Deep Multiplicative State Space Models for Intermittent Time Series Forecasting

(Głębokie multiplikatywne modele przestrzeni stanów
do prognozowania przerywanych szeregów czasowych)

Tomasz Nanowski

Praca magisterska

Promotor: dr hab. Jan Chorowski

Uniwersytet Wrocławski
Wydział Matematyki i Informatyki
Instytut Informatyki

16 marca 2021

Abstract

Product demand forecasting is crucial for logistics and production to have the right item in the right place at the right time. This affects the performance of the entire supply chain. The problem of predicting future consumption becomes especially challenging for intermittent demand, where demand occurrences appear sporadically in time. This pattern is often experienced in slow-moving, high-value items, which happens in e-commerce, manufacturing, and automotive industries. However, many of the forecasting methods may perform poorly for this kind of item demand. Moreover, the typical methods for this problem do not have underlying statistical models, so they do not provide access to estimates of forecast uncertainty. It is essential in an intermittent demand context where inventory control is inherently difficult and meaningful.

This thesis proposes an approach to probabilistic time series forecasting that combines state-space models with deep learning. The usage of the state-space model provides interpretability and statistical assumptions. Simultaneously, the deep learning approach offers the ability to learn complex patterns from multiple time series using additional input like weather or calendar events. This is ensured by the separate parameterization of the state space model per time series via a recursive neural network.

I then conducted experiments on simulated data and two real data sets with intermittent time series. Also, I tested the performance of the model on data consisting of three other demand types. In all cases, I compared results with the best baseline models, using a variety of relevant metrics. Finally, it presented the advantages of the proposed approach.

Streszczenie

Prognozowanie popytu ma kluczowe znaczenie dla logistyki i produkcji, aby mieć właściwy towar we właściwym miejscu i czasie. Wpływa to na wydajność całego łańcucha dostaw. Problem przewidywania przyszłej konsumpcji staje się szczególnie trudny w przypadku popytu przerywanego, gdzie zapotrzebowanie pojawia się sporadycznie w czasie. Taki wzorzec jest często spotykany w przypadku wolno rotujących artykułów o wysokiej wartości, co ma miejsce w handlu elektronicznym, przemyśle wytwórczym i motoryzacyjnym.Jednakże, wiele metod prognozowania może nie sprawdzać się w przypadku tego rodzaju popytu. Co więcej, typowe metody dla tego problemu nie posiadają bazowych modeli statystycznych, więc nie zapewniają dostępu do szacunków niepewności prognozy. Jest to niezwykle istotne w kontekście popytu przerywanego, gdzie kontrola zapasów jest z natury trudna i znacząca.

W niniejszej pracy zaproponowano podejście do probabilistycznego prognozowania szeregów czasowych, które łączy modele przestrzeni stanów z głębokim uczeniem. Wykorzystanie modelu przestrzeni stanów zapewnia interpretowalność i przyjęcie założeń statystycznych. Jednocześnie, podejście głębokiego uczenia oferuje zdolność do uczenia się złożonych wzorców z wielu szeregów czasowych przy użyciu dodatkowych danych wejściowych, takich jak pogoda lub wydarzenia kalendarzowe. Jest to zapewnione przez oddzielną parametryzację modelu przestrzeni stanów dla każdego szeregu czasowego przy pomocy rekurencyjnej sieci neuronowej.

Następnie przeprowadziłem eksperymenty na danych symulowanych i dwóch rzeczywistych zestawach danych z przerywanymi szeregami czasowymi. Ponadto, przetestowałem działanie modelu na danych składających się z trzech innych typów zapotrzebowania. We wszystkich przypadkach, porównałem wyniki z najlepszymi modelami bazowymi, używając różnych i odpowiednich metryk. W końcu, przedstawiłem zalety proponowanego podejścia.

Contents

Acknowledgments

I want to thank my supervisor, dr hab. Jan Chorowski from the Institute of Mathematics and Computer Science at the University of Wrocław, for his care, assistance, and substantive support during my work on this project and bringing it to completion.

Of course, I am also very grateful to my immediate family for their support. I will miss these incessant questions about progress in writing this report at every possible opportunity.

Chapter 1

Introduction

"Prediction is very difficult, especially if it's about the future"

- Nils Bohr, *Nobel laureate in Physics*

Accurate forecasting of time series is one of the most critical factors in many industrial and business decision-making processes. The final effectiveness of each decision depends on the subsequent events following the operation. An important example of such a task is demand forecasting, a fundamental aspect of supply chain management. It is essential that the right product is available in the right place at the right time, taking into account the optimization of logistics and production costs.

A large proportion of inventory catalogs in e-commerce, manufacturing, and logistics are plagued by "intermittency", where positive demand occurs at irregular frequency. In general, this pattern concerns rare events and the occurrence of demand for slow-moving, high-values items. In the case of intermittent demand, which refers to the sporadic occurrence, it contains only zeroes and positive, usually low integer values. For example, intermittent demand can be identified in products, such as heavy machinery and respective spare parts, aircraft service parts, electronics, and marine spare parts.

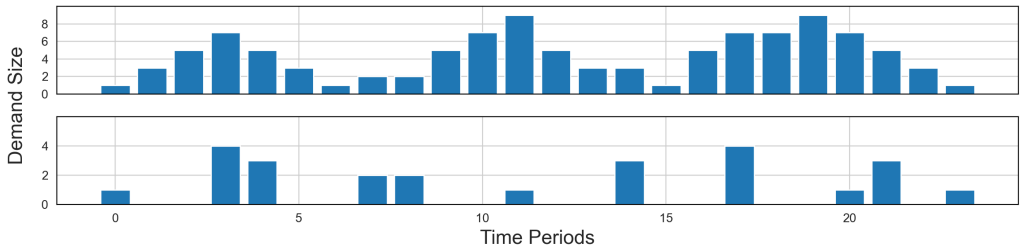

Figure 1.1: Comparing of non-intermittent time series (above) with intermittent time series (below).

State-space models (SSMs), as a statistical model and by using a time series structure, provide a principled framework for modeling and learning patterns. Therefore, SSMs are particularly well suited for applications where the time series structure is well understood, as it allows these assumptions to be incorporated into the model. By defining such components, it is possible to interpret the model, and the learning procedure becomes data-efficient but requires a sufficiently long history. However, SSMs are not sufficient for much diverse time series and cannot infer a pattern from a group of series with a similar structure. It makes them challenging to forecast time-series with little or no history using SSMs.

Deep Learning, specifically Deep Neural Networks (DNN), has shown excellent performance in many different fields. They can identify higher-order features, and recurrent neural networks (RNNs) can identify complex patterns within and between time series. They can do so based on a set of raw time series with limited additional human input. However, as these models have fewer structural assumptions, they require more data. Besides, these models are difficult to interpret and control assumptions.

1.1 Problem motivation

Modern forecasting applications require the ability to process huge collections consisting of even thousands of time series. This can now be seen very well in e-commerce, where a supply chain is required for a vast number of varied products. Such products often have just intermittent demand dynamics, or they become so when demand for them ends. Besides, in e-commerce, where new products are continually appearing and there is no sales history, it is necessary to extract this knowledge from the group of products with a similar demand pattern. An additional need is to include external information in preparing forecasts such as weather or other special calendar events.

Furthermore, from a practical point of view of the supply chain, decisions on supplements should be based on reliable estimates. Reasonable estimates of the probability of demand are required for systems based on filling rates (percentage of immediate demand from stocks). However, for stockholding systems based on the probability of depletion, good estimates of the upper percentiles of demand are required.

1.2 Related work

Large parts of inventory catalogs in e-commerce, manufacturing, and logistics are plagued by "intermittency". The standard approach to intermittent demand forecast, and the first paper to address it, is the method of Croston [3]. This method

has then been studied for a very long time, and many other models based on it have been developed. In [33], the authors proposed an adjustment, known as the Syntetos-Boylan Approximation (SBA), to Croston's forecast. In contrast, Teunter et al. [35] suggested a different way to model the probability of demand occurrence than in Croston's approach. Snyder et al. [28] used the Poisson distribution rather than the Bernoulli to model the occurrence of demand. The problem of forecasting intermittent demand was further developed by Petropoulos and Kourentzes [21], Pour et al. [22], Teunter and Duncan [34], and Willemain et al. [39].

Hyndman et al. [11], Durbin and Koopman [5], and Murphy [18] presented a comprehensive review of the SSMs. The most recent literature work on machine learning in the field of state-space models includes Seeger et al. [27]. Svetunkov and Boylan [31] have proposed multiplicative state spaces for intermittent time series needs, which have achieved satisfactory results.

In time series forecasting, neural networks have been investigated for a long time, particularly recent work considering LSTM cells' use. In the case of intermittent data, Kourentzes [15] applied neural networks but obtained mixed results. Moreover, Turkmen et al. [36] used recurrent neural networks in the context of Croston's model. Usually, in forecasting, neural networks are applied to each time series separately. On the other hand, apart from the forecasting community, time series models using recurrent neural networks are successfully applied in other areas such as sound modeling, natural language processing, and image generation.

Merging state-space models (SSM) with the RNN has already been proposed before by Rangapuram et al. [24] and Salinas et al. [25]. In the first case, the RNN parameterizes the transitions between the Gaussian states. The second one is the recent probabilistic forecasting method, based on autoregressive recurrent networks. However, none of these methods were for intermittent demand, which requires different treatment due to its unique pattern.

1.3 The goal

Accurate forecasting of intermittent time series is essential for making optimal decisions in the logistics and production industry. Therefore, it is a significant but also tricky problem, which has already been studied for decades.

Recent approaches combining traditional methods of mathematical modeling with modern deep neural networks have shown excellent efficiency. This results from the use of the advantages of both of these tools, that is to say, the already extensive knowledge and transparency of the long examined classical methods with a tool capable of detecting very intricate patterns.

Therefore, this project aims to prepare a model that combines state-space models with deep recurrent neural networks to solve the intermittent time series fore-

casting.

1.4 Report structure

This report is structured in the following way:

1. **Introduction**: Introduction to and motivation of the problem, previous related and useful work, and the project's goal.
2. **Background**: The general theory of the different fields, methods, and models used in the project, including time-series basics, the general state-space approach to forecasting, and neural networks.
3. **Methods**: Details about the proposed approach with a description of the training and inference procedure and forecasting methods.
4. **Experiments**: Presentation of results from the performed experiments and comparison of models.
5. **Conclusions**: The conclusions that can be drawn from the results are listed.

Chapter 2

Background

2.1 Time Series

Predicting the future has fascinated humanity for centuries and is a critical element in planning and decision-making. It is used in such everyday things as tomorrow's weather, up to stock prices for a few months in advance. Thus, it is already an indispensable part of everyday life, and correct forecasts have an extremely positive impact on the community's functioning and optimization of many processes. On the other hand, incorrect predictions can have very negative consequences. Moreover, since this problem concerns the future, which can take shape in a completely unpredictable way, it is not easy to verify.

Therefore, people have tried different approaches to know the future. In ancient Babylon, the analysis of animal guts has been done, and in ancient Greece, questions have been asked to the Delphic oracle. However, it turned out that misguided predictions could be so damaging that they were banned in Byzantium. They were also severely punished in the 18th century in the United Kingdom as the cause of many deceptions. However, this thesis will focus on civilizations' recent achievements in this area, using mathematical modeling, which seems to be more effective than centuries ago.

Before going any further, it is essential to introduce standard terminology used in a time series context.

Definition 2.1. A time series is a series of data points indexed (or listed or graphed) in time order. Most commonly, a time series is a sequence taken at successive equally spaced points in time. Thus it is a sequence of discrete-time data.

Analysis and prediction of time series are significant issues in statistics and machine learning. Nevertheless, they are often overlooked and taken for granted. However, they deserve a great deal of attention because their use is ubiquitous, and they can help predict sales, economists, budget analysis, and the stock market,

as an example. Moreover, even though their use is almost inextricably linked to industry and social science, time series analysis and prediction are among the least understood machine learning methods by data engineers.

2.1.1 Time Series Forecasting

Forecasting involves preparing models based on historical data and using them to predict future observations. An essential difficulty in forecasting is that the future is entirely inaccessible and must be estimated only based on what has already happened. Therefore, time series forecasting is a difficult problem. In some respects, it is a more difficult problem than typical classification or regression. It requires taking into account the relationship between observations through time. Besides, there may be a seasonal trend, which makes the data variability depend on time.

The fact that predicting the future is a difficult task is perfectly illustrated by examples from the past when experts in their fields have made mistakes in their future assumptions. Consider the following examples of time-verified failed predictions:

- "I think there is a world market for maybe five computers.", *Thomas Watson, president of IBM, 1943.*
- "I predict the Internet will soon go spectacularly supernova and in 1996 catastrophically collapse.", *Robert Metcalfe, founder of 3Com, 1995.*
- "There is no reason anyone would want a computer in their home.", *Ken Olsen, co-founder of DEC, 1997.*
- "There's no chance that the iPhone is going to get any significant market share.", *Steve Ballmer, Microsoft CEO, 2007.*

So, as can be seen, even people with close links to the industries concerned can make a catastrophic mistake that costs them much.

Here are some examples of how time forecast is used in the industry:

- Energy: in the energy sector, it is necessary to plan the electricity demand due to the difficulties in storing it.
- Retail: forecasting demand for specific products and planning logistics operations in this connection.
- State government: predicting tax receipts.
- Transport: planning future travel needs and managing the flow of goods.

Its effectiveness determines the ability of a time series forecasting model in predicting the future. This is often at the expense of explaining why a particular forecast was made, the confidence intervals, and even a better understanding of the underlying causes.

A forecasting problem that requires predicting the next step in time is called single-step forecasting. In contrast, a problem that requires forecasting more than a one-time step is called multi-step forecasting. As the length of the prediction interval increases, the problem is more and more difficult because each step carries more estimation uncertainty.

2.1.2 Probabilistic forecasting

Probabilistic prediction estimates the probability distribution of future time series values based on the past instead of predicting a single number. Furthermore, while such predictions contain much more information, they are also more difficult to obtain accurately. Therefore, a simplified version of the problem is sometimes considered to forecast the quantiles of distribution, which is sufficient to solve the problem.

2.1.3 Demand forecasting

Demand forecasting is a forecasting analysis field that seeks to understand and predict customer demand to optimize purchasing decisions by corporate supply chain and business management. Demand forecasting includes data-driven methods, especially historical sales data and statistical techniques. Understanding and forecasting customer demand for products is essential for manufacturers and distributors to plan their logistics adequately, thereby avoiding inventory build-up and maintaining adequate product quantities. Although forecasts are never perfect, they are essential to prepare for actual demand. To this end, maintaining optimal inventories and planning an efficient supply chain requires accurate demand forecasts.

This thesis focuses mainly on the problem of demand forecasting. This means considering a time series in which the values are non-negative and integer. This type of time series is also called count data time series.

2.1.4 The categorization of demand patterns

When it comes to predicting demand, time series can be categorized in a certain way. It turns out that some time series are more comfortable to forecast, and others are impossible. This can be seen in the Figure 1.1. Therefore, the accuracy of a given product's forecasting method depends on the characteristics of the product's historical demand. For this reason, demand time series are sometimes divided into

several discrete categories in order to assign the best forecasting method.

Williams [40] initially conceived the idea of categorizing demand patterns. He tested products' classification by type of demand, stock control policies for different product categories, and demand forecasting methods for different product categories. Similar ideas for categorizing demand patterns for forecasting and stock control purposes are considered in Eaves and Kingsman [6]. However, Syntetos et al. [32] proposed a more formal way of categorizing time series. For this classification, to determine a product forecastability, we apply two coefficients:

- **the Average Demand Interval (ADI)**: it measures the demand regularity in time by computing the average interval between two demands.
- **the square of the Coefficient of Variation (CV^2)**: it measures the variation in quantities.

Based on these properties, they classified the demand profiles into 4 different categories:

1. **Smooth demand** ($ADI < 1.32$ and $CV^2 < 0.49$): the demand is very regular in time and quantity.
2. **Intermittent demand** ($ADI \geq 1.32$ and $CV^2 < 0.49$): the demand history shows a tiny variation in demand quantity but a high variation in the interval between two demands.
3. **Erratic demand** ($ADI < 1.32$ and $CV^2 \geq 0.49$): the demand has regular occurrences in time with high quantity variations.
4. **Lumpy demand** ($ADI \geq 1.32$ and $CV^2 \geq 0.49$): the demand is characterized by a large variation in quantity and in time. It is actually impossible to produce a reliable forecast, no matter which forecasting tools you use. This particular type of demand pattern is unforecastable.

Example series from each category are presented in Figure 2.1.

2.1.5 Traditional Methods for Time Series Forecasting

In this section, I present one of the classical and simplest time series forecasting methods for the needs of later considerations.

2.1.5.1 Simple exponential smoothing

Exponential smoothing was introduced in the late 1950s and is the basis of some of the most successful forecasting methods. Forecasts produced using this approach are weighted averages of previous observations, with the weights decreasing

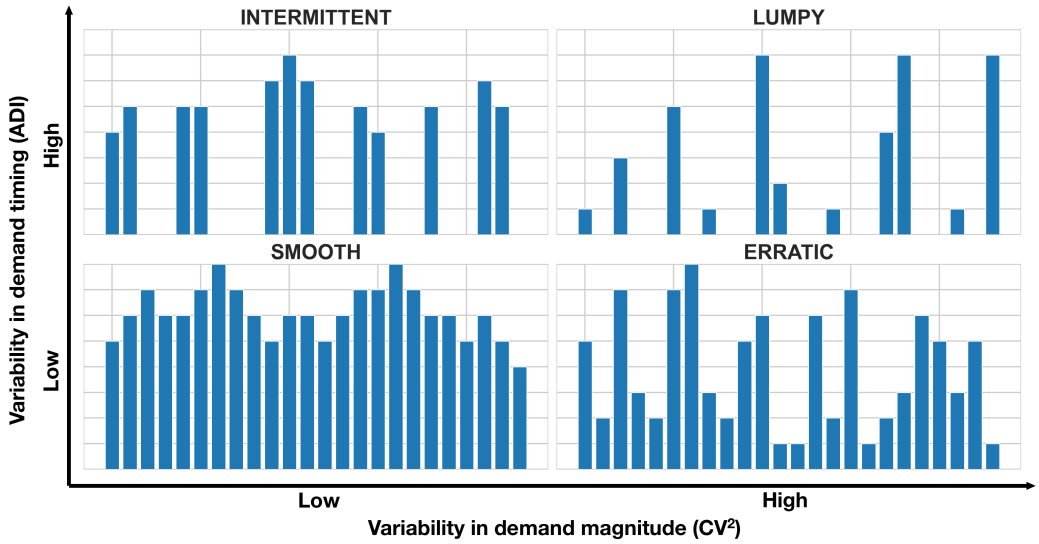

Figure 2.1: The categorization of demand patterns.

exponentially as the age of the observations. This means that the more recent observation, the greater the associated weight, and older observations have less influence on the outcome. This framework generates reliable forecasts quickly and for a wide range of time series, which is advantageous and essential for industrial applications. These methods are well described by Hyndman and Athanasopoulos [12].

The simplest of the exponential smoothing methods, which assumes the lack of trend and seasonality in the data, is called simple exponential smoothing (SES). The following system of equations describes it:

$$\hat{y}_{t+h|t} = l_t \tag{2.1}$$

$$l_t = \alpha y_t + (1-\alpha) l_{t-1} \tag{2.2}$$

where l_t is the level at time t and $\alpha \in [0, 1]$ is the smoothing parameter. The smaller α is, the more past observations have a more significant influence on the forecast.

An extension of this approach that additionally takes trend into account is Holt's method, and another extension that takes seasonality into account is the Holt-Winters' method.

2.1.6 Methods for Intermittent Demand Forecasting

The many zero values present in the intermittent time-series make typical forecasting methods challenging to apply. For example, simple exponential smoothing (SES, 2.1.5.1) performs poorly in forecasting intermittent timeseries. Croston was the first to note that when demand is intermittent, there is an upward bias in the SES's forecast in the period directly after a non-zero demand.

In the remainder of this section, some methods for forecasting intermittent time

series are presented in more detail.

2.1.6.1 Croston's method

Croston [3] proposed the most popular method for forecasting intermittent demand. His method has been widely studied in recent years and is widely used in the industry. He proposed to divide the observed data into two parts: demand volume and demand incidence. The proposed Croston model has the following form:

$$y_t = o_t z_t \tag{2.3}$$

where y_t is the actual observation, o_t is a binary value from the Bernoulli distribution, indicating the demand occurrence. Also, z_t is the quantity of demand, with a conditional distribution. Each of these two parts is modelled separately. Croston proposed to estimate the occurrence using intervals between demand. For this purpose, he introduced the q_t variable, which represents the demand interval between consecutive non-zero demands. Both demand sizes z_t, and demand intervals q_t are forecasted separately using simple exponential smoothing (2.1.5.1). This leads to the following system of equations:

$$\hat{y}_{j_t} = \frac{1}{\hat{q}_{j_t}} \hat{z}_{j_t} \tag{2.4}$$

$$\hat{z}_{j_t} = \alpha_z z_{j_t-1} + (1-\alpha_z)\, \hat{z}_{j_t-1} \tag{2.5}$$

$$\hat{q}_{jt} = \alpha_q q_{jt-1} + (1-\alpha_q)\, \hat{q}_{j_t-1} \tag{2.6}$$

$$\hat{y}_t = \hat{y}_{j_t} \tag{2.7}$$

$$j_t = j_{t-1} + o_t \tag{2.8}$$

where $\hat{y}_{j_t}$ is the predicted mean demand, $\hat{z}_{j_t}$ is the predicted demand size, $\hat{q}_{jt}$ is the predicted demand interval, α_q and α_z are smoothing parameters for intervals and sizes.

Syntetos and Boylan [33] proved that the original method is biased. To overcome this problem, they proposed a modified version that gives a more accurate forecasts (known as Syntetos-Boylan approximation, SBA):

$$\hat{y}_t = \left(1 - \frac{\alpha_q}{2}\right) \frac{1}{\hat{q}_t} \hat{z}_t \tag{2.9}$$

2.1.6.2 TSB method

A method incorporating the possibility of item obsolescence was proposed by Teunter et al. [35]. The authors proposed to apply the same principle as Croston, but to estimate the time-varying probability of demand occurrence p_t using simple exponential smoothing rather than modeling the intervals between demands. The

Teunter, Syntetos, and Babai (TSB) method can be presented using the following set of equations:

$$\hat{y}_t = \hat{p}_t \hat{z}_t \tag{2.10}$$

$$\hat{z}_{j_t} = \alpha_z z_{j_{t-1}} + (1 - \alpha_z)\, \hat{z}_{j_{t-1}} \tag{2.11}$$

$$\hat{p}_t = \alpha_p o_{t-1} + (1 - \alpha_p)\, \hat{p}_{t-1} \tag{2.12}$$

$$\hat{z}_t = \hat{z}_{j_t} \tag{2.13}$$

However, this approach as well as this proposed by Croston, doesn't have statistical model. Nevertheless, this method's advantage is that the conditional expectation does not need any corrections similar to Syntetos and Boylan [33].

2.1.6.3 iETS model

Svetunkov and Boylan [31] proposed a different approach to intermittent series forecasting. In their paper, they presented a statistical model that derives from state-space models and takes into account the intermittency of data. Furthermore, it uses the same principle as Croston in its assumptions about dividing observations into two components. The following system of equations can present the basis of their method iETS:

$$y_t = o_t l_{z,t-1} (1 + \epsilon_t) \tag{2.14}$$

$$l_{z,t} = l_{z,t-1} (1 + \alpha_{z,t} \epsilon_t) \tag{2.15}$$

$$(1 + \epsilon_t) \sim \log \mathcal{N}\left(\mu_\epsilon, \sigma^2_{\epsilon,t}\right) \tag{2.16}$$

Furthermore, they propose few different approaches to modeling the distribution of the variable o_t, i.e., the probability of demand occurrence. They also presented a framework to select the best version automatically. This model allows for a systematic approach to method selection, rigorous parameterization, and estimation of upper percentiles of demand.

2.1.7 Traditional Accuracy Metrics

The effectiveness of any forecasting method must be evaluated using some indicator to measure how closely the forecasted value matches the actual one. Depending on the measure chosen, forecasts can produce quite different results, making it very difficult to assess correctly. Intermittent or lumpy demand series is challenging because typical forecasting accuracy indicators are often inadequate or impossible to use. Furthermore, since sets of time series are considered in this thesis, they require metrics that allow meaningful aggregation of multiple values. Therefore, additional scale-independent variants of them will be introduced when defining metrics.

The following sections contain examples of traditional forecasting accuracy metrics, where T is the number of in-sample observations, H is the total length of the

prediction range, y_{T+h} is the actual value of the h^{th} out-of-sample period and $\hat{y}_h$ is the h-steps-ahead forecast.

2.1.7.1 Mean Signed Error

The Mean Signed Error (ME) of the forecasts is measured to determine if an examined approach is consistently positively or negatively biased.

$$\text{ME} = \frac{1}{H} \sum_{t=1}^{H} (y_{T+t} - \hat{y}_t) \tag{2.17}$$

To average across all series, we scale the out-of-sample mean error of each series using the mean value of all in-sample periods. Thus, the scaled Mean Signed Error (sME) is given by:

$$\text{sME} = \frac{\text{ME}}{\frac{1}{T} \sum_{t=1}^{T} y_t} \tag{2.18}$$

2.1.7.2 Mean Squared Error

The Mean Squared Error (MSE) measure the performance in terms of variance.

$$\text{MSE} = \frac{1}{H} \sum_{t=1}^{H} (y_{T+t} - \hat{y}_t)^2 \tag{2.19}$$

In order to make the measure scale independent, it is scaled by the squared average of the actual demands. So, the scaled Mean Squared Error (sMSE) is given by:

$$\text{sMSE} = \frac{1}{H} \sum_{t=1}^{H} \left(\frac{y_{T+h} - \hat{y}_h}{\frac{1}{T} \sum_{t=1}^{T} y_t} \right)^2 = \frac{\text{MSE}}{\left(\frac{1}{T} \sum_{t=1}^{T} y_t \right)^2} \tag{2.20}$$

2.1.7.3 Mean Absolute Percentage Error

The Mean Absolute Percentage Error (MAPE) is a measure of prediction accuracy of a forecasting method. It usually expresses the accuracy as a ratio defined by the formula:

$$\text{MAPE} = \frac{1}{H} \sum_{t=1}^{H} \left| \frac{y_{T+t} - \hat{y}_t}{y_{T+t}} \right| \tag{2.21}$$

Percentage errors have the advantage of being independent of scale but have the disadvantage of being infinite or undefined if there are zero values in the series. Furthermore, percentage errors can have an extremely skewed distribution when the actual values are close to zero.

Moreover, MAPE imposes a larger penalty on positive errors than on negative errors. The symmetric Mean Absolute Percentage Error (sMAPE) is therefore considered:

$$\text{SMAPE} = \frac{1}{H}\sum_{t=1}^{H}\frac{|\hat{y}_t - y_{T+t}|}{(|y_{T+t}| + |\hat{y}_t|)/2} \tag{2.22}$$

However, if the actual value is zero, the prediction will likely be close to zero, so the measurement will still require dividing by a number close to zero. Also, it is not entirely symmetric since over- and under-forecasts are not treated equally.

2.1.7.4 The unsuitability of Traditional Accuracy Metrics

For intermittent or lumpy demand patterns where there are a large number of zero demand cases, the most popular metrics, such as MAPE and MSE, are not suitable for assessing forecasting errors [34, 38]. They do not sufficiently consider, for example, dividing by zero or temporal shifts (prediction before or after actual demand), as shown in the Figure 2.2. The bias measure ME can break down in certain circumstances, while the mean square error and, most importantly, the mean absolute error focus on periods with zero demand, favoring distorted forecasts. Consequently, most existing metrics only work with smooth and linear patterns but become less accurate or even useless as the frequency of intermittent patterns increases.

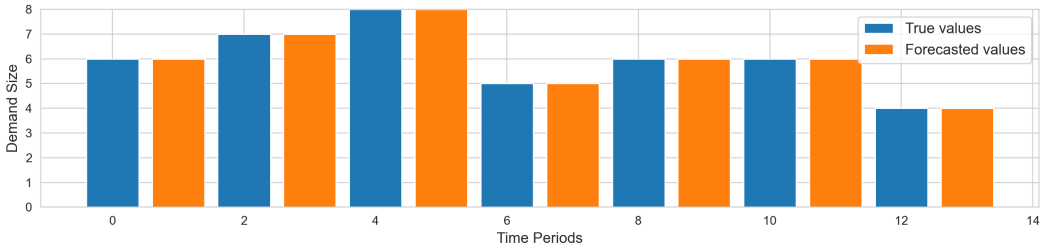

Figure 2.2: Example of a time series in which the forecasted values are shifted by one in time, relative to the true values. Which, for metrics such as RMSE, will lead to large errors. However, from an inventory supply perspective, such forecasts could be acceptable to manage inventory over the long term.

2.1.8 New and Recommended Accuracy Metrics

Because of the problems with traditional metrics described in the subsubsection 2.1.7.4, other special metrics have been prepared for intermittent time series. Some of them are directly linked to inventory control.

2.1.8.1 Mean Absolute Scaled Error

The Mean Absolute Scaled Error(MASE) was proposed by Hyndman and Koehler [10] as a generally applicable measure of forecast accuracy. They proposed scaling errors based on the in-sample MAE from the naïve forecasting method. Using the naïve method, we generate one-period-ahead forecasts from each data point in the sample. The formula is as follows:

$$\text{MASE} = \frac{1}{H}\sum_{t=1}^{H}\frac{|y_{T+t} - \hat{y}_t|}{\frac{1}{T-1}\sum_{i=2}^{T}|y_i - y_{i-1}|} \tag{2.23}$$

MASE is scale-independent, relative (so forecast error measures for different products might be compared), and could be computed for almost all intermittent demand series. If MASE > 1, then a given forecasting method is worse than the average naïve method. Otherwise (MASE < 1), the analyzed forecasting method is better than the naïve method. However, it was shown that MASE is equivalent to the weighted arithmetic mean of MAEs [4], so it induces a bias towards overrating benchmark and is affected by extreme cases when the MAE of benchmark forecast is relatively low. MASE is also vulnerable to outliers or structural breaks in time series history [13].

2.1.8.2 Cumulative Forecast Error

Cumulative Forecast Error (CFE) is the cumulative error over the entire prediction horizon, with negative and positive errors canceling each other out. This has direct implications to over or under stocking in a supply chain. A zero CFE can also happen due to "luck", and we can omit many details in between, so two additional versions are also being considered. $\text{CFE}_{\max}$ is equal to the greatest shortage during the forecast, and $\text{CFE}_{\min}$ is equal to the greatest surplus.

$$\text{CFE}_t = \sum_{i=1}^{t}(y_{T+i} - \hat{y}_i) = y_{T+t} - \hat{y}_t + \text{CFE}_{t-1} \tag{2.24}$$

$$\text{CFE}_{\min} = \min_{t\epsilon[1,2,\ldots H]} \text{CFE}_t \tag{2.25}$$

$$\text{CFE}_{\max} = \max_{t\epsilon[1,2,\ldots H]} \text{CFE}_t \tag{2.26}$$

2.1.8.3 Number of Shortages

Number of Shortages (NOS) measure, more commonly represented as Percentage of Number of Shortages (NOSp), counts the number of instances where the cumulative forecast error (CFE) was greater than zero, resulting in a shortage. Few or none NOS may indicate the method creates a fictive surplus stock because demand is overestimated. The reverse situation where almost every demand results in

a backorder is also a sign of bias.

$$\text{If } X_t \neq 0 \text{ and } \text{CFE}_t > 0 \text{ then } \text{NOS} \leftarrow \text{NOS} + 1, \quad t = 1, 2, \ldots, H \tag{2.27}$$

$$\text{NOSp} = \frac{\text{NOS}}{H} 100 \tag{2.28}$$

2.1.8.4 Periods in Stock

NOS does not identify systematic errors because it does not consider the temporal dimension of stock carryover. Periods in Stock (PIS) goes a step further and measures the total number of periods the forecast items have spent in stock or the number of out-of-stock periods.

To understand how PIS works, suppose we forecast one unit per day over a three-day horizon. At the start of the first period, one item goes into a fictitious stock (this is a simplification compared to reality). If there was no demand during the first day, the result is plus one PIS. When there is demand, the demand is subtracted from the forecast. A demand of one in period 1 results in zero PIS in period 1. If the demand is zero in all three periods, the PIS in period 3 is plus six. The item from day one was in stock for three days, the item from day two was in stock for two days, and the last item was in stock for one day.

A positive number indicates that the forecasting method tends to overestimate demand. A negative number is a sign that demand is underestimated.

$$\text{PIS}_t = \text{PIS}_{t-1} + \sum_{i=1}^{t} (\hat{y}_t - y_{T+t}) = \sum_{i=1}^{t} (\hat{y}_i - y_{T+i}) (t + 1 - i) \tag{2.29}$$

$$\text{PIS}_t = \text{PIS}_{t-1} - \text{CFE}_t = -\sum_{i=1}^{t} \text{CFE}_i \tag{2.30}$$

$$\text{PIS} = \text{PIS}_H = -\sum_{t=1}^{H} \text{CFE}_t \tag{2.31}$$

In order to make the measure scale independent, it is scaled by $\frac{1}{T}\sum_{t=1}^{T} y_t$. So, the scaled Periods in Stock (sPIS) is given by:

$$\text{sPIS} = \frac{\text{PIS}}{\frac{1}{T}\sum_{t=1}^{T} y_t} \tag{2.32}$$

The sPIS acts complementarily to the sCE by providing evidence of systematic behavior with respect to the direction of the forecast error. The absolute value of sPIS is also considered as a measure of the magnitude of the bias. The scaled Absolute Periods in Stock (sAPIS) for series i are calculated by:

$$\text{sAPIS} = \frac{|\text{PIS}|}{\frac{1}{T}\sum_{t=1}^{T} y_t} \tag{2.33}$$

2.1.8.5 Stock-keeping-oriented Prediction Error Costs

Stock-keeping-oriented Prediction Error Costs (SPEC) is a newer metric [17], which tries to take a similar approach as Periods in Stock (PIS), but in a more slightly sophisticated way. It is defined in the following way:

$$SPEC_{\alpha_1,\alpha_2} = \frac{1}{T}\sum_{t=1}^{T}\sum_{i=1}^{t}\left(\max\left[0; \min\left[y_i; \sum_{k=1}^{i} y_k - \sum_{j=1}^{t}\hat{y}_j\right]\cdot\alpha_1; \min\left[\hat{y}_i; \sum_{k=1}^{i}\hat{y}_k - \sum_{j=1}^{t} y_j\right]\cdot\alpha_2\right]\cdot(t-i+1)\right)$$

Although it looks complicated at first, we can understand it intuitively. Two minimum internal conditions deal with the heart of the calculation - Opportunity cost and Stock Keeping costs. We need to balance these two costs in the supply chain from an inventory management perspective. The two parameters, α_1 and α_2, allow us to apply different weights to opportunity cost and storage cost. Depending on the organization's strategy, we can choose the appropriate type of weights. It is recommended that the summation of the weights is equal to 1 and $\alpha_1 = 0.75$ and $\alpha_2 = 0.25$ is a common choice in retail. The disadvantage of this is time complexity. We need nested loops to calculate this metric, which makes it slow to compute.

2.1.9 Probabilistic Forecasts Accuracy Metrics

This section discusses the quantile loss metric for evaluating probabilistic forecasts. It applies to models that generate probabilistic forecasts, i.e., prediction intervals, in contrast to point forecasts, which consist only of predicted values. By calculating the prediction quantiles, the model shows how much uncertainty is associated with each forecast and provides further information for the user. Without predictive quantiles, point forecasts have limited value for different decisions. The quantile loss calculates how far the forecast is from actual demand in either direction as a percentage of demand on average in each quantile. This metric helps capture the inherent bias in each quantile. For a given quantile $\rho \in (0, 1)$, the ρ-quantile loss is defined as:

$$\mathrm{QL}_\rho = 2\frac{\sum_{t=1}^{H} P_\rho\left(y_{T+t}, \hat{y}_t\right)}{\sum_{t=1}^{H}\left|y_{T+t}\right|}, \quad P_\rho(y,\hat{y}) = \begin{cases} \rho(y-\hat{y}) & \text{if } y > \hat{y} \\ (1-\rho)(\hat{y}-y) & \text{otherwise} \end{cases} \tag{2.34}$$

Lower values indicate better overall forecast accuracy. Generating forecasts for different quantiles is particularly useful when the costs of underestimating and overestimating are different.

For a p10 forecast, the actual value is expected to be lower than the predicted value 10% of the time. For a use case where there is not much storage space and the cost of capital invested is high or the price of overstocked goods is a concern, the p10 quantile forecast is useful for ordering a relatively small number of stocks. A

p10 forecast overestimates demand for an item only 10% of the time, meaning that it will be sold out about 90% of the time.

For a p50 forecast (often referred to as the median), the actual value is expected to be lower than the predicted value 50% of the time. When excess inventory is not too worrisome and demand for an item is moderate, the p50 quantile forecast can be useful.

For a p90 forecast, the actual value is expected to be lower than the predicted value 90% of the time. When not stocking an item will result in a large amount of lost revenue, the cost of not selling the item is too high, or the cost of capital invested is low. The p90 forecast may be useful for ordering a relatively large number of stocks.

2.2 State-Space Model

Two approaches are often found in the literature used in time series forecasting, and these are the use of filters and statistical models. Filter-based methods assume that observations are used as inputs to a system of equations. The classic example of a filter is single exponential smoothing (SES, 2.1.5.1), which has the form:

$$\hat{y}_t = \alpha y_{t-1} + (1-\alpha)\hat{y}_{t-1} \tag{2.35}$$

The advantage of filters is their simplicity and a low number of assumptions. For example, SES is straightforward to interpret and can be used without assumptions about the residuals' distributions. The main disadvantage is the lack of a statistical basis, which leads to difficulties in estimating parameters and constructing confidence intervals.

A statistical model is a mathematical representation of observed data using statistical analysis. They have their advantages as well as disadvantages. For example, statistical models usually have strict assumptions about the error component, which means choosing the right model requires certain assumptions. The advantages of having a statistical model are the simplified model selection procedure, the statistical estimation of parameters, and prediction ranges. In general, these models allow working with distributions of variables. Filters, in this case, focus on point values. Finally, it is known that there is no such thing as a true model, but even an incorrect one can be useful.

2.2.1 Latent variable models

One significant problem in machine learning is the unsupervised learning of a complex probability distribution $p(y)$ given a finite sample of observations y drawn from that distribution. So instead of modeling this distribution directly, we can

introduce an unobserved latent variable l and define a conditional probability $p(y \mid l)$ on the data. Each observed variable y will depend on a latent variable l, which we can think of as containing some simpler information that makes it easier to describe the observations. The desired distribution of the data is then obtained by marginalizing after the latent variable as follows:

$$p(y) = \int p(y, l)\mathrm{d}l = \int p(y \mid l)p(l)\mathrm{d}l \tag{2.36}$$

Thus, by introducing a latent variable into the model, we can express the complex marginal distribution $p(y)$ in terms of a more tractable joint distribution, whose components $p(y \mid l)$ and $p(l)$ are usually simpler to define.

Latent variable models can be used as black-box density models. However, we can also incorporate some prior knowledge of the generative mechanism that produced the data with distributions that define the joint distribution $p(y, l)$, e.g., using probabilistic graphical models.

To infer the latent variable from observations as a posterior distribution $p(l \mid y)$, we can use Bayes' rule:

$$p(l \mid y) = \frac{p(y \mid l)p(l)}{p(y)} \tag{2.37}$$

However, the posterior is intractable in many cases due to the lack of analytical solution of the integral in Equation 2.37, which appears in the denominator.

2.2.2 State-Space Models

State-space models (SSMs) are a general and flexible methodology for sequential data modeling. SSMs are particularly well suited for applications where the time series structure is well known because they allow these assumptions to be incorporated into the model. This allows the model to be interpreted, and the learning procedure is data-efficient but requires time series with a suitable history. However, in modern forecasting applications with a large body of time series, this requires high computing power. Also, traditional SSMs cannot infer common patterns for similar time series, as they are selected for each time series. This makes it difficult to make forecasts for time series with little or no history. Hyndman et al. [11], Durbin and Koopman [5], and Murphy [18] presented a comprehensive overview of state-space models.

In an SSM, we are given a sequence of T observations $y_{1:T} = [y_1, \ldots, y_T]$, that possibly depends on some inputs $x_{1:T} = [x_1, \ldots, x_T]$, and we are interested in modelling the distribution $p(y_{1:T} \mid x_{1:T})$. It is a very general formulation that can be used in various applications. SSMs model the temporal structure of the data via a latent state l_t at each time step that summarizes all the information coming from the past and determines the system's present and future evolution. They can be

used to encode time series components such as level, trend and seasonality patterns. SSMs can then be seen as a temporal extension of the latent variable models, in which the prior over the latent variables l_t at each time step varies over time as it depends on the previous state l_{t-1} and possibly some inputs x_t to the model. The model can be written in the following generic form:

$$\begin{aligned} l_t &= g\left(x_t, l_{t-1}, \epsilon_t\right) \\ y_t &= h\left(l_t, x_t, \delta_t\right) \end{aligned} \tag{2.38}$$

where l_t is the hidden state, x_t is an optional input or control signal, y_t is the observation, g is the transition model, h is the observation model, ϵ_t is the system noise at time t, and δ_t is the observation noise at time t. Graphical representation of the SSM is given in Figure 2.3.

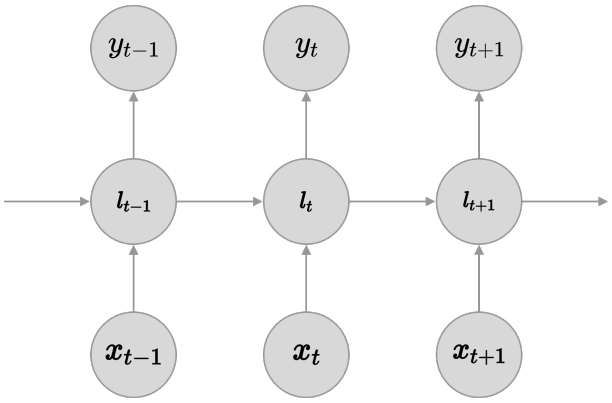

Figure 2.3: Graphical representation of a SSM.

Based on the assumptions of SSMs, we can obtain the following nice properties of them:

- $p\left(y_t \mid l_{1:t}, y_{1:t-1}, x_{1:t}\right) = p\left(y_t \mid l_t\right)$
  The latent state fully determines the observation at time t, so it does not depend on the model's past states, inputs, and outputs.

- $p\left(l_t \mid l_{1:t-1}, y_{1:t-1}, x_{1:t}\right) = p\left(l_t \mid l_{t-1}, x_t\right)$
  Conditioned on l_{t-1}, the current state l_t does not depend on the previous states $l_{1:t-2}$, nor the past inputs or outputs.

- $p\left(l_t \mid l_{t+1:T}, y_{t+1:T}, x_{t+1:T}\right) = p\left(l_t \mid l_{t+1}\right)$
  Given the next state l_{t+1}, l_t does not depend on the future states, inputs and outputs.

The emission distribution $p(y_t \mid z_t)$ specifies how observation y_t depends on the latent state l_t. Furthermore, $p(l_t \mid l_{t-1}, x_t)$ is called the transition distribution and represents the prior distribution for the state at each time step given the previous state and the current input to the model. This distribution fully determines the

temporal evolution of the system. Then the marginal distribution in the observations can be obtained by integrating the states:

$$p\left(y_{1:T} \mid x_{1:T}\right) = \int p\left(y_{1:T}, l_{1:T} \mid x_{1:T}\right) \, dl_{1:T} \tag{2.39}$$

$$\begin{aligned} p\left(y_{1:T}, l_{1:T} \mid x_{1:T}\right) &= p\left(y_{1:T} \mid l_{1:T}\right) p\left(l_{1:T} \mid x_{1:T}\right) \\ &= \prod_{t=1}^{T} p\left(y_t \mid l_t\right) \cdot p\left(l_1\right) \prod_{t=2}^{T} p\left(l_t \mid l_{t-1}, x_t\right) \end{aligned} \tag{2.40}$$

2.2.2.1 Posterior inference in the sequential setting

The inference problem consists of calculating the posterior probability of latent states for a given output sequence. This computation can be done both forward and backward. We can distinguish the following inference tasks:

- **Filtering.** We want to compute $p\left(l_t \mid y_{1:t}, x_{1:t}\right)$. This is interesting in an online setting as it allows the state to be estimated on an ongoing basis.

- **Smoothing.** We are interested in the posterior $p\left(l_t \mid y_{1:T}, x_{1:T}\right)$. Since the smoothed posterior requires the whole sequence's knowledge, it can be computed only offline.

- **Prediction.** We can also be interested in predicting the state of the system h steps in the future given only past information by computing $p\left(l_{t+h} \mid y_{1:t}, x_{1:t+h}\right)$.

2.2.2.2 Parameter learning

In some cases, the model parameters are unknown, and we have to learn them from the data. We can perform inference in the model for known parameters θ, but this time we turn to estimate these parameters. Unfortunately, there is no maximum likelihood estimation (MLE) of closed-form, so we turn to the EM algorithm.

The expectation-maximization (EM) algorithm is an iterative method for finding maximum likelihood or maximum a posteriori (MAP) estimates of a statistical model's parameters. The approach is based on the fact that the model depends on unobservable variables. The EM algorithm alternates between an expectation step (E), which provides the expectation of the conditional logarithm of the reliability estimated using the current parameter estimate, and a maximization step (M), which computes the parameters to maximize the expected found log-likelihood at step E. These parameter estimates are then used to determine the latent variables' distribution in the next step E.

2.2.3 General Intermittent State-Space Model

Svetunkov and Boylan [31] proposed a general state-space model for intermittent time series. It is derived from Croston's original model with the division of non-regulatory demand y_t into two components o_t and z_t. Only z_t has its own statistical model. The component z_t corresponds to the demand of potential customers for the product, while o_t corresponds to the realisation of demand when the customer makes a purchase. In their work, they adopted the single source error (SSOE) state-space model, where the same error term is used in the transition and measurement equations because it is well established. The following system of equations defines this model:

$$y_t = o_t z_t \tag{2.41}$$

$$z_t = w\left(l_{t-1}\right) + r\left(l_{t-1}\right)\epsilon_t \tag{2.42}$$

$$l_t = f\left(l_{t-1}\right) + g\left(l_{t-1}\right)\epsilon_t \tag{2.43}$$

where o_t is a Bernoulli distributed random variable, l_t is the state vector, ϵ_t is the error term, $f(\cdot)$ is the transition function, $w(\cdot)$ is the measurement function, $g(\cdot)$ is the persistence function and $r(\cdot)$ is the error term function.

The first equation corresponds to the original formula from the Croston model, while the second, called the measurement equation, reflects the potential evolution of demand over time. The third equation is the standard transitions for the SSOE model, describing the change of state over time. The general intermittent state-space model's interpretation is that the potential magnitude of demand can change over time, even if actual demand is not observed. In this case $o_t = 0$, which leads to $y_t = 0$ in the first equation. However, measurement (2.42) and transition (2.43) equations are not affected by o_t, leading to the evolution of z_t regardless of whether there is actual demand.

2.3 Artificial neural networks

Artificial neural networks (ANN) or simpler Neural Networks (NN) are models used for pattern recognition, inspired by the human brain. A large part of machine learning research has been devoted in recent times. Due to their popularity in an application, a separate branch of machine learning called Deep Learning has been dedicated.

While neural networks themselves are complex mathematical functions with unique properties and assumptions, an analogy to the human brain anatomy can be used to imagine how they work. In this case, neural networks consist of interconnected nodes that resemble biological neurons in the human brain, and the connections resemble axons. In this case, when a biological neuron receives input signals via axons, it produces an output signal. The nodes in neural networks work

on the same principle. In the rest of this chapter, nodes in artificial neural networks will be referenced as neurons.

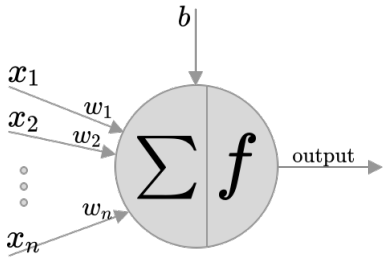

Figure 2.4: Visualization of a single neural network neuron.

The operation of a single neuron with input signals $\{x_1, x_2, \ldots, x_n\}$, weights $\{w_1, w_2, \ldots, w_n\}$, bias b and activation function f is visualized in the Figure 2.4. This represents the following equation:

$$\text{output} = f\left(\sum_{i=1}^{n} x_i w_i + b\right)$$

While weights and bias values are determined by a learning process described in a later section, the activation function must be defined in advance. Moreover, to match non-linear patterns in the data, and since assembling multiple linear functions is a linear function, the activation function is usually a non-linear transformation.

Many candidates for activation functions are known, and new proposals are still being investigated. However, the final choice depends on the problem we need to solve and the required properties. The only requirement for activation functions is that they must be differentiated so that optimization and learning are possible. Examples of activation functions:

$$\text{Sigmoid}: \qquad \sigma(x) = \frac{1}{1+e^{-x}} \tag{2.44}$$

$$\text{Hyperbolic tangent}: \qquad \tanh(x) = \frac{e^{2x}-1}{e^{2x}+1} \tag{2.45}$$

$$\text{ReLU}: \qquad \text{ReLU}(x) = \max(0, x) \tag{2.46}$$

Then, such neurons, which are the fundamental component of neural networks, are combined and stacked, allowing various types of networks, often referred to as network architectures. The next sections explain the following types of networks:

- Feed-forward neural network
- Recurrent neural network
- LSTM

2.3.1 Feed-forward neural network

Feed-Forward Neural Network (FFNN) is one of the simplest types of a neural network, where data is just transferred between neurons. We distinguish three components in such a network: the input layer, hidden layers, and the last layer called the output layer. Despite its simplicity, FFNN is a powerful model. The example is presented in the Figure 2.5.

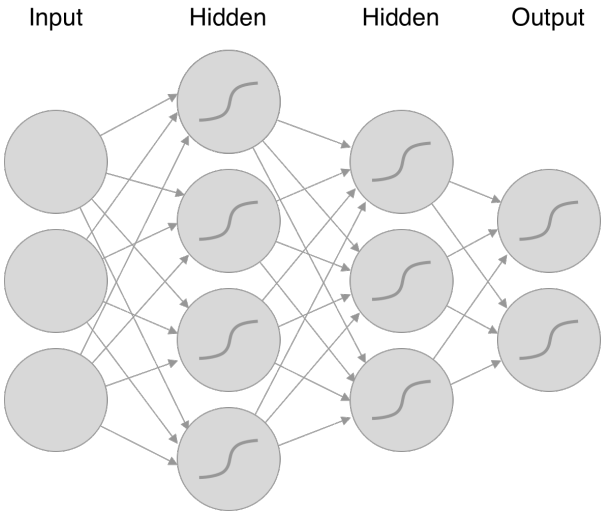

Figure 2.5: Visualization of a 3-layer feed-forward neural network.

Hidden layers number means the network's depth. The deeper the network is, the greater its ability to learn abstract combinations of features from data due to its complexity compared to shallow networks that may not have the necessary capacity. However, greater depth brings with it additional problems. One of the problems is the risk of over-fitting when the network begins to model the noise in the training dataset. In this way, the ability to generalize is lost. It is usually a compromise between the amount of training data and the difficulty of the problem which is tried to solve.

Denoting $\mathcal{W}$ as the set of network parameters (every neurons' weights and biases), the network function f calculating the output for a given input is uniquely defined as

$$\hat{\mathbf{y}} = f(\mathbf{x} \mid \mathcal{W}), \tag{2.47}$$

i.e. the parameters define the network. The loss function for the network for which the weights should be optimal must be defined to find the optimal parameters for a given neural network and training dataset.

2.3.2 Recurrent neural network

The FFNN presented in the previous subsection 2.3.1 is a powerful model, but it has its restrictions. One of them is that it is a memoryless model, whats means

that for each input, the network does not have information about the previous ones. However, this property can be achieved with the usage of recurrent neural networks (RNNs).

RNN's are identical to FFNN's with the difference that each layer transfers the output to the next layer and itself, hence the recurring connections. It is visualized on the Figure 2.6.

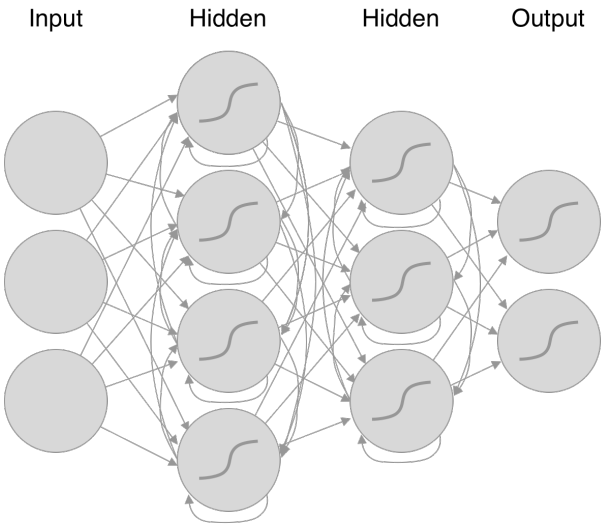

Figure 2.6: Visualization of a 3-layer recurrent neural network.

A popular way of visualizing the RNN's is by network unfolding, thus clearly showing the repeated connections between each step t. An example of an unfolded network is presented at the Figure 2.7.

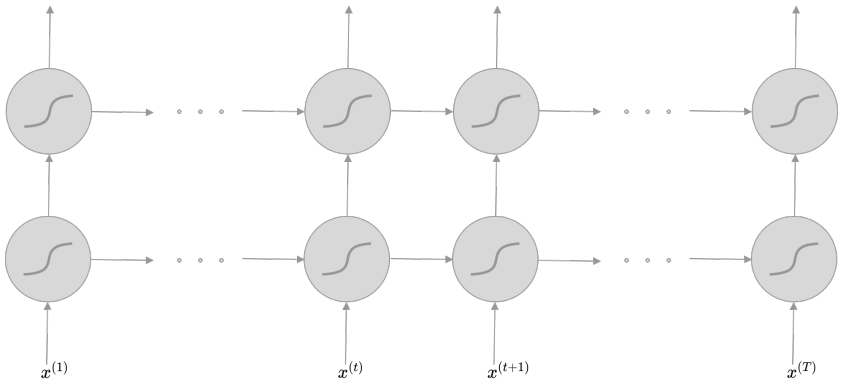

Figure 2.7: Visualization of an unfolded 3-layer recurrent neural network.

Although this recurrent time-dependency of RNN's makes them capable of learning long-term dependency patterns and even using the information of any input value that has ever occurred in theory, it turns out that there are problems with this practice. How to solve them will be discussed in more detail in the next section.

2.3.3 Long short-term memory

As mentioned in the previous section, the standard architecture of the RNN has a particular problem with handling recurring information on the network. It has turned out that in practice, learning long-term dependencies is a real difficulty for ordinary RNN. Besides, the fact that the information is fed back into its hidden layer at every step of the time causes the output to fade or explode exponentially. This is called an exploding gradient problem and a fading gradient problem. A similar problem also exists with very depth neural networks.

A famous architecture called "Long Short-Term Memory" (LSTM) was created to deal with this problem. LSTM can handle long-term dependencies and retain gradient information over time, better than ordinary RNN architecture, making it the preferred RNN architect over the basic version. These properties have been obtained by extending the LSTM of ordinary neurons so that they contain many operations. It makes sense to call every neuron in the RNN a block. A simple illustration of a regular RNN block is shown in the Figure 2.8.

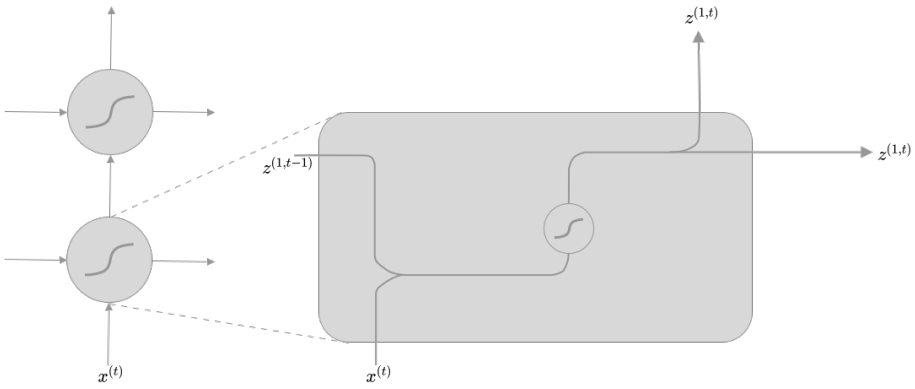

Figure 2.8: A regular RNN block.

LSTM expands the simple RNN's block shown in figure 2.8 in several ways. First, the cell state is introduced, which can contain information through time if necessary. This can be seen as the memory of the LSTM block. Secondly, several different gateways are inserted, each of which takes combined input and previous output.

- The **Forget gate:** Determine what information should be forgotten when performing an elementwise multiplication between data and the cell's previous state.
- The **Input gate:** Determines what new information from cell values should be added to the cell state by performing an elementwise addition between the updated cell state and modified cell values, resulting in a new cell state.
- The **Output gate:** It determines what values should be derived by performing an elementwise multiplication between the non-linear transformation of the

new cell state and the gateway output values, leading to the final result.

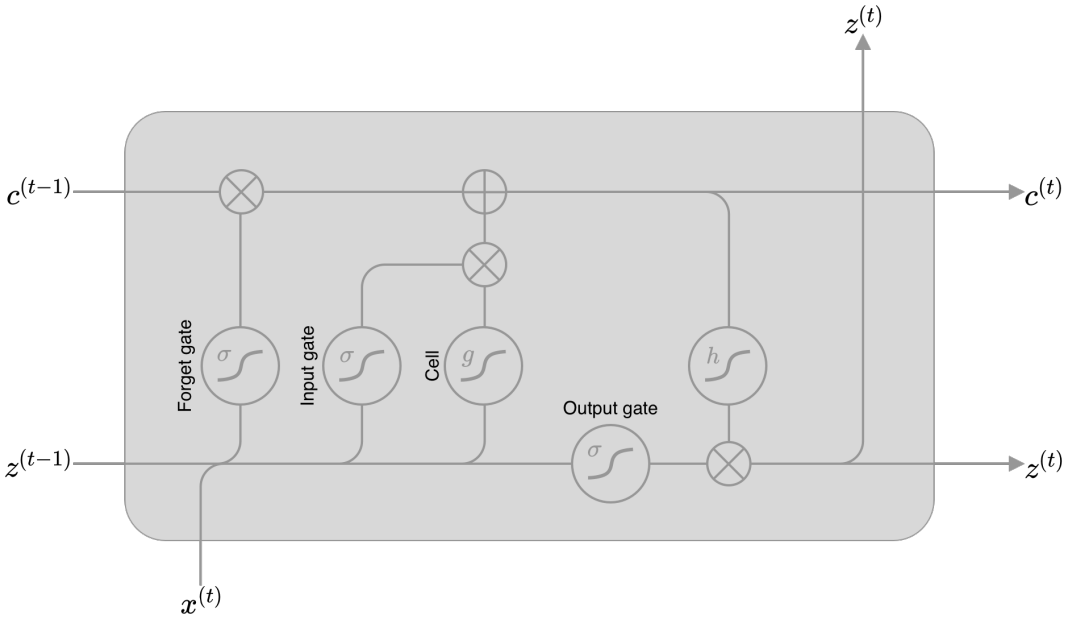

Figure 2.9: LSTM block.

The LSTM block's visualisation is shown in the figure 2.9, and the full set of equations is presented.

$$\text{Forget gate :} \quad \mathbf{f}^{(t)} = \sigma\left(\mathrm{W}_f\left[\mathbf{z}^{(t-1)}, \mathbf{x}^{(t)}\right] + \mathbf{b}_f\right) \tag{2.48}$$

$$\text{Input gate :} \quad \mathbf{i}^{(t)} = \sigma\left(\mathrm{W}_i\left[\mathbf{z}^{(t-1)}, \mathbf{x}^{(t)}\right] + \mathbf{b}_i\right) \tag{2.49}$$

$$\tilde{\mathbf{c}}^{(t)} = g\left(\mathrm{W}_c\left[\mathbf{z}^{(t-1)}, \mathbf{x}^{(t)}\right] + \mathbf{b}_c\right) \tag{2.50}$$

$$\text{State update :} \quad \mathbf{c}^{(t)} = \mathbf{f}^{(t)} \otimes \mathbf{c}^{(t-1)} + \mathbf{i}^{(t)} \otimes \mathbf{c}^{(t)} \tag{2.51}$$

$$\text{Output gate :} \quad \mathbf{o}^{(t)} = \sigma\left(\mathrm{W}_o\left[\mathbf{z}^{(t-1)}, \mathbf{x}^{(t)}\right] + \mathbf{b}_o\right) \tag{2.52}$$

$$\mathbf{z}^{(t)} = \mathbf{o}^{(t)} \otimes h\left(\mathbf{c}^{(t)}\right) \tag{2.53}$$

2.3.4 Optimization

Training neural networks consist of minimizing the network target's function (loss function) related to network parameters. This optimization is usually performed using gradient-based optimization algorithms such as Gradient Descent. In this project, another gradient-based algorithm, called the Adam algorithm, is used.

2.3.5 The Promise of Deep Learning for Time Series

Deep learning neural networks can automatically learn any complex mapping from input to output and handle multiple inputs and outputs. These powerful

functions promise time series forecasting, especially when there are problems with complex non-linear relationships, multi-value input data, and multi-stage forecasting. Together with the capabilities of more modern neural networks, these features can be up-and-coming, such as the automatic learning of features and native support for sequential data in recurrent neural networks.

Furthermore, traditional forecasting methods cannot infer shared patterns from a set of similar time series, as they are determined separately for each time series. With deep learning, model parameters are shared between and learned jointly from all time series in the dataset. This enables knowledge sharing, for example, by considering the impact of special events such as holidays on demand size and allowing forecasts to apply for time series with limited historical data or high levels of noise.

Thanks to this ability to learn long-term correlations in a sequence, LSTM networks eliminate the need for a predefined time window and accurately model complex multidimensional sequences. They can do so based on a set of raw time series data with little human effort. However, as these models have fewer structural assumptions, they usually require larger training data sets to learn accurate models. Unlike SSM, these models are challenging to interpret and, more importantly, they make it challenging to enforce assumptions such as time smoothness.

Chapter 3

Methods

This thesis proposes a method for intermittent time series forecasting that involves parameterizing the intermittent state-space model (iETS, 2.1.6.3) using recurrent neural networks (RNN). The result of this combination is named as **Deep iETS**. RNN parameters are learned together from the set of time series and related feature variables. This approach allows automatic modeling, feature extraction, and recognition of intricate temporal patterns present in the data. Furthermore, as each time series is ultimately modeled using iETS, the method allows for interpretability. The parameters for each time series can be controlled and adjusted to the required properties if necessary.

The problem statement is the following. A set of N non-negative real-valued processes in discrete time denoted $\left\{y_{1:T_i}^{(i)}\right\}_{i=1}^{N}$ is considered, where

$$y_{1:T_i}^{(i)} = \left(y_1^{(i)}, y_2^{(i)}, \ldots, y_{T_i}^{(i)}\right)$$

and $y_t^{(i)}$ denotes the value at time t of the i-th time series. For each time series, the same forecast horizon $h \in \mathrm{N}_{>0}$ is investigated, then $y_{1:T_i}^{(i)}$ is referred to as target time series, with time range $\{1, 2, \ldots, T_i\}$ as the training range, and $\{T_i + 1, T_i + 2, \ldots, T_i + h\}$ as the prediction range. Moreover, the set of associated, time-varying feature vectors $\left\{x_{1:T_i+h}^{(i)}\right\}_{i=1}^{N}$ is delivered, emphasising that these are also provided for the prediction range.

Finally, the aim is to model the probability distribution for each time series $i = 1, \ldots, N$ of future values $y_{T_i+1:T_i+h}^{(i)}$ given the past. Formally, this is formulated as follows:

$$p\left(y_{T_i+1:T_i+h}^{(i)} \mid y_{1:T_i}^{(i)}, x_{1:T_i+h}^{(i)}; \Phi\right) \tag{3.1}$$

where Φ denotes the set of model parameters shared between and learned jointly from all N time series. It is in contrast to many traditional approaches (Croston and others) that are local, meaning that each time series has its own model, and no information is shared across time series.

3.1 Intermittent State Space Model

An adjusted version of the multiplicative state-space model iETS(M,N,N) proposed by Svetunkov and Boylan [31] is considered. Intermittent state-space model (iETS) is formulated with the following system of equations:

$$y_t = o_t z_t \tag{3.2}$$
$$z_t = l_{z,t-1}\left(1 + \epsilon_t\right) \tag{3.3}$$
$$l_{z,t} = l_{z,t-1}\left(1 + \alpha_{z,t}\epsilon_t\right) \tag{3.4}$$
$$o_t \sim \text{ Bernoulli } \left(p_{o,t}\right) \tag{3.5}$$
$$\left(1 + \epsilon_t\right) \sim \log \mathcal{N}\left(0, \sigma^2_{\epsilon,t}\right) \tag{3.6}$$

Here at time t, the o_t models the demand occurrence with probability $p_{o,t}$, while z_t models the demand size, with the latent state $l_{z,t-1}$, smooth parameter $\alpha_{z,t}$, and variance of innovation $\sigma^2_{\epsilon,t}$. The model's interpretation is that the potential size of demand can change over time, even if actual demand is not observed. Furthermore, the multiplicative model for the measurement equation has the advantage of restricting the space of demand sizes to positive values. According to the authors' analysis, it is assumed that the demand size smoothing parameter, because it guarantees that the level of time series is always positive whatever the error value is $\alpha_{z,t} \in [0, 1]$. For boundary conditions, when $\alpha_{z,t} = 0$, the level is not updated, while when $\alpha_{z,t} = 1$, the level has the dynamics of a random walk. The difference from the original version of the model is that the smoothing parameters and the variance have become time-variant, so they are no longer static as before. The probabilities $p_{o,t}$ are not modeled but are model parameters.

3.1.1 Model Properties

This section presents the properties of the considered model together with the distributions of the individual variables, derived from the original iETS model [31]. This multiplicative model is also well studied by Akram et al. [1].

Since according to the model's assumption $o_t \sim$ Bernoulli $(p_{o,t})$, where $p_{o,t}$ is delivered as the parameter, the conditional expectation and the conditional variance can be easily calculated as:

$$\mathbb{E}\left(o_{t+h|t}\right) = p_{o,t+h} \tag{3.7}$$
$$\mathbb{V}\left(o_{t+h|t}\right) = p_{o,t+h}\left(1 - p_{o,t+h}\right) \tag{3.8}$$

An expectation conditional on the last non-zero demand should be used for zero demand periods when the model's demand realization is not observable ($o_t = 0$). This means that the equation of transition at some observation $t+k$ is defined based

on previous k values:

$$l_{z,t+k|t} = l_{z,t} \prod_{j=1}^{k} (1 + \alpha_{z,t+j}\epsilon_{t+j}) = l_{z,t+k-1|t} (1 + \alpha_{z,t+k}\epsilon_{t+k}) \tag{3.9}$$

It can be shown that $(1 + \alpha_{z,t}\epsilon_t)$ has a three-parameter log normal distribution [26]. Firstly, we can regroup it as follows:

$$1 + \alpha_{z,t}\epsilon_t = (1 - \alpha_{z,t}) + \alpha_{z,t} (1 + \epsilon_t) \tag{3.10}$$

where $(1 + \epsilon_t) \sim \log \mathcal{N}\left(0, \sigma^2_{\epsilon,t}\right)$. Hence, we obtain that

$$1 + \alpha_{z,t}\epsilon_t \sim 3\,\text{Plog}\,\mathcal{N}\left(\log \alpha_{z,t}, \sigma^2_{\epsilon}, 1 - \alpha_{z,t}\right) \tag{3.11}$$

Then, the conditional expectation and conditional variance can be calculated as:

$$\mathbb{E}\,(1 + \alpha_{z,t}\epsilon_t) = (1 - \alpha_{z,t}) + \alpha_{z,t} \exp\left(\frac{\sigma^2_{\epsilon}}{2}\right) \tag{3.12}$$

$$\mathbb{V}\,(1 + \alpha_{z,t}\epsilon_t) = \alpha^2_{z,t} \left(\exp\left(\sigma^2_{\epsilon}\right) - 1\right) \exp\left(\sigma^2_{\epsilon}\right) \tag{3.13}$$

The conditional expectation, conditional variance, and conditional median of the level $l_{z,t+k}$ given the value of $l_{z,t}$, defined by (3.9), and assuming that the error term is not autocorrelated, are as follows:

$$\mathbb{E}\left(l_{z,t+k|t}\right) = l_{z,t} \prod_{j=1}^{k} \mathbb{E}\,(1 + \alpha_{z,t+j}\epsilon_{t+j}) \tag{3.14}$$

$$= \mathbb{E}\left(l_{z,t+k-1|t}\right) \mathbb{E}\,(1 + \alpha_{z,t+k}\epsilon_{t+k}) \tag{3.15}$$

$$\begin{aligned}\mathbb{V}\left(l_{z,t+k|t}\right) = l^2_{z,t} \Bigg(& \prod_{j=1}^{k} \left(\mathbb{V}\,(1 + \alpha_{t+j}\epsilon_{t+j}) + \mathbb{E}\,(1 + \alpha_{t+j}\epsilon_{t+j})^2\right) \\ & - \prod_{j=1}^{k} \mathbb{E}\,(1 + \alpha_{t+j}\epsilon_{t+j})^2 \Bigg)\end{aligned} \tag{3.16}$$

$$\begin{aligned}= {} & \mathbb{V}\left(l_{z,t+k-1|t}\right) \mathbb{V}\,(1 + \alpha_{z,t+k}\epsilon_{t+k}) \\ & + \mathbb{E}\left(l_{z,t+k-1|t}\right)^2 \mathbb{V}\,(1 + \alpha_{z,t+k}\epsilon_{t+k}) \\ & + \mathbb{V}\left(l_{z,t+k-1|t}\right) \mathbb{E}\,(1 + \alpha_{z,t+k}\epsilon_{t+k})^2\end{aligned} \tag{3.17}$$

$$\text{Md}\left(l_{z,t+k|t}\right) = l_{z,t} \tag{3.18}$$

where $\mathbb{E}\,(1 + \alpha_{z,t+k}\epsilon_{t+k})$ and $\mathbb{V}\,(1 + \alpha_{z,t+k}\epsilon_{t+k})$ are respectively defined by (3.12) and (3.13).

In order to separate the level from the uncertainty coming from the error term, the transition of states in model with the log-normal distribution, when the demand is not observed ($o_t = 0$), should be governed by the median (3.18) instead of the mean (3.14). This is because the error component's distribution is skewed, and the conditional mean absorbs some of the uncertainty due to the error.

The error term $(1+\epsilon_t) \sim \log \mathcal{N}\left(0, \sigma^2_{\epsilon,t}\right)$, so the expectation and variance are as follows:

$$\mathbb{E}\left(1+\epsilon_t\right) = \exp\left(\frac{\sigma^2_{\epsilon,t}}{2}\right) \tag{3.19}$$

$$\mathbb{V}\left(1+\epsilon_t\right) = \left(\exp\left(\sigma^2_{\epsilon,t}\right) - 1\right)\exp\left(\sigma^2_{\epsilon}\right) \tag{3.20}$$

Prediction intervals of h steps ahead for z_{t+h} should be derived separately. Rewriting z_{t+h} in terms of $l_{z,t}$ and error terms, we obtain

$$z_{t+h|t} = l_{z,t}\left(1+\epsilon_{t+h}\right)\prod_{j=1}^{h-1}\left(1+\alpha_{t+j}\epsilon_{t+j}\right) \tag{3.21}$$

The conditional mean, variance, and median for h-steps ahead will be:

$$\mathbb{E}\left(z_{t+h|t}\right) = l_{z,t}\exp\left(\frac{\sigma^2_{\epsilon,t+h}}{2}\right)\prod_{j=1}^{h}\mathbb{E}\left(1+\alpha_{z,t+j}\epsilon_{t+j}\right) \tag{3.22}$$

$$\begin{aligned}\mathbb{V}\left(z_{t+h|t}\right) = {} & \left(\exp\left(\sigma^2_{\epsilon,t+h}\right) - 1\right)\exp\left(\sigma^2_{\epsilon,t+h}\right)\mathbb{E}\left(l_{z,t+h-1|t}\right)^2 \\ & + \exp\left(2\sigma^2_{\epsilon,t+h}\right)\mathbb{V}\left(l_{z,t+h-1|t}\right)\end{aligned} \tag{3.23}$$

$$\mathrm{Md}\left(z_{t+h|t}\right) = l_{z,t} \tag{3.24}$$

While $l_{t+h|t}$ and $z_{t+h|t}$ no longer have a log-normal distribution, we use it as a good approximation of the actual conditional distribution.

The model is based on the one-step-ahead error, but can generate the h steps-ahead conditional mean and variance. The considered intermittent state space model, by assuming the conditional independence of the variables $o_{t+h|t}$ and $z_{t+h|t}$, allows these values to be calculated using the following formulas:

$$\mathbb{E}\left(y_{t+h|t}\right) = \mathbb{E}\left(o_{t+h|t}\right)\mathbb{E}\left(z_{t+h|t}\right) \tag{3.25}$$

$$\mathbb{V}\left(y_{t+h|t}\right) = \mathbb{V}\left(o_{t+h|t}\right)\mathbb{V}\left(z_{t+h|t}\right) + \mathbb{V}\left(o_{t+h|t}\right)\mathbb{E}\left(z_{t+h|t}\right)^2 + \mathbb{E}\left(o_{t+h|t}\right)^2\mathbb{V}\left(z_{t+h|t}\right) \tag{3.26}$$

where $\mathbb{E}\left(y_{t+h|t}\right)$ and $\mathbb{V}\left(y_{t+h|t}\right)$ are respectively conditional expectation and conditional variance of y_t; $\mathbb{E}\left(o_{t+h|t}\right)$ and $\mathbb{V}\left(o_{t+h|t}\right)$ are conditional expectation and variance of occurrence variable o_t and finally $\mathbb{E}\left(z_{t+h|t}\right)$ and $\mathbb{V}\left(z_{t+h|t}\right)$ are the respective values for the demand size z_t.

3.1.2 Parameter learning

This state space model is fully specified by the parameters $\Theta_t = \left(l_{z,0}, \alpha_{z,t}, \sigma^2_{\epsilon,t}, p_{o,t}\right)$, $\forall t > 0$. Besides, the following conditions are assumed for the parameters: $l_{z,0} \in \mathrm{R}_+$, $\alpha_{z,t} \in [0,1]$, $\sigma^2_{\epsilon,t} \in \mathrm{R}_+$ and $p_{o,t} \in [0,1]$. Because it is a statistical model, parameters can be estimated via likelihood maximisation, $\Theta^*_{1:T} = \mathrm{argmax}_{\Theta_{1:T}}\, p\left(y_{1:T} \mid \Theta_{1:T}\right)$.

For the intermittent demand model, there are two cases which need to be investigated: when demand occurs and when it does not. In the first case, the probability of obtaining the observation y_t is equal:

$$p\left(y_t, o_t = 1 \mid \Theta_{1:t}, y_{1:t-1}\right) = p_{o,t} \cdot p\left(z_t \mid \Theta_{1:t}, y_{1:t-1}\right) \tag{3.27}$$

In the second case, it is just equal to the probability of non-occurrence:

$$p\left(y_t, o_t = 0 \mid \Theta_{1:t}, y_{1:t-1}\right) = (1 - p_{o,t}) \tag{3.28}$$

In summary, the likelihood function for all T observations is, therefore, as follows:

$$p_{\text{iETS}}\left(y_{1:T} \mid \Theta_{1:T}\right) = \prod_{o_t=1} p_{o,t} \cdot p\left(z_t \mid \Theta_{1:t}, y_{1:t-1}\right) \prod_{o_t=0} (1 - p_{o,t}) \tag{3.29}$$

where $p\left(z_t \mid \Theta_{1:t}, y_{1:t-1}\right)$ is a log-normal distribution.

Since not all the variables' values are known, such as the latent states, it is impossible to optimize likelihood directly. For this purpose, the EM algorithm is used, where first the values of the unknown variables are estimated, and only then the likelihood value is maximized.

It is worth noting that when using classical methods for multiple time series, a separate set of parameters $\Theta^{(i)}$ is learned independently for each time series $y^{(i)}_{1:T_i}$. This approach's disadvantage is that no information is shared between time series, making it challenging to apply the method to time series with limited historical data or a high noise level.

3.2 Deep Intermittent State Space Model

Alternatively to learning the parameters $\Theta^{(i)}$ independently for each time series, this approach learns and uses a global shared mapping from the feature vectors $x^{(i)}_{1:T_i}$ associated with each target time series $y^{(i)}_{1:T_i}$ to the time-varying parameters $\Theta^{(i)}_t$ of the intermittent state-space model for the i-th time series. The mapping under consideration is defined as follows:

$$\Theta^{(i)}_t = \Psi\left(\mathbf{x}^{(i)}_{1:t}, \Phi\right), \quad i = 1, \ldots, N, \quad t = 1, \ldots, T_i + h \tag{3.30}$$

as a function of the feature vectors $x^{(i)}_{1:t}$ up to time t, as well as the shared parameters Φ. Then, based on the considered model, the $y^{(i)}_{1:T_i}$ values have a distribution defined as:

$$p\left(y^{(i)}_{1:T_i} \mid x^{(i)}_{1:T_i}, \Phi\right) = p_{\text{iETS}}\left(y^{(i)}_{1:T_i} \mid \Theta^{(i)}_{1:T_i}\right), \quad i = 1, \ldots, N \tag{3.31}$$

where p_{iETS} denotes the marginal likelihood under a intermittent state space model as defined in Equation 3.29, given its time-varying parameters $\Theta^{(i)}_t$.

A long short-term memory network (LSTM) is used to parameterize the mapping function Ψ from the feature vectors to the iETS parameters. The general

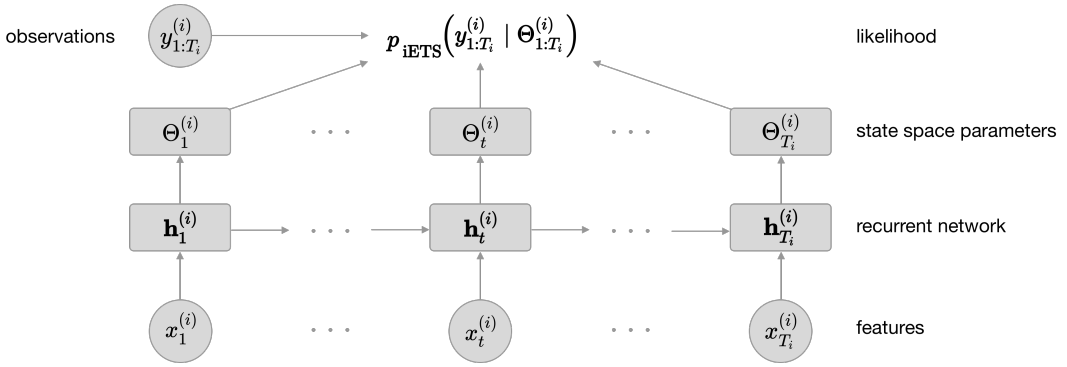

Figure 3.1: Summary of the Deep iETS model.

scheme of the model's operation unrolled for the training range has been presented on the Figure 3.1. To obtain the state-space parameters $\Theta_t^{(i)}$ at each time t in the training range $\{1, 2, \ldots, T_i\}$ for the target time series $y_{1:T_i}^{(i)}$, with associated features $x_{1:T_i}^{(i)}$ and network parameters Φ, the LSTM cell's output $\mathbf{o}_t$ is first computed, together with the hidden state $\mathbf{h}_t$ via a recurrent function as follows:

$$\left(\mathbf{o}_t^{(i)}, \mathbf{h}_t^{(i)}\right) = cell\left(\mathbf{h}_{t-1}^{(i)}, \mathbf{x}_t^{(i)}, \Phi\right) \tag{3.32}$$

Next, to constrain the real-valued output $\mathbf{o}_t$ to the parameters domain of the iETS, the following transformations are applied:

- for positive parameters $\theta > 0$: the softplus function $\theta = \log(1 + \exp(\cdot))$
- for bounded parameters $\theta \in [0, 1]$: a scaled and shifted sigmoid $\theta = \frac{1}{1+\frac{1}{\exp(\cdot)}}$

Parameters $\Theta_{1:T_i}^{(i)}$ are then used to compute the likelihood of the given observations $y_{1:T_i}^{(i)}$, which is used for optimizing of the network parameters Φ.

3.3 Training

The model parameters Φ are learned by maximizing the probability of observing the data $\left\{y_{1:T_i}^{(i)}\right\}_{i=1}^{N}$ in the training range, i.e., by maximizing the (log-)likelihood:

$$\mathcal{L}(\Phi) = \sum_{i=1}^{N} \log p\left(y_{1:T_i}^{(i)} \mid \mathbf{x}_{1:T_i}^{(i)}, \Phi\right) = \sum_{i=1}^{N} \log p_{\text{iETS}}\left(y_{1:T_i}^{(i)} \mid \Theta_{1:T_i}^{(i)}\right) = \tag{3.33}$$

$$= \sum_{i=1}^{N} \left[- \sum_{o_t^{(i)}=1} \left(\log\left(z_t^{(i)}\right) + \frac{1}{2}\log\left(2\pi\sigma_{\epsilon,t}^{2(i)}\right) + \frac{1}{2}\frac{\left(\log z_t^{(i)} - \log \mu_{z,t|t-1}^{(i)}\right)^2}{\sigma_{\epsilon,t}^{2(i)}} \right) \right.$$

$$\left. + \sum_{o_t^{(i)}=1} \log\left(p_{o,t}^{(i)}\right) + \sum_{o_t^{(i)}=0} \log\left(1 - p_{o,t}^{(i)}\right) \right] \tag{3.34}$$

where $\mu^{(i)}_{z,t|t-1} = E\left[z^{(i)}_t \mid y^{(i)}_{1:t-1}, \Theta^{(i)}_{1:t-1}\right]$ is the conditional mean of one-step-ahead forecast error for demand sizes. Each summand of $\mathcal{L}(\Phi)$ in the Equation 3.33 can be viewed as a (negative) loss function that measures the compatibility between the state-space model parameters produced by the LSTM with input $x^{(i)}_{1:T_i}$ and the real observations $y^{(i)}_{1:T_i}$. Each of these terms is a standard likelihood calculation in the iETS.

3.4 Forecast

This section presents how to make probabilistic forecasts for a given time series, once the Φ parameters of the network are known. This will be proposed using two significantly different methods. In the first approach, the joint probability of the predicted variables within the prediction range is estimated. The second approach follows that presented in the original paper for iETS, where we assume that the predictions are independent of each other and we forecast each variable separately.

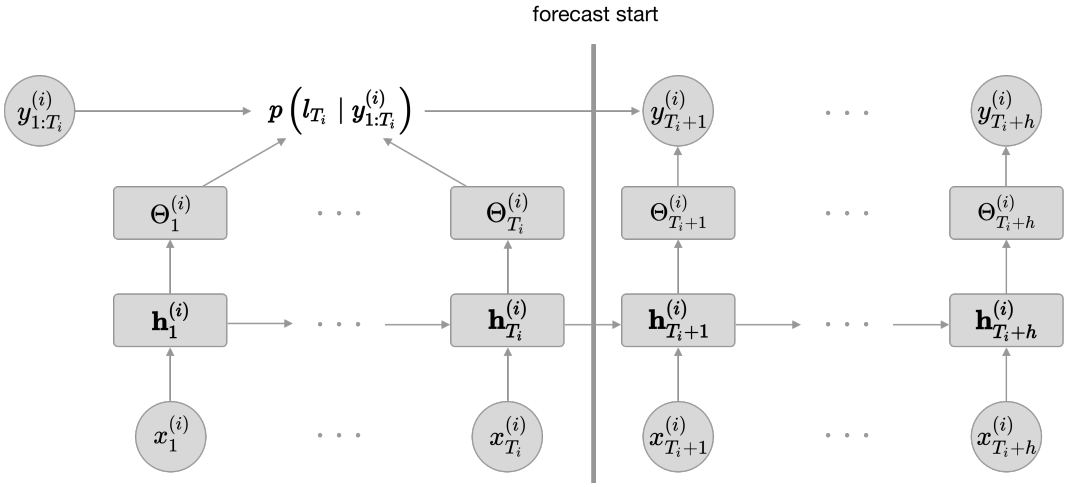

Figure 3.2: Illustration of how the model is used to make forecasts after the network parameters Φ are learned.

The scheme for using the model to make forecast, after learning the Φ parameters of the network, is demonstrated in the Figure 3.2. Given a time series $y^{(i)}_{1:T_i}$ in the training range $\{1, 2, \ldots, T_i\}$ and associated features $x^{(i)}_{1:T_i+h}$ for both training and prediction ranges, forecasts are produced as follows:

1. First, the latent state's posterior $p\left(l^{(i)}_{T_i} \mid z^{(i)}_{1:T_i}\right)$ for the last time step T_i in the training range is computed using the observations $y^{(i)}_{1:T_i}$ and the state space parameters $\Theta^{(i)}_{1:T_i}$ obtained by unrolling the RNN network in the training range and then using the filtering algorithm.

2. Given the latent state's posterior $p\left(l^{(i)}_{T_i} \mid z^{(i)}_{1:T_i}\right)$, prediction samples are gen-

erated where the state space parameters $\Theta^{(i)}_{T_i+1:T_i+h}$ for the prediction range $\Theta^{(i)}_{T_i+1:T_i+h}$ are obtained by unrolling the RNN in the prediction range. This is processed in two ways:

(a) Computing the joint distribution over the prediction range $\hat{y}_{T+1:T+h} \sim p\left(y_{T+1:T+h} \mid y_{1:T}, x_{1:T+h}, \Theta_{1:T+h}\right)$.

(b) Estimating each forecasted value separately $\hat{y}_{T+1:T+h} \sim \left[y_{T+1|T}, y_{T+2|T}, \ldots, y_{T+h|T}\right]$.

3.4.1 Deep iETS$_{\mathbf{MC}}$

In this method, given Φ, we are interested in computing the joint distribution over the prediction range for each time series. However, in practice this is difficult to determine analytically, so it is often more convenient to represent the forecast distribution in terms of K Monte Carlo samples,

$$\hat{y}^{(i)}_{k,T_i+1:T_i+h} \sim p\left(y^{(i)}_{T_i+1:T_i+h} \mid y^{(i)}_{1:T_i}, x^{(i)}_{1:T_i+h}, \Theta^{(i)}_{1:T_i+h}\right), \quad k = 1, \ldots, K \tag{3.35}$$

Then recursively apply the transition equation and the observation model to generate the prediction samples. More precisely, starting with a sample $\ell_T \sim p\left(\ell_T \mid y_{1:T}\right)$, we recursively apply

$$\begin{aligned}
&(1+\epsilon_{T+t}) \sim \log\mathcal{N}\left(0, \sigma^2_{\epsilon,T+t}\right), && t = 1, \ldots h \\
&l_{z,T+t} = l_{z,T+t-1}\left(1+\alpha_{z,T+t}\epsilon_{T+t}\right), && t = 1, \ldots h-1 \\
&z_{T+t} = l_{z,T+t-1}\left(1+\epsilon_{T+t}\right), && t = 1, \ldots h \\
&o_{T+t} \sim \text{ Bernoulli } (p_{o,T+t}), && t = 1, \ldots h \\
&\hat{y}_{T+t} = o_{T+t} z_{T+t}, && t = 1, \ldots h
\end{aligned} \tag{3.36}$$

We generate forecast samples by recursively applying the above equations K times. Finally, we use the mean value for point forecasts or calculate the corresponding quantiles. This method is referred to hereafter as **Deep iETS$_{\mathbf{MC}}$**.

3.4.2 Deep iETS$_{\mathbf{mu}}$ & Deep iETS$_{\mathbf{md}}$

In this prediction method, some assumptions are made about the independence of the prediction range variables, and each value is forecasted separately. This is the approach proposed in the original paper. The predictive values are then as follows:

$$\hat{y}^{(i)}_{T_i+1:T_i+h} \sim \left[y^{(i)}_{T_i+1|T_i}, y^{(i)}_{T_i+2|T_i}, \ldots, y^{(i)}_{T_i+h|T_i}\right] \tag{3.37}$$

In subsection 3.1.1, the conditional expectations, conditional variances, and conditional medians for future observations were discussed. From these values, it is clear that expectations and variances will increase over the forecast horizon. This is due

to the skewness of the logarithmic normal distribution and the increase in uncertainty. This may be a desirable property in some cases, but conditional medians are considered preferable to conditional means. Furthermore, median values have been shown to give more accurate predictions than averages for a logarithmic normal distribution. Medians are considered to be more robust and, in this case, easier to work with. Therefore, two versions of this method are considered later in this thesis. The final forecast for the first version of the proposed approach is the mean forecast. In contrast, the second version is a multiplication of the median demand forecast and the mean demand occurrence. These have been named **Deep iETS$_{\mathbf{mu}}$** and **Deep iETS$_{\mathbf{md}}$**, respectively.

Due to the assumption of logarithmic normality of the residuals, the prediction ranges will be asymmetric. A critical feature of the iETS model used is that it allows for significant one-sided intervals, which is essential when calculating safety stocks. The cumulative distribution function (CDF) can be used to compute prediction intervals for models with intermittent state space:

$$F_y\left(y_{t+h|t} < Q\right) = p_{t+h} F_z\left(z_{t+h|t} < Q\right) + (1 - p_{t+h}) \tag{3.38}$$

where $F_z\left(z_{t+h|t}\right)$ is the h-steps ahead CDF of log normal distribution for $z_{t+h|t}$, $F_y\left(y_{t+h|t}\right)$ is the final CDF of the variable $y_{t+h|t}$ and Q is the value of the desired quantile of the distribution. $F_y\left(y_{t+h|t}\right)$ should correspond to the desired probability (for example 0.9), so the only unknown element in (3.38) is $F_z\left(z_{t+h|t}\right)$, which can be calculated as:

$$F_z\left(z_{t+h|t} < Q\right) = \frac{F_y\left(y_{t+h|t} < Q\right) - (1 - p_{t+h})}{p_{t+h}} \tag{3.39}$$

Thus, in the construction of prediction intervals, the formula can be used to calculate the necessary quantiles of the logarithmic normal distribution of $z_{t+h|t}$.

3.5 Features

The feature vectors $x^{(i)}_{1:T_i+h}$ can be item-dependent, time-dependent, or both. They can be used to provide additional information about the item (e.g., product brand, category) or the time point (e.g., product price, week of the year) to the model. They can also include features that one expects to influence the outcome, as long as the features' values are also available in the prediction range.

3.6 Remarks

In this model, unlike classical and deep learning-based autoregressive models, target values are not used directly as inputs. It is a crucial feature of the method and has several advantages:

1. It makes the model more robust to noise, as target values are only included in the likelihood component, where noise is correctly processed.

2. Generating a sample path of the forecast is more computationally efficient because the RNN only needs to be expanded once for the entire forecast (regardless of the number of samples).

3. Missing target values can be easily handled by simply discarding the relevant probability terms.

Chapter 4

Qualitative experiments

This experiment tested whether the model can successfully recover intermittent state-space parameters when trained on synthetic data. Three models with weekly seasonality but different initial parameters $l_{z,0}$, smoothing of demand size $\alpha_{z,t}$ and demand probability $p_{o,t}$, were prepared for this purpose. For convenience, the same noise variance $\sigma^2_{\epsilon,t}$ was used for all models. A group of time series was then generated based on each model. Each time series consists of 8 weeks of daily data, of which the first six weeks are used to train the model. The group identifier and day of the week are used as an input feature using one-hot encoding. Ideally, for each time series, the model should return the parameters of the iETS from which that time series was generated.

The intermittent state space model parameters in this case are given by

$$\Theta_t^{(i)} = \left(l_{z,0}^{(i)}, \alpha_{z,t}^{(i)}, p_{o,t}^{(i)}, \sigma^{2(i)}_{\epsilon,t}\right), \qquad t = 1, \ldots, T_i + h,$$

where $T_i = 48$ and $h = 14$. Except for $\sigma^{2(i)}_{\epsilon,t}$, all the other parameters are different for each group. Besides, $x_t^{(i)} \in \mathbb{R}^{10}$, where The first three dimensions are used to encode the model identifier and the remaining seven to encode the week's day. In both cases, one-hot encoding is used.

An interesting issue is the amount of data needed to reproduce the parameters. For this purpose, various models have been trained based on increasing numbers of time series for groups. The following amounts are considered $N = \{10, 25, 50, 100, 200\}$ per group. Figure 4.1 presents the results of the experiment. All graphs present valid parameters and parameters obtained from models in the forecast range. The columns show the smoothing parameter of demand size, the probability of the demand occurrence, and the noise variance, while the rows correspond to each group. The probability of demand occurrence has been reproduced quite well, even with the smallest number of sample time-series. However, this quality increases with the number of samples. The noise variance is also reproduced reasonably well, but it does not seem that more examples improve the result. Also,

the initial values of level size were well recovered. The most difficult to reproduce is the smoothing of the demand size parameter. Even with the biggest amount of time series, the original parameters cannot be reproduced entirely. This is due to a "lack" of observations and increased uncertainty in estimating demand's size. However, we still manage to discover these parameters' values and seasonality. It is noteworthy that the model could detect these parameters in a set of time series with three different dynamics.

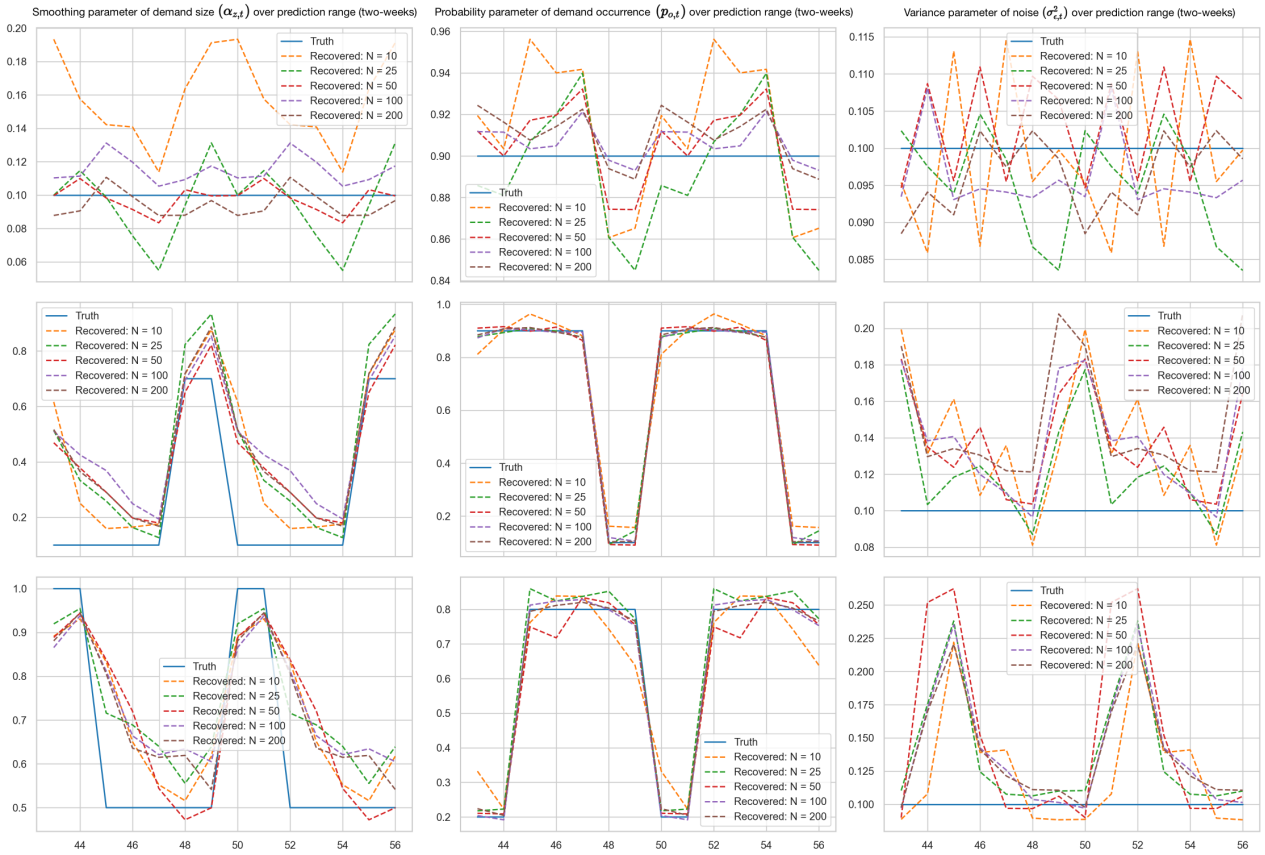

Figure 4.1: Experiments results of the intermittent state-space models' parameters recovery as the number of examples per group increases. The columns show the state space parameters (demand size smoothing parameter, probability of demand occurrence, and noise variance), while the rows correspond to specific groups. Each graph shows the real and recovered parameter values with the increasing number of examples.

Table 4.1 presents in numerical form the results of parameter discovery and compare these values between different sample sizes. The following rows correspond to the results for increasing numbers of sample per group, respectively $\{10, 25, 50, 100, 200\}$. The columns show the summed absolute values of the errors between the parameter's true values and those estimated in the prediction range. Parameters such as demand magnitude smoothing, demand probability, noise variance, initial condition are considered in turn. The previous conclusions are confirmed that as the number

of samples increases, the parameter $p_{o,t}$ is better and better discovered, and the best result is achieved for $N = 100$. A similar situation occurs when predicting the initial state's value $l_{z,0}$, where the best result is achieved for $N = 200$. On the other hand, the most difficult parameters such as $\alpha_{z,t}$ and $\sigma^2_{\epsilon,t}$ were determined best for $N = 50$ and $N = 25$. Therefore, comparing these results in general, satisfactory results are achieved at $N \in \{25, 50\}$, which shows this model's efficiency in terms of the amount of data required.

Number of samples per group	α_z	p_o	σ^2_ϵ	l_0
10	4.6226	2.0863	1.1926	0.2410
25	4.4813	1.1011	**1.0178**	0.1049
50	**4.0481**	0.9538	1.4005	0.0962
100	5.0916	**0.5235**	1.1994	0.0391
200	4.3927	0.5546	1.2956	**0.0338**

Table 4.1: The table presents the summed absolute error values between the true parameters and those estimated when learning the model from different sample time series numbers per group. The columns contain the following parameters: demand size smoothing parameter, probability of demand occurrence, noise variance, initial state. The rows contain the results obtained based on different time series in the training dataset. The best results are marked in bold (less is better).

Chapter 5

Quantitative experiments

5.1 Walmart Sales Dataset

This dataset was made available as part of the M5 Competition on Kaggle, the popular M Competition series's current installment on forecasting [16]. The dataset contains time series data of sales of various Walmart store products organized hierarchically by item level, department, product category, and geographical area. There are 3,075 products classified into 3 product categories (hobby, food, and household) and 7 product departments in which the categories mentioned above are disaggregated. The products are sold across ten stores in three different states (California, Texas, and Wisconsin). In particular, four shops are located in California, three in Texas, and three in Wisconsin. The total number of M5 series across the hierarchy is 42,840. The M5 competition dataset also included exogenous/explanatory variables, including calendar-related information and sales prices. Therefore, in addition to past unit sales of products and relevant timestamps (e.g., date, weekday, week number, month, and year), there was also information available:

- Special events and holidays (e.g., Super Bowl, Valentine's Day, and Orthodox Easter), organized into four classes: sporting, cultural, national, and religious.
- Sales prices are given at weekly shop level (seven-day average). If not available, this means that the product was not sold during the week under-examined. Although prices are fixed each week, they can change over time.
- SNAP activities that serve as promotions. Encoded a binary variable (0 or 1) indicating whether CA, TX or WI shops allow SNAP purchases during the study period.

The forecast horizon (forecasts 28 days ahead) was determined by the nature of the decisions that companies typically support when forecasting data similar to that of M5, i.e., daily series disaggregated across locations and product categories.

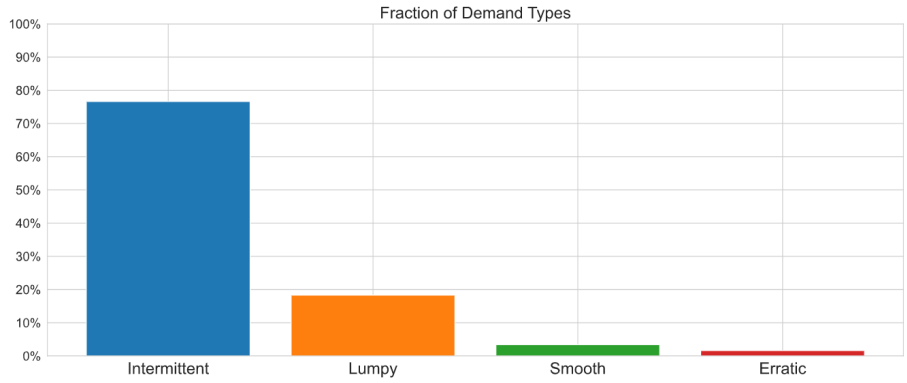

Figure 5.1: Distribution of time series in the dataset according to the type of demand: 23366 intermittent (76.63%), 5587 lumpy (18.32%), 1041 smooth (3.41%), 496 erratic (1.63%).

As shown in Figure 5.1, the fraction of intermittent time series among all time series in the dataset is 76.63%. In comparison, smooth time series account for only 3.41%. This confirms that accurate forecasting of intermittent time series is very much needed and required. Given this fact and the provision of descriptive variables in static (item-dependent) and dynamic (time-dependent) forms, this is an appropriate dataset for the proposed model.

5.1.1 Dataset Processing

The large size of the original dataset and the variety of time series demand categories were divided into six subsets with different characteristics. The categorization of the time series into intermittent, smooth, lumpy, erratic types was performed according to the subsection 2.1.4 conditions. Besides, each collection contains demand series for products from a single department. In all cases, the training period consists of 364 days (52 weeks), while the following 28 days (4 weeks) are used for prediction evaluation. The datasets presented were obtained in the following way. Initially, the time series in the entire set was trimmed to the required length, and only then were they categorized according to demand type. Next, series with a given type of demand and concerning items from a given category (e.g., household 1) were selected. It is worth noting that one item may have several series because it is sold in many stores. Finally, a given number of products and all related time series are selected from such a set. This means that the resulting dataset contains only time series with the same type of demand and linked to each other by items and items' category. The sets under consideration are shown in Table 5.1.

Dataset name	Demand category	Number of time series	Number of items	Items' department
m5_intermittent_foods1	intermittent	281	30	Foods1
m5_intermittent_hobbies2	intermittent	277	30	Hobbies2
m5_smooth_foods3	smooth	291	30	Foods3
m5_lumpy_hobbies1	lumpy	110	12	Hobbies1
m5_erratic_hobbies1	erratic	79	16	Hobbies1

Table 5.1: The table presents the considered data sets that were used to test the capability of the proposed model. In all cases, the training period consists of 364 days (52 weeks), while the following 28 days (4 weeks) are used for prediction evaluation.

5.2 Accuracy comparison

To the best of my knowledge regarding state-of-the-art methods for the problem of demand forecasting, the following models have been used as reference points:

- **Croston**: the Croston method [9] developed for intermittent demand forecasting from R package `tsintermittent.crost()` [14].

- **TSB**: the TSB (Teunter-Syntetos-Babai) method [35], implemented in R package `tsintermittent.tsb()` [14].

- **ETS**: the ETS model [11] with automatic type selection. Used implementation from R package `smooth.es()` [30].

- **iETS**: the occurrence state space exponential smoothing model [31]. It automatically selects the most suitable model from among several versions. Implemented in R package `smooth.oes()` [30].

- **NegBin**: Negative Binomial filter from [28], implemented in the function `counter.negbin()` from R package [29].

- **HSP**: Hurdle shifted Poisson filter discussed in [28] and implemented in the function `counter.hsp()` from R package [29].

- **DeepAR**: recent probabilistic forecasting method, based on autoregressive recurrent networks [25]. To produce non-negative demand output Negative Binomial Distribution has been used. This model uses inference from multiple time series and feature vectors. Applied implementation in Python from `gluonts.model.deepar` [2].

5.3 Accuracy metrics

To measure the prediction accuracy between all competing methods and models, the following error metrics were used. These are described in more detail in subsections 2.1.7, 2.1.8, 2.1.9.

- **sME**: scaled Mean (Signed) Error, determining the potential positive or negative bias in forecast of examined approach;
- **sPIS**: scaled Periods in Stock, acts complementary to sME in providing evidence for systematic behavior with regards to the direction of the forecast error;
- **MASE**: Mean Absolute Scaled Error, measuring the accuracy of point forecasts;
- **sMAPE**: symmetric Mean Absolute Percentage Error, measuring the accuracy of point forecasts;
- **sMSE**: scaled Mean Squared Error, measuring the accuracy of point forecasts;
- **sAPIS**: scaled Absolute Periods in Stock, measuring the magnitude of the systematic bias;
- **SPEC**$_\alpha$: Stock-keeping-oriented Prediction Error Costs, $\alpha = 0.75$ means that Opportunity cost is more important than Stock Keeping cost (common choice in retail), $\alpha = 0.5$ means that both costs are equally important (identical to sAPIS), $\alpha = 0.25$ means that Stock Keeping cost is more critical than Opportunity cost;
- ρ**-Loss**: Quantile loss, p50Loss is adequate when excess inventory is not too worrisome and demand for an item is moderate, p90Loss is appropriate when not stocking an item will result in a large amount of lost revenue;

Firstly, a given metric was calculated for each individual time series, and then the mean of the obtained values was taken. Besides, since it does not interpret some metrics' values, a relative version is considered. It is indicated by adding the prefix *(R)*. The new value was calculated by dividing each model's results by the best result obtained by some baseline models. For example, if the (R)sAPIS of the proposed model is equal to 0.94, it performs 6% more accurately than the best benchmark in terms of the sAPIS.

5.4 Results on intermittent demand

This section tests the proposed model's performance and forecast methods on the target type of time series (intermittent demand). The results are based on

`m5_intermittent_foods1` and `m5_intermittent_hobbies2` datasets, introduced in detail in subsection 5.1.1. An infrequent occurrence of demand characterizes this demand but a small variance between successive values of the demand sizes.

5.4.1 Intermittent: `m5_intermittent_foods1`

The results of the experiment on `m5_intermittent_foods1` data are given in the Table 5.2. As can be seen, the proposed methods outperform the baseline models for most metrics. First, it is the least biased, as shown by the sME and sPIS metrics. Then, all new methods achieve the best result for the metrics sMSE and sMAPE, where the accuracy at each point between the prediction and the target value is calculated. The only TSB has a better score for the MASE metric, but on its basis, it can be seen that all models perform on average worse than the naïve one-step approach. Of course, a multi-step prediction range is considered on these data. Furthermore, the proposed methods are best for metrics such as sAPIS, $SPEC_{0.75}$, and $SPEC_{0.5}$, dedicated to intermittent time series. Although probabilistic prediction and metrics p50Loss and p90Loss, NegBin, and HSP achieved the best result, the proposed methods are next in rank.

	sME	*sPIS*	*MASE*	*(R)sMSE*	*(R)sMAPE*	*(R)sAPIS*	*(R)SPEC$_{0.75}$*	*(R)SPEC$_{0.5}$*	*(R)SPEC$_{0.25}$*	*(R)p50Loss*	*(R)p90Loss*
Deep iETS$_{\text{md}}$	-0.082	-40.561	1.184	**0.979**	0.983	**0.967**	0.849	**0.913**	1.064	1.025	1.046
Deep iETS$_{\text{mu}}$	**0.068**	**18.222**	1.210	0.983	0.965	0.989	0.700	0.923	1.355	1.025	1.046
Deep iETS$_{\text{MC}}$	0.090	23.513	1.215	0.989	**0.964**	0.992	**0.695**	0.939	1.411	1.037	1.053
iETS$_{\text{A}}$	-0.648	-259.339	1.117	1.052	1.100	1.456	2.673	2.284	1.690	1.227	1.590
Croston	-0.375	-148.488	1.119	1.006	1.011	1.075	1.035	1.008	1.003	1.272	1.646
TSB	-0.381	-150.932	**1.115**	1.006	1.011	1.076	1.038	1.009	**1.000**	1.271	1.648
NegBin	-0.297	-116.675	1.116	1.000	1.000	1.000	1.000	1.000	1.042	**1.000**	1.040
HSP	-0.351	-138.451	1.121	1.007	1.009	1.074	1.048	1.026	1.029	2.030	**1.000**
ETS	-0.324	-127.637	1.150	1.032	1.017	1.161	1.253	1.243	1.277	1.297	1.715
DeepAR	-0.286	-118.664	1.127	1.004	1.010	1.032	1.057	1.031	1.028	1.002	1.051

Table 5.2: Accuracy metrics for experiments performed on an `m5_intermittent_foods1` dataset. Metrics with the prefix *(R)* are presented relative to the strongest baseline methods. The best results are marked in bold (closer to zero is better). Additionally, if the result is better than all baseline models, it has been underlined.

5.4.2 Intermittent: `m5_intermittent_hobbies2`

The results of the experiment for the `m5_intermittent_hobbies2` data are shown in Table 5.3. As can be seen, the proposed methods outperform the baseline

models for most metrics. First, it is the least biased, as shown by the sME and sPIS metrics. Then, all new methods achieve the best result for the metric sMSE, where the accuracy at each point between the prediction and the target value is calculated. The original iETS has a better score for the MASE metric, but on its basis, it can be seen that all models perform on average worse than the naïve one-step approach. Furthermore, the proposed methods are best for metrics such as sAPIS, $SPEC_{0.75}$, and $SPEC_{0.5}$, dedicated to intermittent time series. Although probabilistic prediction and metrics p50Loss and p90Loss, NegBin, and HSP achieved the best result, the proposed methods are next in rank.

	sME	*sPIS*	*MASE*	*(R)sMSE*	*(R)sMAPE*	*(R)sAPIS*	*(R)SPEC$_{0.75}$*	*(R)SPEC$_{0.5}$*	*(R)SPEC$_{0.25}$*	*(R)p50Loss*	*(R)p90Loss*
Deep iETS$_{\text{md}}$	-0.226	-107.793	1.456	0.987	1.012	0.931	0.944	0.970	1.045	1.002	1.040
Deep iETS$_{\text{mu}}$	-0.153	-78.905	1.482	0.987	1.008	0.934	0.874	**0.939**	1.084	1.002	1.040
Deep iETS$_{\text{MC}}$	**-0.147**	**-78.170**	1.482	**0.978**	1.008	**0.931**	**0.870**	0.940	1.094	1.002	1.043
iETS$_{\text{A}}$	-0.773	-313.916	**1.232**	1.042	1.027	1.159	1.400	1.266	1.058	1.483	1.227
Croston	-0.558	-226.852	1.312	1.024	1.014	1.036	1.000	1.000	1.029	1.593	1.314
TSB	-0.588	-239.144	1.298	1.034	1.015	1.048	1.012	1.002	1.013	1.582	1.315
NegBin	-0.615	-250.170	1.281	1.039	1.004	1.088	1.061	1.029	**1.000**	**1.000**	1.041
HSP	-0.471	-191.660	1.352	1.000	1.014	1.000	1.002	1.002	1.031	4.928	**1.000**
ETS	-0.651	-264.096	1.362	1.082	1.015	1.330	1.223	1.253	1.344	1.643	1.389
DeepAR	-0.454	-159.742	1.354	1.035	**1.000**	1.126	1.114	1.195	1.377	1.043	1.185

Table 5.3: Accuracy metrics for experiments performed on an `m5_intermittent_hobbies2` dataset. Metrics with the prefix *(R)* are presented relative to the strongest baseline methods. The best results are marked in bold (closer to zero is better). Additionally, if the result is better than all baseline models, it has been underlined.

5.5 Result on other demand types

In this section, I test the proposed model's performance on other demand types such as lumpy, erratic, and smooth, defined in the subsection 2.1.4.

5.5.1 Lumpy: `m5_lumpy_hobbies1`

The results of the experiment for the `m5_lumpy_hobbies1` data are shown in Table 5.4. In this dataset, time series with a lumpy demand-type are considered. Those where demand is infrequent and the variance between successive demand values is large. This is considered the most difficult type of time series to forecast among those studied in this thesis. Moreover, there is no single model here that

outperforms the others, as you can see. Rather, the results are dependent on the metric considered. Firstly, based on MASE, it can be seen that all the models analyzed are performing better than the naïve model on average, as their value is less than 1. In this metric, the original iETS performed best. However, in the other metrics investigating the in point time performance as sMSE and sMAPE, the best score is achieved by the proposed Deep iETS_{mu} and Deep iETS_{MC} approaches, where the original approach has the worst score. For the model bias analysis, Croston and TSB achieved the best results. However, when analyzing the SPEC metric, the proposed approaches achieve better scores than most baseline models, especially Deep iETS. A worse result is achieved by the Deep iETS_{md} version, which may be due to the lack of consideration of the variance of demand values when determining the forecast, which affects the lower variance of values and not fitting into the data type. For the probability distribution predictions, NegBin performed best, but the proposed approaches do not perform much worse and are in the next rank.

	sME	*sPIS*	*MASE*	*(R)sMSE*	*(R)sMAPE*	*(R)sAPIS*	*(R)SPEC$_{0.75}$*	*(R)SPEC$_{0.5}$*	*(R)SPEC$_{0.25}$*	*(R)p50Loss*	*(R)p99Loss*
Deep iETS_{md}	-0.238	-87.774	0.787	1.001	1.023	1.067	1.520	1.143	**0.774**	1.005	1.038
Deep iETS_{mu}	0.023	17.273	0.851	**0.993**	**0.995**	1.023	0.991	1.032	1.072	1.005	1.038
Deep iETS_{MC}	0.046	24.054	0.861	0.993	0.997	1.022	**0.965**	1.025	1.083	1.009	1.043
iETS_{A}	-0.688	-269.090	**0.728**	1.064	1.173	1.707	4.472	2.965	1.487	1.162	1.429
Croston	**-0.019**	2.761	0.846	1.003	1.000	1.086	1.086	1.112	1.139	1.502	1.191
TSB	-0.029	**-1.473**	0.842	1.001	1.001	1.067	1.068	1.092	1.115	1.488	1.195
NegBin	-0.055	-11.819	0.834	1.000	1.011	**1.000**	1.091	1.069	1.047	**1.000**	**1.000**
HSP	-0.030	-1.756	0.839	1.000	1.003	1.001	1.000	**1.000**	1.000	2.087	1.031
ETS	-0.129	-41.358	0.942	1.092	1.043	1.596	1.654	1.701	1.746	1.662	1.534
DeepAR	-0.109	-22.921	0.895	1.102	1.108	1.356	1.964	1.640	1.322	1.083	1.296

Table 5.4: Accuracy metrics for experiments performed on an `m5_lumpy_hobbies1` dataset. Metrics with the prefix *(R)* are presented relative to the strongest baseline methods. The best results are marked in bold (closer to zero is better). Additionally, if the result is better than all baseline models, it has been underlined.

5.5.2 Smooth: `m5_smooth_foods3`

The results of the experiment for the `m5_smooth_foods3` data are shown in Table 5.5. In this dataset, time series with a smooth demand-type are considered. That is, in which the probability of demand occurs is very high, and the variance between successive demand values is low. Furthermore, the demand values themselves are typically larger than in intermittent cases. Thus, this pattern fits the proposed model's definition, but in particular, $p_{o,t} \approx 1$, simplifying to an exponential smoothing state-space model. With this type of time series, it is useful to

analyze metrics comparing the forecasts with targets at each point, such as MASE, sMSE, sMAPE. In this classification, DeepAR is the strongest model, and indeed it performs best. However, Deep iETS_{md} and Deep iETS_{mu} are in second and third position and declassify the other baseline models, showing that they can process this type of series very well. Deep iETS_{MC} performs a little less well, where based on the bias analysis with sME and sPIS, it can be seen that it overestimates. This may be because Monte Carlo simulations of exponential smoothing can quickly explode with the increasing length of the prediction range. Additionally, the proposed methods perform very well in forecasting the probability distribution, as shown by p50Loss and p90Loss, where all versions are not much worse than the strongest DeepAR and outperform the other models. Moreover, the proposed models perform better for the sAPIS metric than the baselines. In other results, it can be seen that NegBin is the least biased and is the best in SPEC metrics, which the original authors did not even consider for smooth demand time series. Hence, it is better not to make any suggestions based on these results.

	sME	*sPIS*	*MASE*	*(R)sMSE*	*(R)sMAPE*	*(R)sAPIS*	*(R)SPEC$_{0.75}$*	*(R)SPEC$_{0.5}$*	*(R)SPEC$_{0.25}$*	*(R)p50Loss*	*(R)p90Loss*
Deep iETS_{md}	-0.052	-11.857	0.818	1.065	1.023	**0.931**	2.995	2.320	1.810	1.040	1.145
Deep iETS_{mu}	0.113	53.727	0.868	1.133	1.042	1.197	2.314	2.899	3.342	1.040	1.145
Deep iETS_{MC}	0.396	125.246	1.201	2.488	1.187	1.985	3.923	6.440	8.347	1.160	1.614
iETS_{A}	-0.515	-199.219	1.061	10.018	3.112	3.318	7.746	4.482	2.010	2.502	5.142
Croston	-0.176	-61.546	0.994	9.577	2.676	2.430	1.344	1.240	1.161	2.540	4.558
TSB	-0.168	-58.462	0.990	9.506	2.667	2.352	1.231	1.141	1.073	2.528	4.538
NegBin	**-0.037**	**-5.040**	0.987	9.003	2.617	2.015	**1.000**	**1.000**	**1.000**	2.066	2.756
HSP	-0.136	-45.308	0.997	9.618	2.662	2.442	1.199	1.139	1.093	3.390	2.732
ETS	-0.108	-34.509	1.057	12.742	2.632	2.866	1.186	1.210	1.227	2.705	4.855
DeepAR	-0.072	-15.712	**0.792**	**1.000**	**1.000**	1.000	4.574	3.134	2.044	**1.000**	**1.000**

Table 5.5: Accuracy metrics for experiments performed on an `m5_smooth_foods3` dataset. Metrics with the prefix *(R)* are presented relative to the strongest baseline methods. The best results are marked in bold (closer to zero is better). Additionally, if the result is better than all baseline models, it has been underlined.

5.5.3 Erratic: `m5_erratic_hobbies1`

The results of the experiment for the `m5_erratic_hobbies1` data are shown in Table 5.6. In this dataset, time series of the erratic demand-type are considered. This is a type of demand characterized by a high frequency of occurrence and a large variance between the demand quantity's successive values. It is worth noting that, as for the smooth type, the SPEC metric was not originally considered for this type of series and be omitted in this analysis. Besides, as can be seen from the

experiment results analysis, the proposed methods outperform the baseline models. Both in the case of metrics comparing in time point predictions like MASE, sMSE, sMAPE and aggregated values like sAPIS. The new approach also achieved a much better result when forecasting the distribution of values in metrics such as p50Loss and p90Loss. Only NegBin achieved a better result for sME and sPIS, indicating a less biased model.

	sME	*sPIS*	*MASE*	*(R)sMSE*	*(R)sMAPE*	*(R)sAPIS*	*(R)SPEC$_{0.75}$*	*(R)SPEC$_{0.5}$*	*(R)SPEC$_{0.25}$*	*(R)p50Loss*	*(R)p90Loss*
Deep iETS$_{\text{md}}$	-0.141	-60.570	**0.679**	**0.479**	**0.870**	**0.692**	5.510	3.430	1.854	**0.815**	**0.818**
Deep iETS$_{\text{mu}}$	0.232	91.013	0.768	0.501	0.888	0.942	2.719	3.702	4.446	**0.815**	**0.818**
Deep iETS$_{\text{MC}}$	0.269	102.030	0.787	0.519	0.896	1.005	2.830	4.007	4.899	0.824	0.842
iETS$_{\text{A}}$	-0.515	-199.219	1.061	1.608	1.537	1.647	7.746	4.482	2.010	1.211	1.882
Croston	-0.176	-61.546	0.994	1.537	1.321	1.206	1.344	1.240	1.161	1.229	1.668
TSB	-0.168	-58.462	0.990	1.526	1.317	1.168	1.231	1.141	1.073	1.223	1.661
NegBin	**-0.037**	**-5.040**	0.987	1.445	1.292	1.000	**1.000**	**1.000**	**1.000**	1.000	1.009
HSP	-0.136	-45.308	0.997	1.544	1.314	1.212	1.199	1.139	1.093	1.640	1.000
ETS	-0.108	-34.509	1.057	2.045	1.300	1.422	1.186	1.210	1.227	1.309	1.777
DeepAR	0.545	248.907	1.067	1.000	1.000	1.934	7.420	12.391	16.156	1.047	1.151

Table 5.6: Accuracy metrics for experiments performed on an `m5_erratic_hobbies1` dataset. Metrics with the prefix *(R)* are presented relative to the strongest baseline methods. The best results are marked in bold (closer to zero is better). Additionally, if the result is better than all baseline models, it has been underlined.

5.6 Experiment conclusions

In this chapter, I have presented the results of experiments on real-world datasets to compare the proposed model with the best, to my knowledge, currently used baseline models for forecasting intermittent time series presented in the literature. On two sets containing time series with intermittent demand, my proposed approach outperformed other existing forecasting methods and several filters in both cases, showing its effectiveness in real-world applications. Moreover, this is the case for all three proposed methods (Deep iETS$_{\text{md}}$, Deep iETS$_{\text{mu}}$, Deep iETS$_{\text{MC}}$) which demonstrates the overall stability of this result. Only the models proposed by Snyder (NegBin, HSP) proved better in quantile losses. In particular, I was able to show that the introduced modification in the form of deep recurrent neural networks improved the performance of the original model very positively.

I also compared the performance of my proposed approach in time series with other demand types such as lumpy, smooth, and erratic. This is essential for forecasting a wide range of products, as their dynamics can evolve and change from

slow-moving to fast-moving (or vice versa). In all these cases, my model has shown good performance, which demonstrates its versatility. For lumpy time series, which is the most challenging type to predict, none of the models is dominant, and my approach for several metrics turns out to be the best, but this result is not stable. In the case of smooth type demand, only DeepAR proved to be better, but it was dedicated to this type of time series, and the model I proposed ranks second. This is important because, even if demand for products is generally smooth, it can be intermittent during introduction and withdrawal from sale. Furthermore, my model was better for in-time point comparing and probabilistic metrics than baseline models for erratic type demand.

Chapter 6

Conclusions

This thesis has proposed an improved statistical model for intermittent demand. This approach extends the basic state-space model proposed by Svetunkov and Boylan [31] by using a deep recurrent neural network for parameterization. This combination allows the structural assumptions of intermittent time series to be explicitly taken into account and, on the other hand, complex patterns to be learned from raw time series data with the additional use of external features. I have presented the mechanism of model construction and proposed approaches to estimating both point and probabilistic values.

I presented the experiments' results initially on synthetic data, where I demonstrated the model's ability to reproduce initially set parameters for different groups of time series occurring in a single data set. I also demonstrated that this is an efficient process due to the number of sample time series needed.

Finally, I compared the proposed model with the best, to my knowledge, currently used baseline models for intermittent time series forecasting proposed in the literature. I started by presenting experiments on two datasets containing time series with intermittent demand. My proposed model outperformed other existing forecasting methods and several filters in both cases, showing its effectiveness when used in real applications. In particular, I was able to show that the introduced modification in the form of deep recurrent neural networks very positively improved the performance of the original model. I also compared my proposed approach's performance on time series with other demand types such as lumpy, smooth, and erratic. My model showed good or outstanding performance in all these cases, which proves its universal applicability. This is essential for forecasting a wide range of SKUs, which can evolve from slow-moving to fast-moving products (or vice versa). Despite this, the proposed model has shown its best apply when the data is heavily intermittent.

As in the original authors' remark, it is also worth mentioning that it is possible to apply another statistical model to forecast the demand size in the proposed ap-

proach, as long as the necessary transformations are made, ensuring that the model gives only positive values. Investigating these modified models' properties could be another area of future research.

Bibliography

[1] Muhammad Akram, Rob J. Hyndman, and J. Keith Ord. „Exponential Smoothing And Non-negative Data". In: *Australian & New Zealand Journal of Statistics* 51.4 (Dec. 2009), pp. 415–432. ISSN: 13691473, 1467842x. DOI: 10.1111/j.1467-842X.2009.00555.x.

[2] Alexander Alexandrov et al. „GluonTS: Probabilistic Time Series Models in Python". In: *arXiv* (June 2019). eprint: 1906.05264. URL: https://arxiv.org/abs/1906.05264v2.

[3] J. D. Croston. „Forecasting and Stock Control for Intermittent Demands". In: *J Oper Res Soc* 23.3 (Sept. 1972), pp. 289–303. ISSN: 1476-9360. DOI: 10.1057/jors.1972.50.

[4] Andrey Davydenko and Robert Fildes. „Measuring forecasting accuracy: The case of judgmental adjustments to SKU-level demand forecasts". In: *International Journal of Forecasting* 29.3 (July 2013), pp. 510–522. ISSN: 0169-2070. DOI: 10.1016/j.ijforecast.2012.09.002.

[5] J. Durbin and S. J. Koopman. *Time Series Analysis by State Space Methods*. 2nd ed. Oxford Statistical Science Series 38. Oxford: Oxford University Press, 2012. 346 pp. ISBN: 978-0-19-964117-8.

[6] A. H. C. Eaves and B. G. Kingsman. „Forecasting for the ordering and stockholding of spare parts". In: *Journal of The Operational Research Society - J OPER RES SOC* 55.4 (Apr. 2004), pp. 431–437. ISSN: 0160-5682. DOI: 10.1057/palgrave.jors.2601697.

[7] Marco Fraccaro. „Deep Latent Variable Models for Sequential Data". English. PhD thesis. 2018.

[8] Rob J. Hyndman. „Another Look at Forecast Accuracy Metrics for Intermittent Demand". In: *Foresight: The International Journal of Applied Forecasting* 4.4 (Jan. 2006), pp. 43–46. URL: https://www.researchgate.net/publication/5055536_Another_Look_at_Forecast_Accuracy_Metrics_for_Intermittent_Demand.

[9] Rob J. Hyndman and Yeasmin Khandakar. „Automatic Time Series Forecasting: The forecast Package for R". In: *Journal of Statistical Software, Articles*

27.3 (2008), pp. 1–22. ISSN: 1548-7660. DOI: 10.18637/jss.v027.i03. URL: https://www.jstatsoft.org/v027/i03.

[10] Rob J. Hyndman and Anne B. Koehler. „Another look at measures of forecast accuracy". In: *International Journal of Forecasting* 22.4 (Oct. 2006), pp. 679–688. ISSN: 0169-2070. DOI: 10.1016/j.ijforecast.2006.03.001.

[11] Robin Hyndman, Anne B. Koehler, J. Keith Ord, and Ralph D. Snyder. *Forecasting with Exponential Smoothing*. Berlin, Germany: Springer-Verlag, 2008. ISBN: 978-3-540-71916-8. DOI: 10.1007/978-3-540-71918-2.

[12] Robin John Hyndman and George Athanasopoulos. *Forecasting: Principles and Practice*. English. 2nd. Australia: OTexts, 2018.

[13] Stephan Kolassa and Wolfgang Schütz. „Advantages of the MAD/mean ratio over the MAPE". In: *Foresight: The International Journal of Applied Forecasting* 6.6 (Jan. 2007), pp. 40–43. ISSN: 1463-6689. URL: https://www.researchgate.net/publication/5055557_Advantages_of_the_MADmean_ratio_over_the_MAPE.

[14] N. Kourentzes and F. Petropoulos. *tsintermittent: Intermittent Time Series Forecasting*. 2016. URL: https://cran.r-project.org/web/packages/tsintermittent/index.html.

[15] Nikolaos Kourentzes. „Intermittent demand forecasts with neural networks". In: *Int J Prod Econ* 143.1 (May 2013), pp. 198–206. ISSN: 0925-5273. DOI: 10.1016/j.ijpe.2013.01.009.

[16] Spyros Makridakis, Evangelos Spiliotis, and Vassilis Assimakopoulos. *The M5 Accuracy Competition: Results, Findings and Conclusions*. Oct. 2020.

[17] Dominik Martin, Philipp Spitzer, and Niklas Kühl. „A New Metric for Lumpy and Intermittent Demand Forecasts: Stock-keeping-oriented Prediction Error Costs". In: *ResearchGate* (Jan. 2020). Comment: Proceedings of the 53rd Annual Hawaii International Conference on System Sciences (HICSS-53), Grand Wailea, Maui, HI, January 7-10, 2020. DOI: 10.24251/hicss.2020.121. arXiv: 2004.10537 [cs, q-fin, stat]. URL: http://arxiv.org/abs/2004.10537.

[18] Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012. ISBN: 0262018020.

[19] Adam Paszke et al. „PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *Advances in Neural Information Processing Systems 32*. Ed. by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett. Curran Associates, Inc., 2019, pp. 8024–8035. URL: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[20] K. B. Petersen and M. S. Pedersen. *The Matrix Cookbook*. Version 20081110. Oct. 2008. URL: http://www2.imm.dtu.dk/pubdb/p.php?3274.

[21] Fotios Petropoulos and Nikolaos Kourentzes. „Forecast Combinations for Intermittent Demand". In: *Journal of the Operational Research Society* 66.6 (June 2015), pp. 914–924. ISSN: 0160-5682, 1476-9360. DOI: 10.1057/jors.2014.62. URL: https://www.tandfonline.com/doi/full/10.1057/jors.2014.62.

[22] A. Nasiri Pour, B. Rostami Tabar, and A. Rahimzadeh. „A Hybrid Neural Network and Traditional Approach for Forecasting Lumpy Demand". In: *World Academy of Science, Engineering and Technology* 40 (Apr. 2009). URL: https://www.researchgate.net/publication/242677209_A_Hybrid_Neural_Network_and_Traditional_Approach_for_Forecasting_Lumpy_Demand.

[23] *R: The R Project for Statistical Computing*. Jan. 2021. URL: https://www.r-project.org.

[24] Syama Sundar Rangapuram, Matthias Seeger, Jan Gasthaus, Lorenzo Stella, Yuyang Wang, and Tim Januschowski. „Deep State Space Models for Time Series Forecasting". In: *Proceedings of the 32nd International Conference on Neural Information Processing Systems*. Nips'18. Montréal, Canada: Curran Associates Inc., Dec. 3, 2018, pp. 7796–7805.

[25] David Salinas, Valentin Flunkert, and Jan Gasthaus. *DeepAR: Probabilistic Forecasting with Autoregressive Recurrent Networks*. Feb. 22, 2019. arXiv: 1704.04110 [cs, stat]. URL: http://arxiv.org/abs/1704.04110.

[26] B. P. Sangal and Asit K. Biswas. „The 3-Parameter Log Normal Distribution and Its Applications in Hydrology". In: *Water Resources Research - WATER RESOUR RES* 6.2 (Apr. 1970), pp. 505–515. ISSN: 0043-1397. DOI: 10.1029/WR006i002p00505.

[27] Matthias W. Seeger, David Salinas, and Valentin Flunkert. „Bayesian Intermittent Demand Forecasting for Large Inventories". In: *Advances in Neural Information Processing Systems* 29 (2016). URL: https://papers.nips.cc/paper/2016/hash/03255088ed63354a54e0e5ed957e9008-Abstract.html.

[28] Ralph D. Snyder, J. Keith Ord, and Adrian Beaumont. „Forecasting the intermittent demand for slow-moving inventories: A modelling approach". In: *International Journal of Forecasting* 28.2 (Apr. 2012), pp. 485–496. ISSN: 0169-2070. DOI: 10.1016/j.ijforecast.2011.03.009.

[29] I. Svetunkov. *counter*. Feb. 2021. URL: https://github.com/config-i1/counter.

[30] I. Svetunkov. *smooth*. Feb. 2021. URL: https://github.com/config-i1/smooth.

[31] Ivan Svetunkov and John E. Boylan. „Multiplicative State-Space Models for Intermittent Time Series". In: *ResearchGate* (Nov. 2017). DOI: 10.13140/rg.2.2.35897.06242.

[32] A A Syntetos, J E Boylan, and J D Croston. „On the Categorization of Demand Patterns". In: *Journal of the Operational Research Society* 56.5 (May 2005), pp. 495–503. ISSN: 0160-5682, 1476-9360. DOI: 10.1057/palgrave.jors.2601841. URL: https://www.tandfonline.com/doi/full/10.1057/palgrave.jors.2601841.

[33] Aris A. Syntetos and John E. Boylan. „The accuracy of intermittent demand estimates". In: *International Journal of Forecasting* 21.2 (Apr. 2005), pp. 303–314. ISSN: 0169-2070. DOI: 10.1016/j.ijforecast.2004.10.001.

[34] R. H. Teunter and L. Duncan. „Forecasting intermittent demand: a comparative study". In: *J Oper Res Soc* 60.3 (Mar. 2009), pp. 321–329. ISSN: 0160-5682. DOI: 10.1057/palgrave.jors.2602569.

[35] Ruud H. Teunter, Aris A. Syntetos, and M. Zied Babai. „Intermittent demand: Linking forecasting to inventory obsolescence". In: *Eur J Oper Res* 214.3 (Nov. 2011), pp. 606–615. ISSN: 0377-2217. DOI: 10.1016/j.ejor.2011.05.018.

[36] Ali Caner Turkmen, Yuyang Wang, and Tim Januschowski. *Intermittent Demand Forecasting with Deep Renewal Processes*. Comment: NeurIPS 2019 Workshop on Temporal Point Processes. Nov. 23, 2019. arXiv: 1911.10416 [cs, stat]. URL: http://arxiv.org/abs/1911.10416.

[37] D. Waller. „Methods for Intermittent Demand Forecasting". In: 2015.

[38] Peter Wallström and Anders Segerstedt. „Evaluation of Forecasting Error Measurements and Techniques for Intermittent Demand". In: *International Journal of Production Economics* 128.2 (Dec. 2010), pp. 625–636. ISSN: 09255273. DOI: 10.1016/j.ijpe.2010.07.013. URL: https://linkinghub.elsevier.com/retrieve/pii/S0925527310002537.

[39] Thomas R. Willemain, Charles N. Smart, and Henry F. Schwarz. „A new approach to forecasting intermittent demand for service parts inventories". In: *International Journal of Forecasting* 20.3 (Feb. 2004), pp. 375–387. ISSN: 0169-2070. DOI: 10.1016/s0169-2070(03)00013-x.

[40] T. M. Williams. „Stock Control with Sporadic and Slow-Moving Demand". In: *Journal of The Operational Research Society - J OPER RES SOC* 35.10 (Oct. 1984), pp. 939–948. ISSN: 0160-5682. DOI: 10.1057/jors.1984.185.