetos, A.A., Boylan, J.E., 2001. On the bias of intermittent demand estimates. Int. J. Prod. Econ. 71, 457-466. doi:10.1016/S0925-5273(00)00143-2

Syntetos, A.A., Boylan, J.E., 2005. The accuracy of intermittent demand estimates. Int. J. Forecast. 21, 303–314. doi:10.1016/j.ijforecast.2004.10.001

Syntetos, A.A., Boylan, J.E., Croston, J.D., 2005. On the categorization of demand patterns. J. Oper. Res. Soc. 56, 495–503 http://www.jstor.org/stable/41021.

Tang, Z., Fishwick, P.A., 1993. Feedforward Neural Nets as Models for Time Series Forecasting. INFORMS J. Comput. 5, 374–385. doi:10.1287/ijoc.5.4.374

Teunter, R., Sani, B., 2009. On the bias of Croston's forecasting method. Eur. J. Oper. Res. 194, 177–183.

doi:10.1016/j.ejor.2007.12.001

Teunter, R.H., Duncan, L., 2009. Forecasting intermittent demand: a comparative study. J. Oper. Res. Soc. 60, 321–329. doi:10.1057/palgrave.jors.2602569

Teunter, R.H., Syntetos, A.A., Babai, M.Z., 2011. Intermittent demand: Linking forecasting to inventory obsolescence. Eur. J. Oper. Res. 214, 606–615. doi:10.1016/j.ejor.2011.05.018

Wallström, P., Segerstedt, A., 2010. Evaluation of forecasting error measurements and techniques for intermittent demand. Int. J. Prod. Econ. 128, 625–636. doi:10.1016/j.ijpe.2010.07.013

West, D., Dellana, S., 2011. An empirical analysis of neural network memory structures for basin water quality forecasting. Int. J. Forecast. 27, 777–803. doi:10.1016/j.ijforecast.2010.09.003

Willemain, T.R., Smart, C.N., Shockor, J.H., DeSautels, P. a., 1994. Forecasting intermittent demand in manufacturing: a comparative evaluation of Croston's method. Int. J. Forecast. 10, 529–538. doi:10.1016/0169-2070(94)90021-3

Xiang, C., Ding, S.Q., Lee, T.H., 2005. Geometrical interpretation and architecture selection of MLP. IEEE Trans. Neural Networks 16, 84–96. doi:10.1109/TNN.2004.836197

Zhang, G.P., Qi, M., 2005. Neural network forecasting for seasonal and trend time series. Eur. J. Oper. Res. 160, 501–514. doi:10.1016/j.ejor.2003.08.037