An Evaluation of Multivariate Forecasting Methods on Modelling Unilever's Shipment and Point-of-Sales Data

Steve Kim (10033379952)

Supervised by Prof. Chi-Guhn Lee

University of Toronto Faculty of Applied Science and Engineering

Division of Engineering Science BASc Undergraduate Thesis

April 2021

Abstract

Demand forecasting can help supply chains avoid stock outs and over stocking, improve production lead times, and minimize costs, making it a crucial aspect of effective supply chain management. Unilever, a multinational consumer goods company, is therefore on a mission to model their shipment and point-of-sales data to improve their demand forecasting strategies. In this paper, we investigate different multivariate forecasting models and their ability to forecast Unilever's shipment demands on different products and categories. Several forecasting methods including a vector autoregression model, a convolutional and recurrent neural network model, a temporal convolutional neural network, and an attention based transformer, are evaluated in detail in this paper. We use two data sets for our experiments, both derived from Unilever's shipment and point-of-sales dataset: the Case UPC dataset, consisting of multiple multivariate time series of products, and the Category dataset, consisting of multiple multivariate time series of categories. Our results showed that a convolutional and recurrent neural network called LSTNet performed the best in generating the desired 12 week forecasts. Additionally, forecasting models were shown to have better success in modeling the Category dataset compared to the Case UPC dataset. All the experiment codes are available online.

Contents

1 Introduction

Demand forecasting is a crucial aspect of effective supply chain management. It can provide key insights to be used for important managerial decisions in supply chain operations such as production planning, demand planning, raw material planning, and inventory control. Unilever, a multinational consumer goods company, is therefore on a mission to model their shipment and point-of-sales data to generate twelve week shipment demand forecasts for their products.

Research in improving demand forecasting and methods of efficient supply chain management is abundant in literature. However, due to the *non-stationarity* of time series data in the real-world, forecasting results for multivariate time series are often non-satisfactory in both literature and in practice [38]. More importantly, since all real-world time series data is different in nature, the problem lies in finding a forecasting method that works best for a given dataset.

For background, non-stationary data is defined as time series data that has a variable variance and mean [6]. In the context of this research, non-stationarity also refers to varying demands in a volatile market resulting from unexpected circumstances. Naturally, modelling such data is immensely difficult. While statistically non-stationary data that consists of trends, seasonalities, or random walks can be made stationary using well known methods that use differencing, detrending, and deseasonalizing techniques [20], market demand data are often entangled with unpredictable external variables that lead to spurious behaviours in the data. External factors such as these make the data much harder to model and difficult to extract key variables and their effects on the market demand.

Therefore, the main objective of this research is to provide comprehensive comparisons between multiple multivariate forecasting methods, in an effort to find the most effective methods for modelling Unilever's non-stationary shipment data. Empirical and statistical analysis on Unilever's data is also conducted to facilitate in the development of our methods. As a result, this paper will be contributing valuable information to Unilever that can be used in their own research, and act as a starting point for developing and integrating novel methods to be utilized in Unilever's supply chain management processes.

2 Literature Review

Forecasting time series data has many practical significance across multiple disciplines, attracting countless academics and individuals to contribute to the research. This review aims to examine well known methods as well as interesting approaches that exist in literature that

can be applied to our research problem, and that are comparable to the proposed methods in this paper. A review of hybrid machine learning models is also included for exploring future opportunities in moving this research forward.

2.1 Statistical Methods

2.1.1 The Need for Statistical Methods

With the rise of machine learning in recent literature, statistical models are often overlooked and seen as less superior in forecasting non-linear time series when compared to new state-of-art machine learning models. Although statistical models require more assumptions on a given dataset, they have been shown to outperform complex machine learning learning models in forecasting market data in certain scenarios [15], and are computationally more efficient.

Statistical methods include tools that can help in understanding the data such as the Augmented Dickey-Fuller test and Levene's test, which helps determine the stationarity and homoscedasticity of a dataset [18, 27]. Data analysis and understanding is a prudent step in time series forecasting, as necessary transformations or adjustments can be made to the data to fit the specific models this research aims to implement [20]. Thus, statistical methods should not be overlooked in this research and be used as a benchmark for more complicated models and data understanding.

2.1.2 ARIMA

Statistical models for forecasting have been researched heavily in the past with promising results for some time series in research and industry. Most notably, the Autoregressive integrated Moving Averages (ARIMA) model [3], a seminal work of Box and Jenkins in the late 70s, has been widely used in application and research for forecasting stationary and non-stationary data using mathematical linear models and data transformations with promising results. This class of models had been popularized in both academic and professional fields in the past [32]. However, only using single statistical models such as ARIMA can fail to capture changing probability distributions of the time series data, and thus perform poorly on data that follow a more heteroskedastic behaviour.

2.1.3 A non-parametric statistical method for adapting to changing distributions

Nonetheless, statistical methods that can adapt to changing probability distributions of a time series data for forecasting demand do exist. A non-parametric method for adapting to changes in probability distributions was proposed by Murty in 2006, and used to forecast demand for supply chain operations [17]. He proposed the use of empirical distributions[1] to model random variables and update probability vectors using weighted least squares. More formally, an updated probability vector is defined as the weighted sum of the currently held empirical distribution and the most recently observed histogram. The updated empirical distribution is used to determine the optimal order quantity for an upcoming period using the newsvendor model, which takes the overage and underage cost functions to be piecewise linear, and calculates the expected sum of costs for a given order quantity. Murty also extends this to portfolio management strategies to use the forecast as a suggestion to improve a supply chain's portfolio. These methods are applicable to the non-stationary demand data this research aims to forecast, and provides a simple method of updating probability distributions that is worth investigating.

2.2 Machine Learning Models

Despite the research efforts in statistical forecasting methods, many of the models exhibit insufficiencies to represent real-world time series data that exhibits more nonlinear behaviours in practice [4, 11]. Thus, a wider range of models have become apparent in literature, such as models based on schemes [11], and one that has found the most popularity in recent years: Artificial Neural Networks (ANNs).

2.2.1 Performance of LSTMs

A recent paper published in 2020 by Cordoni provides a comprehensive comparison of modern deep neural network architectures for energy spot price forecasting [5]. The results demonstrate that the Long short-term memory (LSTM) [8] model performed the best when compared to the multilayer perceptron and convolutional neural network models. Similar results were found in other literature as well [1, 21, 14]. Thus, LSTMs have been firmly established as state of the art in sequence modeling. It has also been shown to learn non-linear and non-stationary nature of a time series better than any other state-of-art algorithms such as Neural Networks, ELMs, and Support Vector Regression models [23]. This makes LSTM a desirable model to be evaluated for the purposes of this research.

[1] discrete probability distributions constructed by relative frequency histogram

2.2.2 Temporal Hierarchies with LSTM for improved accuracy

Naturally, the use of LSTM models has been present in the study of forecasting supply chain demands as well, coupled with the use of temporal hierarchies in a recent publication by Punia et al [24]. Temporal hierarchies are defined as a time series that is represented in a hierarchical way, where the deepest child nodes represent the highest available sampling frequency m per year, and k is the number of aggregates that can be constructed in a factor of m. For instance, for quarterly data, m would be 4, and $k \in \{4, 2, 1\}$, starting from 4 three month nodes, then 2 semi-annual nodes, and finally 1 year old at the root. The parent nodes are a summation of its children nodes all the way up to the root. In Punia's research, they applied such temporal hierarchical frameworks with LSTMs as the base forecasting method to determine the initial forecast at each level of the hierarchy [24]. Through reconciliation, initial forecasts were reconciled to be accurate forecasts of each level and used for analysis, utilizing the information from all levels. This proved to be a more accurate model compared to the direct LSTM model for all short, medium, long term forecasts.

2.2.3 Attention based networks and Few Shot Learning

Although LSTMs have been widely accepted as state of the art in sequence modeling in the past, Attention based networks have become more and more used in literature and industry. Companies such as Google [35] and Facebook [7] have turned to attention based models such as ResNet and the Transformer, due to the amount of resources required to train and run LSTM models. By using a new simple network architecture that solely uses attention mechanisms, the model showed to be superior in accuracy and training time compared to LSTMs and other recurrent or convolution neural networks in encoder-decoder configurations [35].

Simply put, attention mechanisms allow models to place more "Attention" on the relevant parts of the input sequence as needed. This concept is also prevalent in Few Shot Learning models that are used in classification problems to learn to classify target tasks given very few data points for training. It uses a combination of LSTMs and attention mechanisms that help in focusing the model to the most relevant information in the encoder hidden states. Few Shot Learning uses a meta learning framework which enables models to learn how to learn to classify a set of training tasks, and classify test tasks which were unseen in the training tasks. This is achieved through three distinct families of meta learning, all of which exploits different types of prior knowledge [37]. A recent publication by Iwata et al. [10] used an attention mechanism that utilized prior knowledge about similarities to be applied to time series forecasting on stock market data using matching networks [36]. The attention

mechanism here gave attention to support hidden states, and learned embeddings in training tasks that tend to separate different classes, even when they had not been seen [10]. For demand data such as the one used for this research, few shot learning architectures can be of great use as it generalizes well given a small input data set, and does not overfit to non-stationary movements of the data. Different meta learning frameworks will be implemented in this research, and evaluated against each other to determine which works best for our dataset.

2.3 Hybrid Machine Learning Models

Market data is dependent on many external variables that are less predictable, and thus cannot be forecasted accurately solely on its past behaviour. Therefore, a hybrid approach will be examined that integrates external data to better predict demand in unforeseen circumstances in an attempt to find unseen variables that the future demand depends on. Additionally, hybrid models have been generally shown to have higher accuracies than traditional models in literature [25, 26, 40, 33]. Methods of integrating different models and data will be briefly discussed in this section.

2.3.1 A general hybrid approach: HDNN - Hybrid Deep Neural Networks

A general hybrid deep neural network architecture was proposed by Yuan et al. outlining an architecture that allows flexible input types for modelling and training [39]. For each dataset input, embedded feature vectors are generated using feature learning methods, and concatenated to be provided as inputs to the final neural network to output a target label. This architecture is very modular, and can be easily modified to fit the specific needs of our research. The generation of learned feature embeddings that can be concatenated and fed into the final neural network is the difficult aspect of this architecture, and is not discussed in detail in the publication. Previously discussed attention mechanisms may be useful in finding effective embeddings if a sequence to sequence model is used.

2.3.2 Incorporating news data in the SI-RCNN model

The stock market follows an unpredictable non-stationary behaviour that depends on external variables such as news and human sentiments similar to our dataset. Vargas et al [34]. integrated news titles and technical indicators in an attempt to improve accuracy of stock price predictions. News titles were vectorized through a Word2Vec model and fed through a Convolutional neural network and pooled to represent semantics of the text. Technical indicators were fed through an LSTM to extract temporal characteristics, and concatenated

with the text semantics to be fed through a fully connected network for a final prediction. By using the techniques used in this paper to extract the news title semantics and coupling it with the output of a more powerful model to embed demand data, the forecast will have taken into account news data. The results of such a model is worth investigating.

2.3.3 Fuzzy logic in the ARIMA-WNN model

Integration of different models can also be done using fuzzy logic as demonstrated in [9]. The final prediction of the ARIMA-WNN is a result of a weighted aggregation of the predictions made by the ARIMA and WNN models. The weighting is updated during training to give more weight to the more accurate model. This can be extended to more than two models, but may result in an overpowering model that drives the final predictions. Models will need to go through a careful selection process such that different models make predictions that optimize on different scenarios with the least amount of overlap. This ensures models make predictions that optimize on different scenarios so that the models are able to effectively mitigate each other's insufficiencies.

2.4 Conclusion

This literature review was a examination of well known and novel forecasting methods that serves as a strong starting point for this research. The review on the significance of statistical methods, state-of-art performances of recurrent neural networks, and novel attention based networks, were heavily considered in selecting the forecasting methods to be evaluated in this research.

3 Unilever's Shipment and Point-of-Sales Data

To better understand the data preparation and exploration methods used in this paper, this section will outline high level information about the raw data provided by Unilever. Unilever's shipment and point-of-sales data consists of 27 columns. Their descriptions provided by Unilever is summarized in Table 1.

Each row is a data entry consisting of shipment and point-of-sales data for a product with product ID CASE_UPC_CD. Each entry is timestamped by three different columns: WeekNumber, YearMonth, and Year. The WeekNumber is the most granular and will be used as the time axis for our time series. Products can be organized into different product groups which have differing levels of granularity; specifically, there are 259 unique promotion groups, 80 brands, 18 categories, and 3 divisions.

Column Name	Description
CASE_UPC_CD	Product ID
WeekNumber	Ordinal Week Number
YearMonth	Year Month Description
Year	Year Number
MIN_WeekNumber	First Shipping Week
MAX_WeekNumber	Last Shipping Week
ShipmentNSV	Dollar Amount Shipeed
ShipmentUnits	Units Amount Shipped
ShipmentCases	Case Amount Shipped
ExplodedShipmentCases	Cases Calculated from other mixed pack
ExplodedShipmentNSV	Dollar Calculated from other mixed pack
ExplodedShipmentUnits	Units Calculated from other mixed pack
ShipmentPricePerUnit	Dollar Per Case
FinalCustomerExpectedOrderQty	Ordered Cases
DispatchedQty	Shipped Cases
ChangeLossesQty	Ordered Cases - Shipped Cases
POSSales	Point of Sales Dollars
POSUnits	Point of Sales Units
POSCases	Point of Sales Units / UnitsPerCase
QtyOnHand	Units in Inventory
POSPricePerUnit	POS Price
FeaturePrice	Internal Event Price
PGDesc	Promotion Group
BrandDesc	Brand
CategoryDesc	Category
DivisionDesc	Division
UnitsPerCase	Number of Units Per Case

Table 1: Unilever's shipment and point-of-sales data column descriptions

4 Problem Formulation

The main objective of this research can be defined as a multivariate, multi-step ahead forecasting problem. More formally, let $X = (\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_T) \in \mathbb{R}^{m \times T}$ define our multivariate time series, where m is the number of variables (or features) in the time series, and $\mathbf{x}_i = (x_i^1, x_i^2, ..., x_i^m)$ contains the measurements of the input variables at time i. T is often referred to as the input sequence length or input window size in literature. The goal is to predict potential values of the multivariate time series at different time steps in the future, which we can denote as $Y = (\mathbf{y}_{T+1}, \mathbf{y}_{T+2}, ..., \mathbf{y}_{T+p}) \in \mathbb{R}^{m' \times p}$, where m' is the number of output variables, and $\mathbf{y}_j = (y_j^1, y_j^2, ..., y_j^{m'})$ is the predicted measurements of the multivariate time series at time j. p is often referred to as the prediction horizon or output sequence length. In this paper, the objective is to generate 12 week forecasts of the *target variable*, meaning p will equal to 12, given that each time step is indexed by week. The *target variable*, or the variable we wish to forecast is "ShipmentCases".

In multi-step forecasting problems, there are two methods that are used in producing

p-step forecasts. One method is to train models to generate predictions for multiple time steps ahead at once, often referred to as *direct forecasting* in literature. The other method is to train models to predict one time step ahead and iteratively predict up to a desired horizon, referred to as *iterative forecasting* in literature. Which approach is better is dependant on the model and dataset, and needs to be examined on a case by case basis [16]. Thus, both methods were implemented in this paper to see their performances on Unilever's dataset.

5 Methods

In this paper, five models are proposed to be evaluated on two datasets derived from Unilever's shipment and point-of-sales data. Justifications on selecting the models are outlined in the models' respective sections. The methods for creating the datasets, conducting data analysis, training and evaluating the models, and the details of the models are outlined in this section. All chosen methods and decisions are justified when necessary. Resulting plots and analysis can be found in Section 6.

5.1 Creating the Case UPC and Category Datasets

As discussed in Section 3, there are many levels in which the products can be grouped into. Unilever is most interested in forecasting on a product (case UPC) level; however, for additional analysis on the data and models' performances, this paper will also evaluate models on the category level.

Thus, two datasets, one consisting of time series for each case UPC, and one consisting of time series for each category, need to be created. First, since Unilever's data is not indexed by a date-time axis, the WeekNumber column consisting of the year and week number needs to be parsed and formatted to be in a 'year-month-day' format, which is common for time series data. For consistency, the date is chosen to be the Monday of the given week number on the given year.

Then, the Case UPC dataset was created by mapping each product's UPC to its time series. Since there is only one data entry for a specific product in a given week, the time series of a product was set to be the product's data entries in the original dataset sorted by date. The Category UPC dataset was created by grouping data entries by category, and summing the columns of all the products that belonged to the category on a given date. Similar to the Case UPC dataset, the categories' time series were mapped to its corresponding category names. Lastly, normalization of data has been proven to be a crucial step in obtaining good performances and speeding up training of neural networks [29]. Thus, all time series

were normalized between 0 and 1 using min-max normalization which is commonly used in literature [22].

5.2 Data Exploration

Understanding the data is a crucial first step to any modelling problem. Data analysis was thus conducted both quantitatively and qualitatively, to first understand the nature of the time series visually, and then extract statistical information that could not be observed.

5.2.1 Feature Visualization and Selection

After reviewing the description of each of the columns provided by Unilever and discarding columns that could not be used as feature inputs, twelve columns were selected to be visualized as time series along a shared time axis. Two random products and categories were selected from the datasets for visualizing their 12 features.

Having an optimal feature subset greatly reduces the feature dimensionality and helps in optimizing the performance of machine learning algorithms [19]. Thus, by observing the plots, stale or repetitive features were discarded from the feature set as part of the feature selection process.

5.2.2 Feature Importance

For further analysis on the features and reducing the feature space, the importance of features at predicting the target variable was analyzed. For computing feature importance, two common statistical measures for time series were considered: the Pearson's Correlation, and Mutual Information.

Since Pearson's correlation captures only linear dependencies between an input variable and a target variable, Mutual information, which can capture any kind of dependency between variables, was selected to be used for our data which is highly volatile and non-linear in nature [31].

The mutual information scores of each feature for twenty five random products and sixteen categories were plotted. The average mutual information scores of the features across all time series in the Case UPC and Category datasets were plotted as well for further analysis.

5.3 The Proposed Models

In this paper, the Vector Autoregression, LSTNet, MultiLSTNet, TCN, and Transformer models were selected to be evaluated on the Unilever dataset. This section outlines each model's architecture, a high level overview of how the model works, and why it was selected for evaluation.

5.3.1 Vector Autoregression Model (VAR)

VAR is a statistical model used to capture relationships between multiple time series that is famously used in literature and in practice. This makes VAR a great model for Unilever's multivariate time series data, and a baseline method to be used for comparison.

VAR can be seen as a generalization of a univariate autoregressive model such as ARIMA discussed in Section 2.1.2. The p-th order VAR model can be summarized by equation 1.

$$\mathbf{y}_t = c + A_1\mathbf{y}_{t-1} + A_2\mathbf{y}_{t-2} + ... + A_p\mathbf{y}_{t-p} + \mathbf{e}_t, \tag{1}$$

$\mathbf{y}_i \in \mathbb{R}^k$ for $i \in [t-p, t]$, k is the number of features we provide, A_i is a time invariant $k \times k$ matrix, $\mathbf{e}_t$ is an error vector of size k, and p is the input window size. $\mathbf{y}_t$ is the set of variables we want to predict at time step t. Each variable in datapoint $\mathbf{y}$ is given an equation modeling its evolution over time. We see that the equation includes the variable's historical values and the lagged values of other variables in the model. After estimating the matrices and regression parameters, we can make forecasts by providing the estimated VAR model with an input sequence of length p with k features.

5.3.2 LSTNet

LSTNet is a convolutional and recurrent neural network proposed by Lai et al [13]. This model was chosen due to its novel approach and results when the paper was first published. For instance, the use of recurrent-skip layers as well as an integrated autoregressive component to help capture linear dependencies of the input data was shown to be innovative and effective, resulting in the LSTNet model outperforming many well known regression models [13]. An overview of the LSTNet architecture is shown in Figure 1.

First, the input data is fed through a convolutional layer which encodes the input and attempts to extract short-term temporal patterns and local dependencies between variables. Then, the data passes through the recurrent and recurrent-skip layers which help capture very long term dependencies that are not captured due to gradient vanishing problems still prevalent in GRU and LSTMs in practice. Lastly, the data is fed through a dense layer for

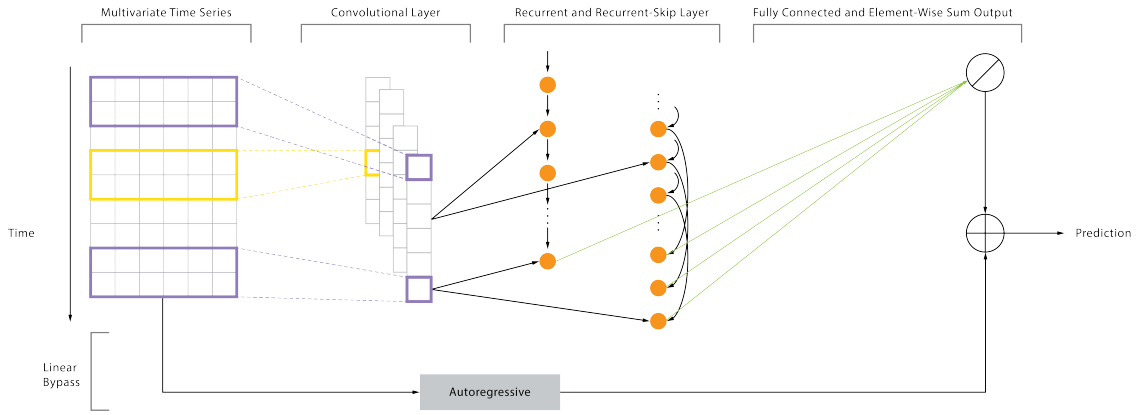


Figure 1: Overview of the LSTNet architecture

a prediction output, which is added with the output of a linear bypass which attempts to tackle scaling input issues that exists in real-world data. The model outputs a single step forecast, and thus requires iterative forecasting as described in Section 4. For more details on each of the components of the LSTNet model, refer to [13].

5.3.3 MultiLSTNet

MultiLSTNet is a simpler variation of the LSTNet model that has been modified to generate direct (multi-step) forecasts. We propose the MultiLSTNet model to observe the difference in performances between multi-step and uni-step forecasting models using similar model architectures. An overview of the MultiLSTNet architecture is shown in Figure 2.

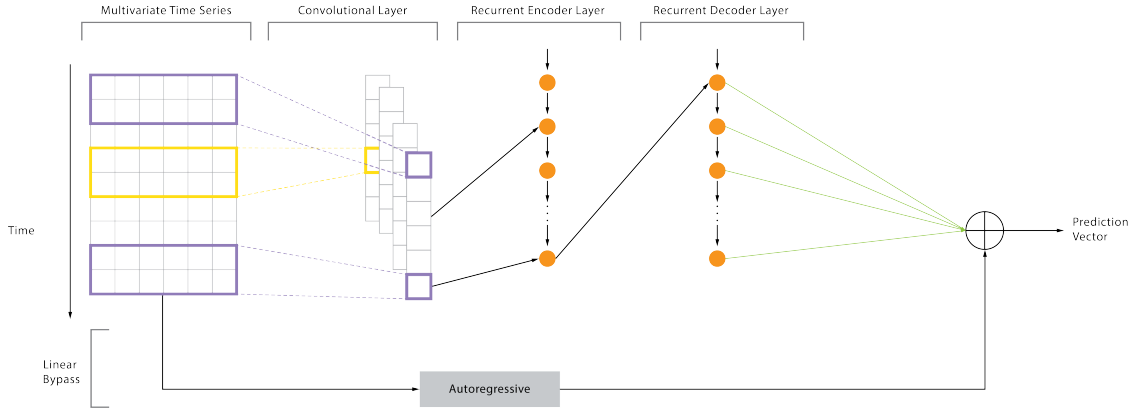


Figure 2: Overview of the proposed MultiLSTNet architecture

The major difference between LSTNet and MultiLSTNet is that the recurrent skip layer and the last dense layer of the LSTNet is replaced with a decoder LSTM that outputs the prediction vector of size (output_size, num_features), where the output_size is the forecasting horizon. Since there is no recurrent skip layer, the last hidden state of the recurrent encoding

layer is stacked and fed into the decoder LSTM as its input. All other components of the LSTNet model were retained.

5.3.4 Temporal Convolutional Neural Network (TCN)

The Temporal Convolutional Neural Network is made up of dilated, causal 1D convolutional layers that have the same input and output lengths. This architecture was first introduced by Bai et al. and was selected for our research due to the paper's novel results. The paper demonstrated that convolutional networks could achieve better performance than RNNs in many sequential modeling tasks while avoiding common drawbacks in recurrent models, such as the vanishing or exploding gradient problems [2]. Thus, we propose to use the TCN architecture as described by Bai et al., using multiple encoding and decoding residual blocks. The overview of the TCN model implemented in this paper is shown in Figure 3.

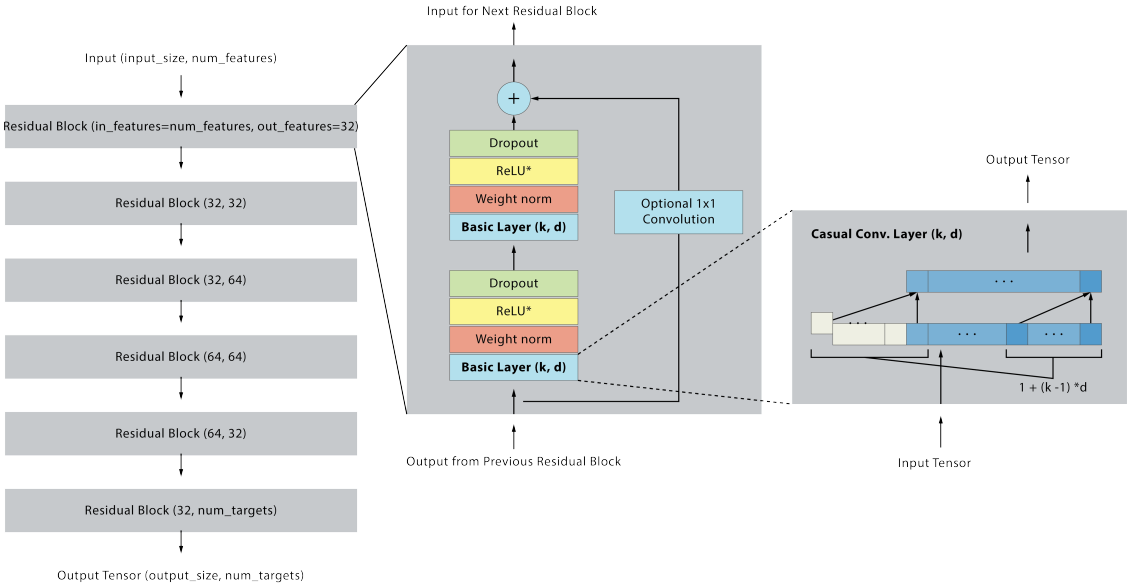


Figure 3: Overview of the proposed TCN architecture

For details on dilated causal convolution layers and residual blocks, refer to [2]. The first four residual blocks act as encoders that encodes the feature space into higher dimensions. The last two blocks act as decoders that outputs the predictions of size (output_size, num_targets), where num_targets is the number of target variables we wish to forecast, and output_size is the desired length of the output sequence. The input tensor is of size (input_size, num_features), where input_size is the length of the input sequence.

5.3.5 Transformer Encoder

The Transformer is a model based solely on attention mechanisms that was introduced by a team of researchers at Google in the famous paper "Attention Is All You Need" [35]. As mentioned in the Literature Review, attention mechanisms helps models place more "attention" on the relevant parts of the input sequence. Using solely this mechanism, the Transformer model was superior in both accuracy and training time compared to LSTMs and other recurrent or convolution neural networks. Thus, this model was naturally selected to see its performance on Unilever's data. In this paper, the Transformer's encoder block is used as described in [35]. The overview of the Transformer encoder architecture is shown in Figure 3.

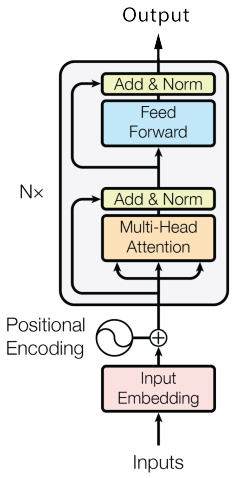


Figure 4: The Transformer Encoder architecture as seen in [35]

The Transformer Encoder consists of three main components: a positional encoder, a multi-headed attention module, and a feed forward neural network. The first component, the positional encoder, encodes input embeddings such that the encoded input contains positional information with respect to other inputs in the input sequence. Secondly, the encoded inputs are fed into a multi-head attention module, which runs self attention mechanisms several times in parallel. Independent attention outputs are then concatenated and linearly transformed into a specified dimension. Lastly, the feed forward neural network is applied to every attention vector to transform the vectors into the output size we desire. For more details on how the Transformer Encoder model works, refer to [35].

5.4 Implementing the Models

All machine learning models were developed using the PyTorch library, closely following the models' architectures as described in the referenced papers. The VAR model was im-

plemented using the `statsmodels.tsa` Python module which contains useful functions and classes that are useful in statistical time series analysis. The code used for this paper can be found on Github: github.com/kimwoo11/thesis-forecasting-models.

5.5 Training Process

The VAR model was estimated using the `statsmodels.tsa` Python module. Since the model requires input time series to be stationary, time series were first evaluated for stationarity using the Augmented Dickey–Fuller test [18]. Then, differencing was applied to time series that were non-stationary, and fed into the VAR model for estimation.

All deep learning models proposed in this paper were trained using the same optimization strategy. First, let's formalize the problem to be a regression task that can be solved using gradient descent. Given a time series $Y = (\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_t, ..., \mathbf{y}_{T-1}, \mathbf{y}_T)$, where $\mathbf{y}_i \in \mathbb{R}^m$ and m is the number of variables/features, we define an input indexed by t to be $X_t = (\mathbf{y}_{t-q+1}, \mathbf{y}_{t-q+2}, ..., \mathbf{y}_t)$, for a tunable input window size q. We define the target to be $Y'_t = (\mathbf{y}'_{t+1}, \mathbf{y}'_{t+2}, ..., \mathbf{y}'_{t+p})$, where p is the forecasting horizon, and $\mathbf{y}'_i \in \mathbb{R}^{m'}$, where m' is the number of target variables. Given a set of input-target pairs $\{X_i, Y'_i\}$ derived from time series Y, the problem then becomes a regression task that can be solved using gradient descent. The Adam optimizer [12] is used to train all models in this paper, as it has been shown to help models escape saddle points faster, and converge faster overall to second-order stationary points during training [30]. The criterion used during training was the Root Mean Squared Error (RMSE) as defined in Section 5.6. The RMSE was selected as the criterion as it penalizes large errors more compared to other metrics such as Mean Absolute Error.

Each time series in the Case UPC and Category datasets were processed, and split into training, validation, and test sets, as commonly done in training and evaluating machine learning models effectively. The test set is made up of the last $q + p$ time steps for a given time series, where q is the size of the input window, and p is the forecasting horizon. The test set therefore has one input-target pair with an input sequence of length q and an output sequence of length p. The remaining time series data was used to create input-target pairs for the training and validation sets, which were split 90%/10%. The validation set was chosen to be 10% which is enough to help indicate over-training, but leaves enough data for the training set.

Each of the five models were trained on the entire Case UPC and Category dataset resulting in two trained models. Since each input-target pair is independant of each other and are normalized separately, models were able to train on all input-target pairs for each dataset. This enabled the models to learn temporal patterns across the entire Case UPC

or Category datasets. In addition, time series of five products and five categories were randomly selected for the models to train on separately, resulting in ten trained models that are each responsible for a specific product or category. This allows for additional analysis and comparisons between different methods of training the model and producing forecasts. For a more comprehensive comparison between models trained for specific time series and models trained across datasets, models will need to be trained for all products and categories. Only ten time series were selected due to limited computing resources.

Hyperparameters of each of the models were manually tuned. With more computing power, it is recommended that a formal grid search is conducted for effective hyperparameter tuning.

5.6 Evaluation Metrics

Models were evaluated on the test sets using three different metrics: RMSE, bias, and accuracy. The three metrics are formally defined in Table 2.

Metric	Equation
RMSE	$\sqrt{\sum_{i=0}^{n} \frac{(\hat{y}_i - y_i)^2}{n}}$
Accuracy	$1 - \frac{abs(target - prediction)}{target}$
Bias	$\frac{target - prediction}{prediction}$

Table 2: Unilever's shipment and point-of-sales data column descriptions

To visualize the model's performances, the model's predictions were plotted against their target values. Histograms of the accuracy of the models' predictions on the test sets were also plotted for further analysis and discussion.

6 Results and Discussion

In this section, results from data exploration, training, and model evaluations are presented. Each Results section consists of a Discussion section that highlights key findings from the results and analyzes them thoroughly.

6.1 Data Exploration Results

6.1.1 Feature Visualization and Selection

As mentioned in Section 5.2.1, twelve features were selected for visualization through a manual selection process. Features of two random time series from the Case UPC and Category datasets were plotted for quantitative analysis and initial feature selections. The resulting plots can be seen in Figure 5.

Figure 5: Visualization of features for 4 sample time series taken from the Case UPC and Category datasets

Discussion. Three main observation can be made from Figure 5. First, many of the features' time series such as 'POSUnits', 'POSSales', and 'POSCases' are identical in shape. This is as expected since many features hold redundant information that is present in other features. For instance, the number of product cases and units that were shipped are are linearly dependant since there are a set number of product units that go into a product case. Thus, the two features are capturing the same data in different units of measurement. Secondly, the 'ChangeLossesQty' feature does not seem to have much movement across time, appearing stationary and mostly consisting of zeros. This will not contribute useful information in the forecasting process and may hinder the learning process. Lastly, the

'QtyOnHand' feature appears to have zero entries for roughly the first half of the time series, making the feature unusable for those time steps.

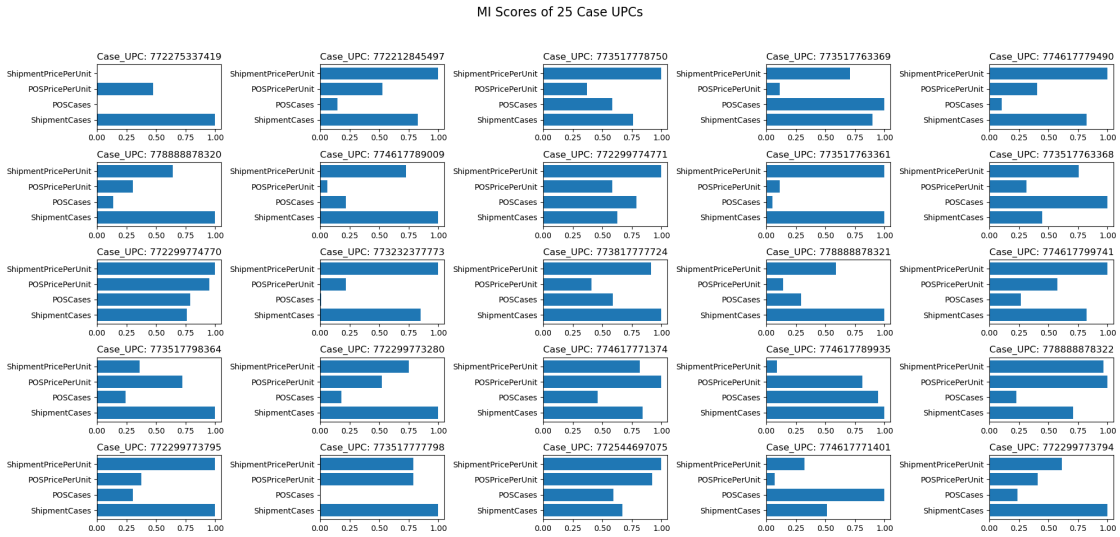
Removing redundant and unusable features, four features remained: 'ShipmentCases', 'POSCases', 'POSPricePerUnit', and 'ShipmentPricePerUnit'. 'ShipmentCases' remained since it is the target variable as described in Section 4, 'POSCases' was selected over other point-of-sales features to be consistent with the target variable, and the price per unit features were selected to be used to test if the price of a product has an influence over shipment demands using the feature importance analysis.

6.1.2 Feature Importance

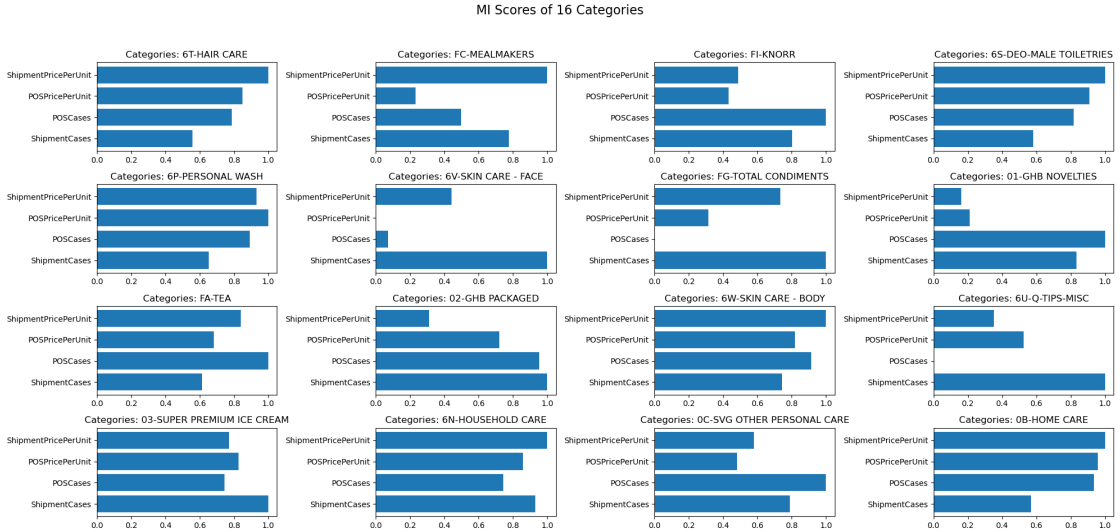
To see if additional features, such as the price per unit features, could be removed from the feature set, the mutual information scores for the four remaining features were calculated and plotted as described in Section 5.2.2. The scores for each feature of twenty five random Case UPC time series and sixteen random Category time series can be seen in Figure 6. For further analysis, the mutual information scores for all time series in Case UPC and Category datasets were averaged as discussed in Section 5.2.2. The resulting plots can be seen in Figure 7.

Discussion. Observing Figures 6a and 6b, we see that the mutual information scores of time series in both datasets are different for each product and category. This suggests that the nature of the time series, and the features that are important in forecasting the target variable, differ across most products and categories. Thus, models trained with the full Case UPC and Category datasets will need to consider all features for producing accurate forecasts across all products and categories.

Figure 7 illustrates "ShipmentCases" having the highest importance across both datasets on average. This is expected, as "ShipmentCases" is the target variable and thus should have the highest degree of importance in forecasting its own future values. "ShipmentPricePerUnit" has the second highest mutual information score on average across both datasets. This supports the hypothesis made during the initial feature selection phase that the price per unit features may have an influence on the shipment demands. The two remaining point of sales features appear to have slightly higher importance for the Category dataset, compared to the Case UPC dataset.



(a) Mutual information scores for 25 random Case UPC time series

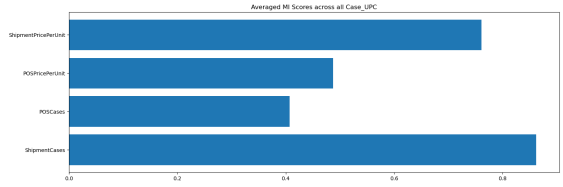


(b) Mutual information scores for 16 random Category time series

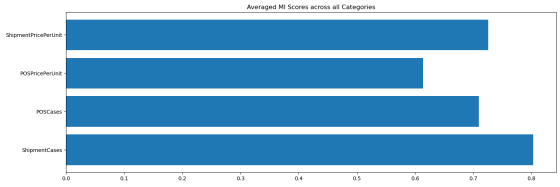
Figure 6: Mutual information scores of features for random time series in the CaseUPC and Category datasets

6.2 Training Results

The training and validation losses during the training of the proposed machine learning models are reported and discussed in this section. The results of all the models trained on specific time series are shown for two sample products and categories instead of all five. All results of time series specific models display results for case UPC 772544621080 (UPC 1) and the 6P-PERSONAL WASH category (Cat. 1) for consistency.



(a) Averaged mutual information scores across all Case UPC time series



(b) Averaged mutual information scores across all Category time series

Figure 7: Mutual information scores of features for random time series in the CaseUPC and Category datasets

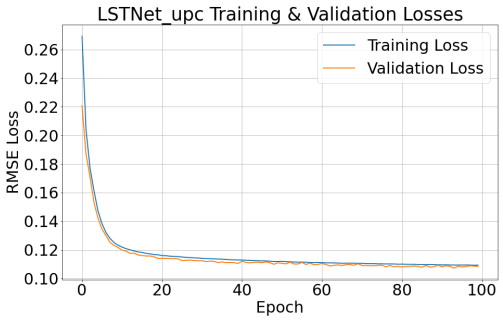
In the Discussion sections, the assumption that a model is not stuck in a saddle point or local minimum is often made when observing converging losses. As discussed in Section 5.5, models were trained using the Adam optimizer, which has been proven to help models escape saddle points faster, and converge faster overall to second-order stationary points [30]. Additionally, different hyperparameters such as the learning rate and the optimizer's weight decay were experimented with in an attempt to improve training processes. Thus, the assumption that the model will have a low probability of being stuck in a saddle point or local minimum is safe.

6.2.1 LSTNet

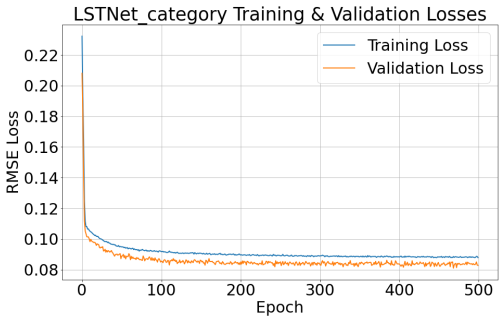
The LSTNet model was trained on both the Case UPC and Category datasets, resulting in two trained models. The training and validation losses for LSTNet are shown in Figure 8.

Discussion. Observing the loss curves of LSTNet trained on the Case UPC dataset shown in Figure 8a, we see that the training and validation losses quickly decrease and converge after roughly 40 epochs. The validation loss does not grow as the training loss converges, indicating that the model did not overfit to the data. This suggests that the training was successful, assuming the model was not stuck in a saddle point or local minimum.

Losses for LSTNet trained on the Category dataset shown in Figure 8b follows a similar curve, with the validation losses being lower than the training losses. This may be due to a validation set that is easier to forecast than the training set, or the model's inability to learn the training set at all. For now, since there was no overfitting and the losses converged, we can hypothesize that the training was successful.



(a) Training and validation losses for LSTNet trained on the Case UPC dataset



(b) Training and validation losses for LSTNet trained on the Category dataset

Figure 8: Training and validation loss curves for LSTNet trained on the Case UPC and Category dataset. The orange curve represents the validation loss and the blue curve represents the training loss

6.2.2 LSTNet-S

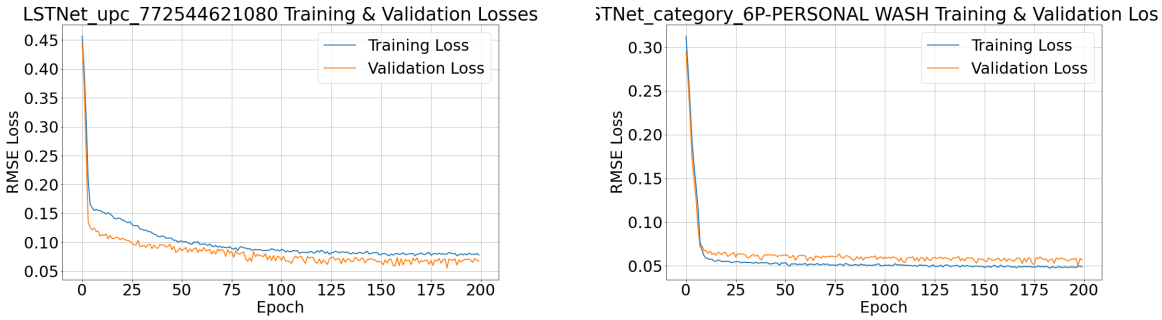
The LSTNet model was also trained on five sample time series from the Case UPC dataset and five sample time series from the Category dataset, resulting in ten trained models that are each responsible for a specific product or category. LSTNet models trained on a single time series are denoted as LSTNet-S for convenience. Two sample loss plots are displayed in Figure 9 for discussion. All ten loss plots can be found in Appendix E.

Discussion. Examining Figure 9, both validation and training losses plateau after roughly 75 epochs. This suggests that the models were successfully trained without overfitting, assuming the models were not stuck in a saddle point or local minimum. LSTnet demonstrated similar results when training on the other products and categories (see Appendix E).

6.2.3 MultiLSTNet

The MultiLSTNet model was trained on both the Case UPC and Category datasets, resulting in two trained models. The training and validation losses for MultiLSTNet are shown in Figure 10.

Discussion. The resulting loss plots show a rapid decrease in both the training and validation losses. When training on the Case UPC dataset, we see in Figure 10a that the validation and training losses are very similar for all epochs. On the contrary, when training on the Category dataset, we see in Figure 10b that the validation loss is lower than the training loss for most epochs, similar to the results of the LSTNet model. Additionally, we see the

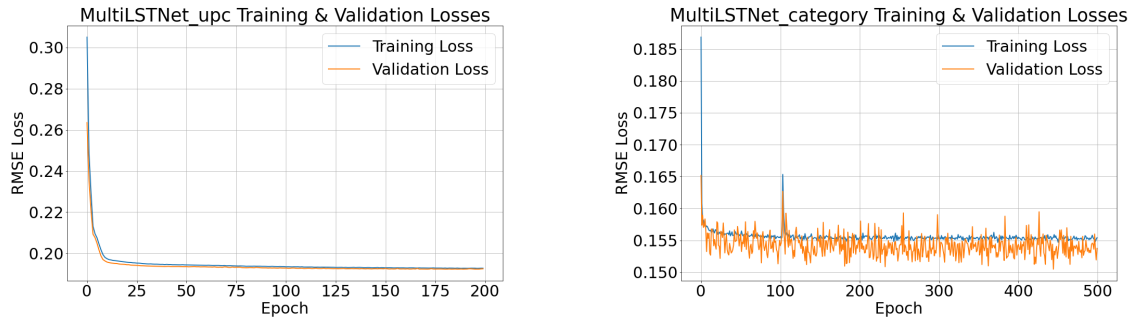


(a) Training and validation losses for LSTNet trained on case UPC 772544621080

(b) Training and validation losses for LSTNet trained on the Category 6P-PERSONAL WASH

Figure 9: Two sample training and validation loss curves for LSTNet trained on individual time series

losses erratically move between high and low values; we can hypothesize that the model was unable to learn the temporal patterns of the Category training set after the first 100 epochs, and the validation losses were dependant on the difficulty of the validation data.



(a) Training and validation losses for MultiLSTNet trained on the Case UPC dataset

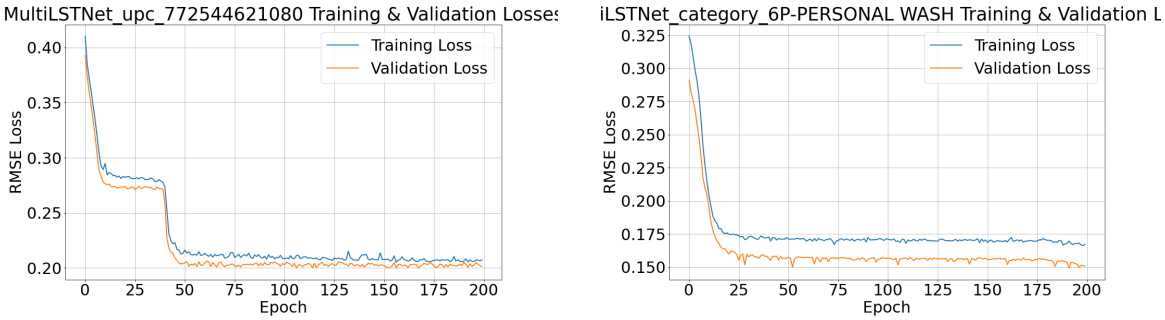
(b) Training and validation losses for MultiLSTNet trained on the Category dataset

Figure 10: Training and validation loss curves for MultiLSTNet trained on the Case UPC and Category dataset. The orange curve represents the validation loss and the blue curve represents the training loss

6.2.4 MultiLSTNet-S

The MultiLSTNet model was also trained on five sample time series from the Case UPC dataset and five sample time series from the Category dataset, resulting in ten trained models that are each responsible for a specific product or category. MultiLSTNet models trained on a single time series are denoted as MultiLSTNet-S for convenience. Two sample loss plots are displayed in Figure 11 for discussion. All ten loss plots can be found in Appendix F.

Discussion. From Figure 11, we observe that the validation and training losses quickly decrease and converge after roughly 50 epochs for both models. Assuming the models are not stuck in a saddle point or local minimum, the loss plots suggest that the models were successfully trained without overfitting. The MultiLSTNet model demonstrated similar results when training on the other products and categories (see Appendix F).



(a) Training and validation losses for MultiLSTNet trained on case UPC 772544621080

(b) Training and validation losses for MultiLSTNet trained on the Category 6P-PERSONAL WASH

Figure 11: Two sample training and validation loss curves for MultiLSTNet trained on individual time series

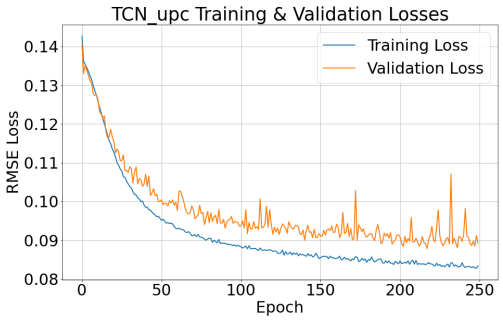
6.2.5 TCN

The TCN model was trained on both the Case UPC and Category datasets, resulting in two trained models. The training and validation losses for the TCN model are shown in Figure 12.

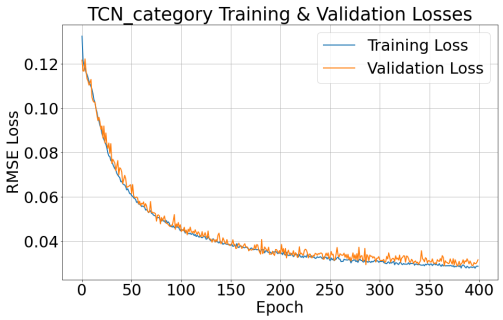
Discussion. The resulting loss curves show a steady decrease in the validation loss along with the training loss. In Figure 12a, the validation loss is always greater than the training loss and plateaus near the end of training, while the training loss continues to decrease. Similarly, we see the validation loss closely following the training loss until the very last epoch for the TCN model trained on the Category dataset, shown in Figure 12b. Thus, we can hypothesize that the models were trained successfully.

6.2.6 TCN-S

The TCN model was also trained on five sample time series from the Case UPC dataset and five sample time series from the Category dataset, resulting in ten trained models that are each responsible for a unique product or category. TCN models trained on a single time



(a) Training and validation losses for TCN trained on the Case UPC dataset

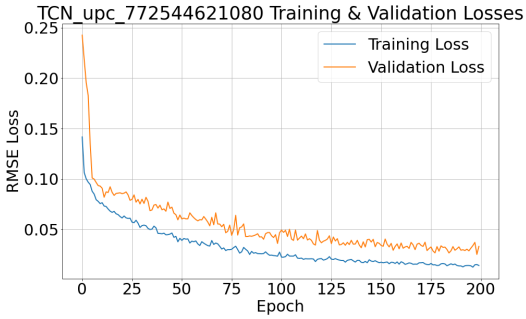


(b) Training and validation losses for TCN trained on the Category dataset

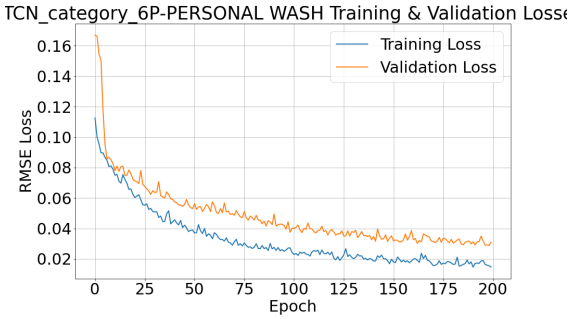
Figure 12: Training and validation loss curves for the TCN model trained on the Case UPC and Category dataset. The orange curve represents the validation loss and the blue curve represents the training loss

series are denoted as TCN-S for convenience. Two sample loss plots are displayed in Figure 13 for discussion. All ten loss plots can be found in Appendix G.

Discussion. From Figure 13, we observe that the validation losses steadily decrease with the training loss for both models. Assuming the models were not stuck in a saddle point or local minimum, the loss plots suggest that the models were successfully trained without overfitting. The TCN model demonstrated similar results when training on the other products and categories (see Appendix G).



(a) Training and validation losses for the TCN model trained on case UPC 772544621080



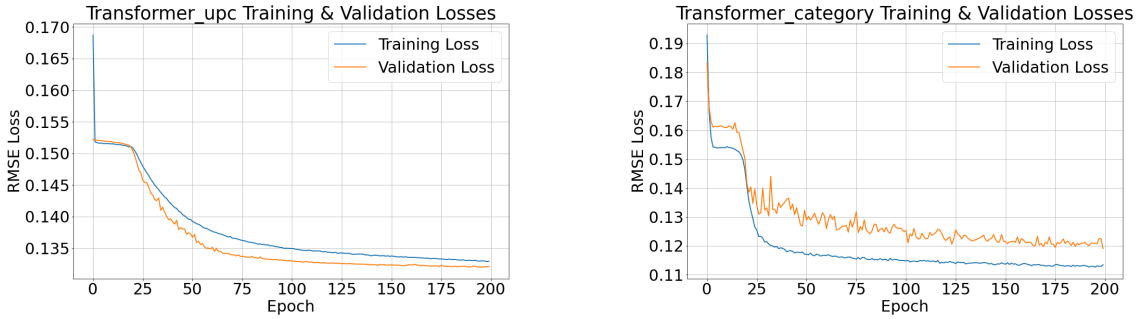
(b) Training and validation losses for the TCN model trained on the Category 6P-PERSONAL WASH

Figure 13: Two sample training and validation loss curves for the TCN model trained on individual time series

6.2.7 Transformer

The Transformer model was trained on both the Case UPC and Category datasets, resulting in two trained models. The training and validation losses for the Transformer model are shown in Figure 14.

Discussion. Both models show a steep decrease in validation loss and a short plateau for roughly 25 epochs, until gradually decreasing as the training progresses. The validation loss is greater than the training loss over all epochs for UPC 1, and is the reverse for Cat. 1 as shown in Figures 14a and 14b. The plateau suggests that the model was stuck for the 25 epochs, but was able to escape the local minimum. Since we see both the validation and training losses decrease and the validation loss plateauing, we can hypothesize that the models were successfully trained.



(a) Training and validation losses for the Transformer model trained on the Case UPC dataset

(b) Training and validation losses for the Transformer model trained on the Category dataset

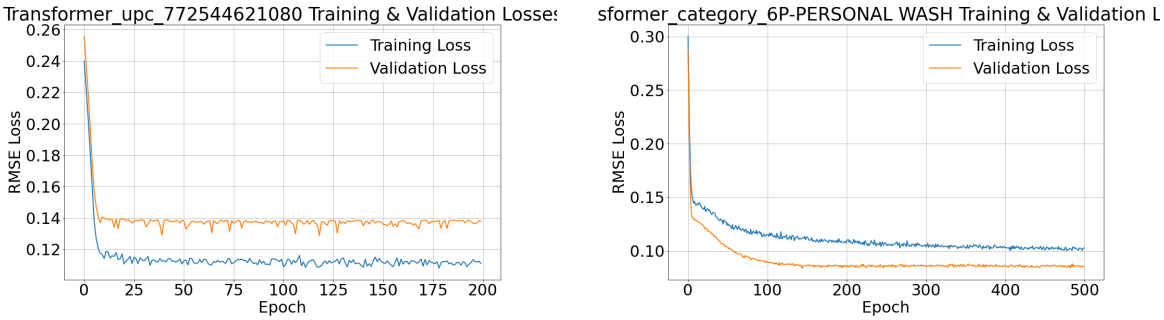
Figure 14: Training and validation loss curves for the Transformer model trained on the Case UPC and Category dataset. The orange curve represents the validation loss and the blue curve represents the training loss

6.2.8 Transformer-S

The Transformer model was also trained on five sample time series from the Case UPC dataset and five sample time series from the Category dataset, resulting in ten trained models that are each responsible for a unique product or category. Transformer models trained on specific time series are denoted as Transformer-S for convenience. Two sample loss plots are displayed in Figure 15 for discussion. All ten loss plots can be found in Appendix H.

Discussion. Figure 15a shows a steep decline over the first 10 epochs for the validation and training losses for UPC 1 , and plateaus for the rest of the training. Similarly, the losses

for the Cat. 1 decreases gradually for the first 100 epochs until the losses plateau. Assuming the models were not stuck in a saddle point or local minimum, we can conclude that the model was successfully trained.



(a) Training and validation losses for the Transformer model trained on case UPC 772544621080

(b) Training and validation losses for the Transformer model trained on the Category 6P-PERSONAL WASH

Figure 15: Two sample training and validation loss curves for the Transformer model trained on individual time series

6.3 Evaluation Results

Table 4 summarizes the evaluation results for all methods on all the test sets on all metrics. The median RMSE, accuracy, and bias measures are reported for test sets consisting of multiple predictions to remove extreme outliers as shown discussed in the following section. Models that have been trained on a single time series of a product or category is denoted as $\langle model\_name \rangle$-S for simplicity. Case UPCs and category descriptions were abbreviated for convenience; a summary of the abbreviations are shown in Table 3. In addition, all models' 12 week forecasts on the UPC 1 and the Cat. 1 time series are shown in Figures 16 and 17 for reference. All 12 week forecasts made by all models on the 5 products and 5 categories can be found in the Appendix. Sample histograms as described in Section 5.6 are shown in Figure 18 for discussion as well.

Discussion. The best result for each data and metric pair is bolded in Table 4. The LSTNet models have 22 bolded results (includes LSTNet-S), the Transformer models have 8, the MultiLSTNet models have 3, the VAR model has 2, and the TCN models have 1 bolded result. From this, we see that the LSTNet model outperformed all other proposed methods on all metrics, including the TCN and Transformer models which are state-of-art and have been shown to outperform recurrent neural networks in literature. In addition, we

Abbreviation	Time Series
UPC 1	UPC: 772544621080
UPC 2	UPC: 772212846677
UPC 3	UPC: 774617779751
UPC 4	UPC: 772544680012
UPC 5	UPC: 772544621075
Cat. 1	Category: 6P-PERSONAL WASH
Cat. 2	Category: 6W-SKIN CARE - BODY
Cat. 3	Category: 6T-HAIR CARE
Cat. 4	Category: 6S-DEO-MALE TOILETRIES
Cat. 5	Category: FI-KNORR

Table 3: Abbreviations of the sample time series

observe that the models generally perform better on the Category dataset, with accuracies ranging between roughly 60% to 75%, while the accuracies for the Case UPC dataset were below 30% on average. Furthermore, models performed better training on the entire Case UPC or Category dataset compared to models trained on a single time series as shown in 4. A final observation is that accuracies differ drastically for each time series as shown in the example accuracy histogram of LSTNet in Figure 18. Figure 18a shows accuracies ranging from -350% to 72.5%.

A few speculations can be made about LSTNet's success on Unilever's datasets. First, one major difference with the LSTNet model is that it performs iterated forecasts as outlined in Section 5.3.2, whereas all other proposed models were trained to produce direct forecasts. Iterated forecasts are less robust to model misspecification compared to direct forecasts [16], and errors and biases accumulate as the forecasting horizon increases. Thus, one could assume that direct forecasts would yield better results when forecasting volatile time series with a large forecasting horizon. However, similar to the surprising empirical results in [16] which showed better performances in iterative forecasting over direct forecasting for longer prediction horizons, the LSTNet outperformed all direct forecasting methods in this paper.

Additionally, TCNs have been considered to be the current state-of-art method for sequence modelling, yet yielded the worst results in this research. For further investigation on LSTNet's success and TCN's poor performance, let's compare the 12 week forecasts made by LSTNet and the TCN model for the UPC 1 time series in Figures 16a and 16e. Forecasts by LSTNet is smooth and captures the trend of the time series well. TCN's forecasts appear to follow the nature of the time series more, and attempts to predict the peaks and dips. The TCN model is also able to capture the trend of the time series successfully. However, the RMSE of the forecasts are 0.1216 for the TCN model, and 0.1047 for LSTNet as seen

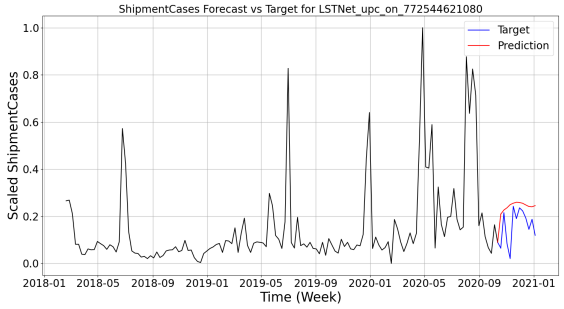
in Table 4. Observing other forecasts made by LSTNet and TCN in Figures 16 and 17, we see that the model generates smooth forecasts that do not deviate much from the input sequence, whereas forecasts made by TCN are more jagged with peaks and dips. Thus, the TCN model is inclined to accumulate much higher errors than the LSTNet model when predicted peaks and dips do not align with the target values. This may be the reason for TCN's poor performance in this research.

Another result was that forecasting on the category level yielded better results compared to forecasting on the product level. This is as expected, as time series at the most granular level would have more variance and local trends that are specific to a product. Aggregating the time series would smooth out local variances and trends to capture larger trends and temporal patterns that models can learn more easily. This is demonstrated in the high accuracy results for forecasts on the Category level, with more than a 30% to 40% difference in accuracy compared to the forecasts made on the Case UPC dataset.

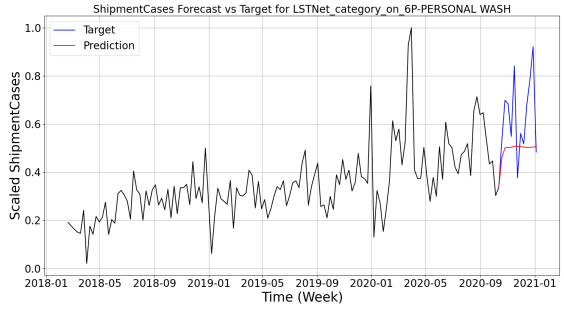
Lastly, models trained on the entire Category or CaseUPC datasets yielded better results compared to models trained on single time series, as expected. The proposed models are deep learning methods that perform the best when ample data is provided, which consists of a variety of different temporal characteristics that can be learned. Thus with more input-target pairs to learn from, the models are able to learn more patterns and trends that can be useful in forecasting future values.

Dataset		Case UPC						Category					
		Time Series						Time Series					
Models	Metrics	All	UPC 1	UPC 2	UPC 3	UPC 4	UPC 5	All	Cat. 1	Cat. 2	Cat. 3	Cat. 4	Cat. 5
LSTNet (18)	RMSE	0.1447	0.1047	0.0685	0.2317	0.1531	**0.0959**	0.1337	**0.1333**	**0.1410**	**0.1289**	**0.1500**	0.1554
	ACC.	**0.2546**	-0.5156	**0.7253**	-1.2082	-0.5732	0.0440	**0.6650**	**0.7704**	**0.7659**	**0.7503**	0.7287	0.6885
	BIAS	0.4953	**0.3493**	**0.2628**	0.9858	**0.4932**	**0.3818**	**0.3174**	**0.3202**	**0.3040**	0.2504	0.3145	0.4058
MultiLSTNet (1)	RMSE	0.2093	0.1595	0.1460	**0.2255**	0.1701	0.1355	0.1846	0.1855	0.1837	0.1462	0.2047	0.1904
	ACC.	-0.3007	-0.8683	-0.0388	-1.3309	-1.5419	-0.3306	0.5920	0.7084	0.7435	0.7502	0.6872	0.4283
	BIAS	0.5132	0.4114	0.4506	0.9321	0.4976	0.3854	0.3410	0.4648	0.3673	0.2842	0.4581	0.4038
TCN (0)	RMSE	0.1389	0.1216	0.1149	0.3243	0.1822	0.2107	0.1552	0.3392	0.2503	0.2106	0.3772	0.1631
	ACC.	0.1878	-0.4540	0.3627	-2.0618	-1.9470	-0.7934	0.5585	0.5605	0.6537	0.7101	0.5167	0.5870
	BIAS	0.6345	0.3738	0.4514	1.0976	0.7229	0.5469	0.3905	1.0576	0.6118	0.5004	0.9942	0.8514
Transformer (4)	RMSE	**0.1075**	0.1561	0.1027	0.2858	0.1839	0.1589	**0.1279**	0.2324	0.1972	0.1869	0.2330	**0.1308**
	ACC.	0.2380	-1.1221	0.2265	-1.5036	-1.9103	-0.6489	0.5272	0.7499	0.7406	0.7393	0.7225	0.6177
	BIAS	0.5363	0.4564	0.3786	**0.8451**	0.5421	0.4053	0.3400	0.3998	0.3693	0.3747	0.3890	0.4576
VAR-S (2)	RMSE	0.1322	0.22s00	0.0964	0.2995	0.1090	0.1323	0.3022	0.6479	0.5420	0.5522	0.6687	0.1364
	ACC.	0.0344	-0.3720	0.3782	-1.2401	0.2922	**0.4095**	0.0203	0.0151	0.0255	0.0080	0.0090	0.6685
	BIAS	**0.4874**	-0.5014	0.3173	0.9631	1.3360	0.9373	-1.5713	6.4719	-39.4518	-17.1201	-96.9217	0.4938
LSTNet-S (4)	RMSE	-	0.1613	0.1422	0.2521	0.1754	0.1958	-	0.2322	0.1537	0.1875	0.1527	0.1396
	ACC.	-	**0.2989**	0.4274	-1.3020	**0.3761**	0.3545	-	0.6750	0.7320	0.6148	**0.7357**	**0.7188**
	BIAS	-	1.2072	0.3304	0.9292	1.3568	1.6910	-	0.5523	0.3934	0.2468	0.2796	0.5494
MultiLSTNet-S (2)	RMSE	-	0.1593	0.1718	0.2258	0.1697	0.1406	-	0.1831	0.1775	0.1449	0.1959	0.1883
	ACC.	-	-0.8654	-0.0364	-1.3404	-1.5386	-0.3251	-	0.7633	0.7504	0.7408	0.7202	0.4295
	BIAS	-	0.4110	0.4500	0.9256	0.4972	0.3856	-	0.3472	0.3553	**0.2227**	0.3544	**0.4035**
TCN-S (1)	RMSE	-	0.1429	0.1271	0.2831	0.1353	0.1468	-	0.2126	0.2126	0.1684	0.1792	0.1529
	ACC.	-	-0.8961	0.3471	-1.3595	-0.2483	0.3457	-	0.7134	0.7502	0.6987	0.7299	0.6079
	BIAS	-	0.5796	0.3403	0.9236	0.6992	1.1802	-	0.4042	0.3543	0.3056	**0.2728**	-0.3493
Transformer-S (4)	RMSE	-	**0.0927**	**0.0605**	0.3143	**0.1008**	0.1372	-	0.3417	0.1916	0.1858	0.2973	0.1482
	ACC.	-	0.2724	0.5564	**-1.0279**	0.3398	0.3824	-	0.5521	0.7544	0.7331	0.6480	0.6930
	BIAS	-	0.8256	0.2812	1.1106	0.8510	1.0316	-	0.9235	0.3412	0.3745	0.6135	0.6306

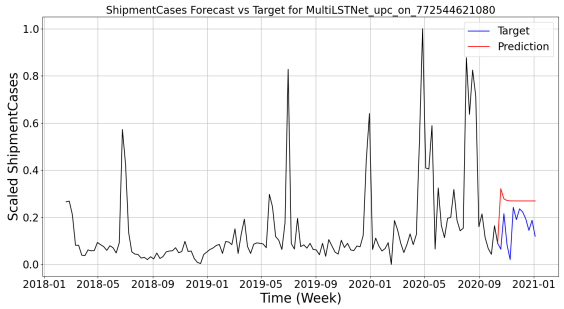
Table 4: Results summary of all methods across all test sets and on five random time series from the Case UPC and Category datasets: 1) each row consists of the results for a given method in a particular metric for the different time series; 2) bold numbers indicate the best result of each column for a particular metric; 3) the total number of bolded results of each model is listed next to the model's name in parentheses



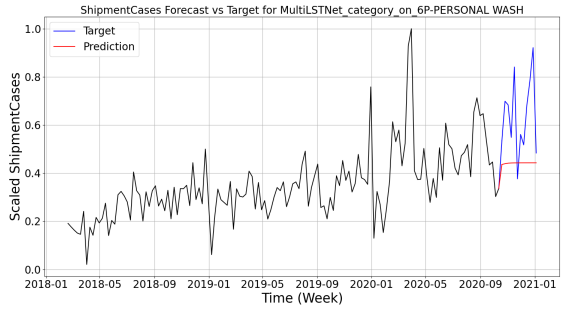
(a) LSTNet 12 week forecast for UPC 1



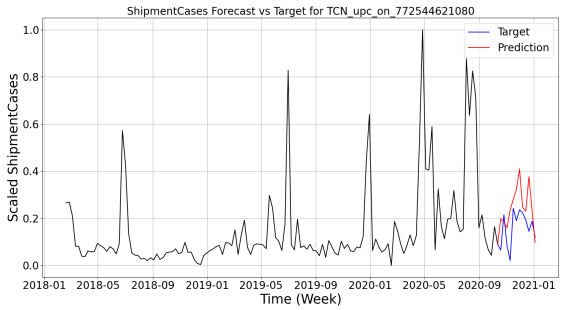
(b) LSTNet 12 week forecast for Cat. 1



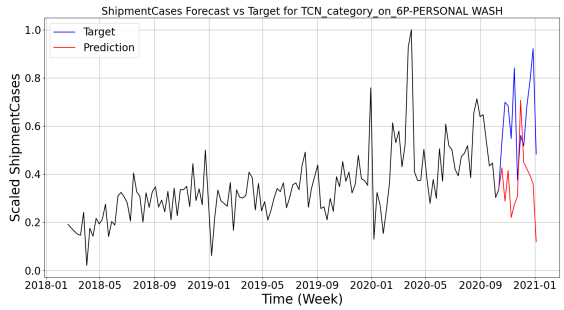
(c) MultiLSTNet 12 week forecast for UPC 1



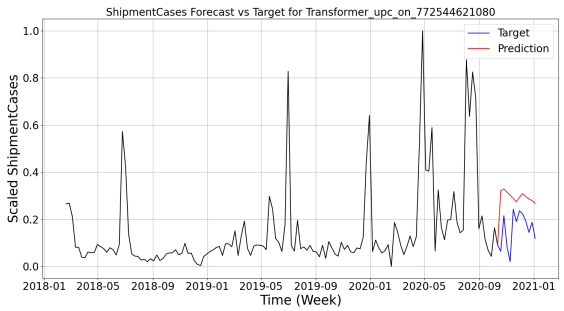
(d) MultiLSTNet 12 week forecast for Cat. 1



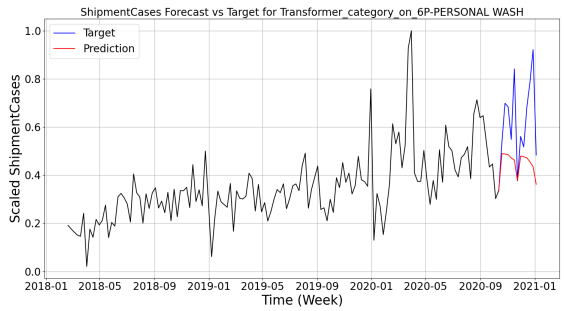
(e) TCN 12 week forecast for UPC 1



(f) TCN 12 week forecast for Cat. 1

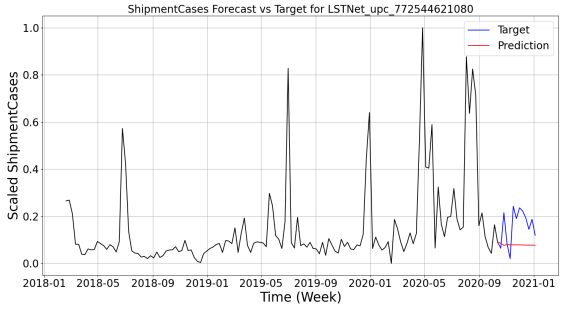


(g) Transformer 12 week forecast for UPC 1

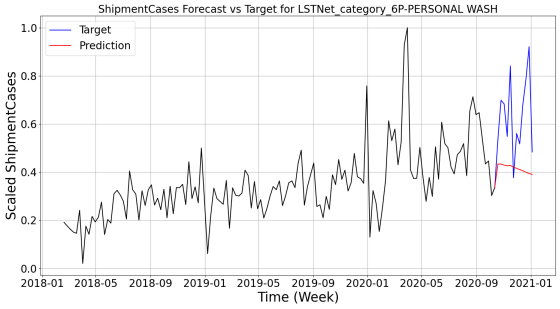


(h) Transformer 12 week forecast for Cat. 1

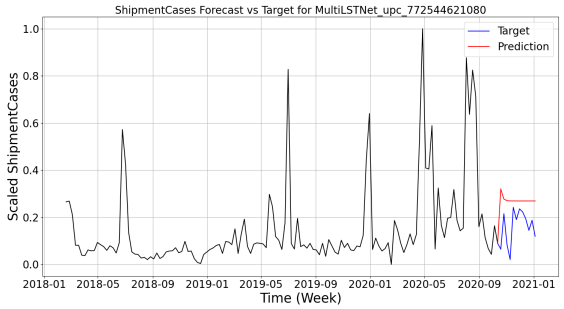
Figure 16: 12 week forecasts of UPC 1 and Cat.1 by all models trained on the entire Case UPC or Category dataset



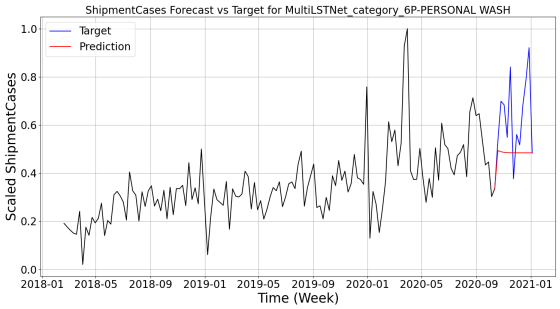
(a) LSTNet-S 12 week forecast for UPC 1



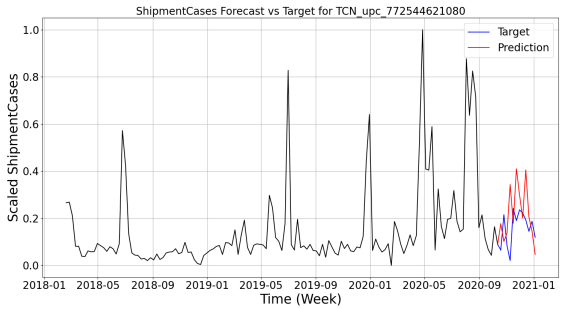
(b) LSTNet-S 12 week forecast for Cat. 1



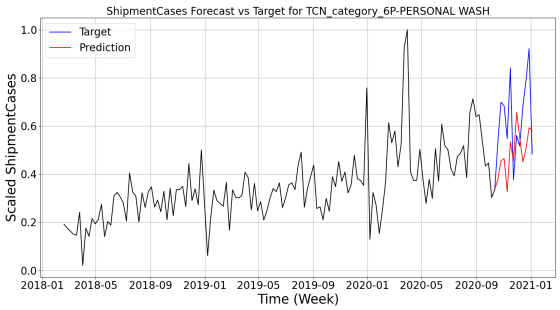
(c) MultiLSTNet-S 12 week forecast for UPC 1



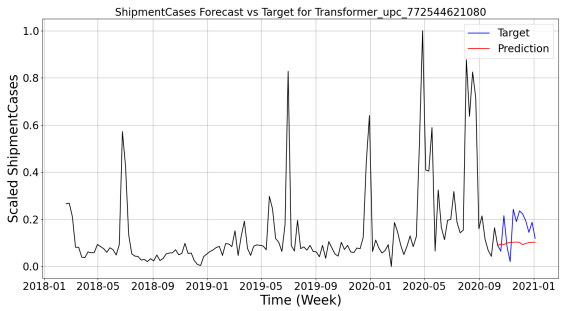
(d) MultiLSTNet-S 12 week forecast for Cat. 1



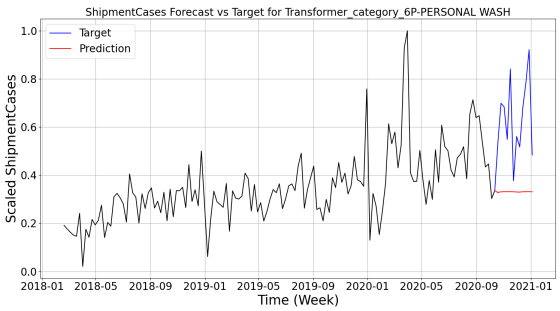
(e) TCN-S 12 week forecast for UPC 1



(f) TCN-S 12 week forecast for Cat. 1

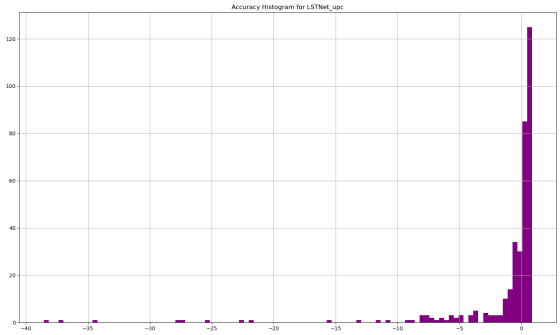


(g) Transformer-S 12 week forecast for UPC 1

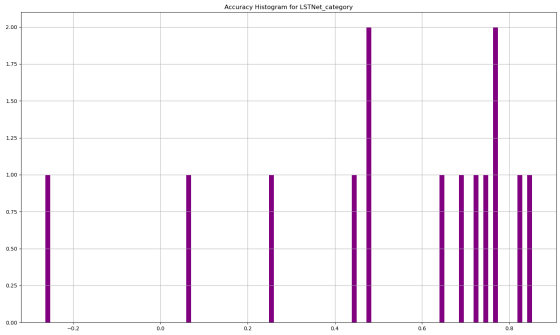


(h) Transformer-S 12 week forecast for Cat. 1

Figure 17: 12 week forecasts of UPC 1 and Cat.1 for all machine learning models trained on the UPC 1 or Cat. 1 time series



(a) Test accuracy histogram for the LSTNet model trained on the Case UPC dataset



(b) Test accuracy histogram for the LSTNet model trained on the Category test set

Figure 18: Test accuracy histograms for the LSTNet models

7 Conclusion

This paper presented an evaluation of Unilever's data and five multivariate forecasting methods on predicting Unilever's shipment demands. There are several key findings in this research that are significant. Firstly, analysis of Unilever's data showed that multiple features were linearly dependant, or scaled values of each other. This allowed for large reductions in the feature space, which has been shown to accelerate loss convergence during training and improve prediction results [19]. Secondly, mutual information of features showed that features' importances on the target variable varied heavy between time series in all datasets. This demonstrates that the time series do not share the same dependencies between variables across products or categories. Thirdly, the models have shown to generate better forecasts on the Category dataset compared to the Case UPC dataset, and also performed better when trained on the entire dataset rather than on a single product or category. Lastly, LSTNet, a convolutional and recurrent neural network proposed by Lai et al., was shown to perform the best in producing 12 week forecasts on Unilever's datasets when compared to the VAR, MultiLSTNet, TCN, and Transformer models in 22 evaluation scenarios out of 36. Forecasts produced by LSTNet have shown to be steady, and fit to the trend of the data well. Unilever now has more information about their shipment and point-of-sales data, as well as performance evaluations of novel methods in modelling their data. These results serve as valuable information to Unilever that can be used in developing and integrating novel methods to be utilized in their supply chain management processes.

There are multiple promising directions that Unilever or other researchers can take in extending this research. Firstly, with enough computing resources, hyperparameters of each of the models can most likely be improved via well known methods such as grid search or Bayesian Optimization [28]. Additionally, product or category specific models can be trained for all products and categories to better evaluate the performance of specialized models trained on single time series. This may also help determine which products and categories are outliers that need special attention to be forecasted.

Since models have shown to perform better when trained on entire datasets, and since time series have shown to have differing dependencies between variables, models' performances may be improved by clustering similar time series together and training on the clustered datasets. This would increase the training set for models, while decreasing the variance and noise between time series in the training set. Additionally, since the results show that models perform better on an aggregated dataset, i.e. the Category dataset, models' performances on the product level may be improved by training models on aggregated time series of products that share similarities in it trend and temporal patterns, and be used

to generate forecasts for time series that were included in the aggregated dataset.

References

[1] Mohammad Assaad, Romuald Boné, and Hubert Cardot. A new boosting algorithm for improved time-series forecasting with recurrent neural networks. *Information Fusion*, 9:41–55, 01 2008.

[2] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.

[3] George.E.P. Box and Gwilym M. Jenkins. *Time Series Analysis: Forecasting and Control*. Holden-Day, 1976.

[4] Michael Clements, Philip Franses, and Norman Swanson. Forecasting economic and financial time-series with non-linear models. *International Journal of Forecasting*, 20:169–183, 04 2004.

[5] Francesco Cordoni. A comparison of modern deep neural network architectures for energy spot price forecasting. *Digital Finance*, 2:1–22, 12 2020.

[6] Petrônio Cândido de Lima e Silva, Carlos Alberto Severiano, Marcos Antonio Alves, Rodrigo Silva, Miri Weiss Cohen, and Frederico Gadelha Guimarães. Forecasting in non-stationary environments with fuzzy time series. *Applied Soft Computing*, 97:106825, 2020.

[7] Michael Auli Denis Yarats, Jonas Gehring. A novel approach to neural machine translation.

[8] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9:1735–80, 12 1997.

[9] Qinzhong Hou, Junqiang Leng, Guosheng Ma, Weiyi Liu, and Yuxing Cheng. An adaptive hybrid model for short-term urban traffic flow prediction. *Physica A: Statistical Mechanics and its Applications*, 527:121065, 04 2019.

[10] Tomoharu Iwata and Atsutoshi Kumagai. Few-shot learning for time-series forecasting, 2020.

[11] Dennis W. Jansen. *Southern Economic Journal*, 61(4):1241–1243, 1995.

[12] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[13] Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long-and short-term temporal patterns with deep neural networks. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 95–104, 2018.

[14] N. Laptev, J. Yosinski, L. Li, and Slawek Smyl. Time-series extreme event forecasting with neural networks at uber. 2017.

[15] Assimakopoulos V. Makridakis S., Spiliotis E. Statistical and machine learning forecasting methods: Concerns and ways forward. *PLoS ONE, 13*, 3.

[16] Massimiliano Marcellino, James H Stock, and Mark W Watson. A comparison of direct and iterated multistep ar methods for forecasting macroeconomic time series. *Journal of econometrics*, 135(1-2):499–526, 2006.

[17] Katta Murty. Forecasting for supply chain and portfolio management. 01 2006.

[18] Rizwan Mushtaq. Augmented dickey fuller test. 2011.

[19] Boaz Nadler and Ronald R Coifman. The prediction error in cls and pls: the importance of feature selection prior to multivariate calibration. *Journal of Chemometrics: A Journal of the Chemometrics Society*, 19(2):107–118, 2005.

[20] Robert Nau. Statistical forecasting: notes on regression and time series analysis.

[21] Olalekan P. Ogunmolu, Xuejun Gu, Steve B. Jiang, and Nicholas R. Gans. Nonlinear systems identification using deep dynamic neural networks. *CoRR*, abs/1610.01439, 2016.

[22] S Patro and Kishore Kumar Sahu. Normalization: A preprocessing stage. *arXiv preprint arXiv:1503.06462*, 2015.

[23] Preeti, R. Bala, and R. P. Singh. Financial and non-stationary time series forecasting using lstm recurrent neural network for short and long horizon. In *2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, pages 1–7, 2019.

[24] Sushil Punia, Surya P. Singh, and Jitendra K. Madaan. A cross-temporal hierarchical framework and deep learning for supply chain forecasting. *Computers & Industrial Engineering*, 149:106796, 2020.

[25] Chongchong Qi, Li Guo, Hai-Bang Ly, Hiep Van Le, and Binh Thai Pham. Improving pressure drops estimation of fresh cemented paste backfill slurry using a hybrid machine learning method. *Minerals Engineering*, 163:106790, 2021.

[26] Mohammad Sabah, Mohammad Mehrad, Seyed Babak Ashrafi, David A. Wood, and Shadi Fathi. Hybrid machine learning algorithms to enhance lost-circulation prediction and management in the marun oil field. *Journal of Petroleum Science and Engineering*, 198:108125, 2021.

[27] Brian B Schultz. Levene's test for relative variation. *Systematic Zoology*, 34(4):449–456, 1985.

[28] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. *arXiv preprint arXiv:1206.2944*, 2012.

[29] Jorge Sola and Joaquin Sevilla. Importance of input data normalization for the application of neural networks to complex industrial problems. *IEEE Transactions on nuclear science*, 44(3):1464–1468, 1997.

[30] Matthew Staib, Sashank Reddi, Satyen Kale, Sanjiv Kumar, and Suvrit Sra. Escaping saddle points with adaptive gradient methods. In *International Conference on Machine Learning*, pages 5956–5965. PMLR, 2019.

[31] Ralf Steuer, Jürgen Kurths, Carsten O Daub, Janko Weise, and Joachim Selbig. The mutual information: detecting and evaluating dependencies between variables. *Bioinformatics*, 18(suppl_2):S231–S240, 2002.

[32] Ahmed Tealab. Time series forecasting using artificial neural networks methodologies: A systematic review. *Future Computing and Informatics Journal*, 3(2):334 – 340, 2018.

[33] Muhammed Kürşad Uçar, Zeliha Uçar, Fatih Köksal, and Nihat Daldal. Estimation of body fat percentage using hybrid machine learning algorithms. *Measurement*, 167:108173, 2021.

[34] M. R. Vargas, C. E. M. dos Anjos, G. L. G. Bichara, and A. G. Evsukoff. Deep leaming for stock market prediction using technical indicators and financial news articles. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2018.

[35] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017.

[36] Oriol Vinyals, Charles Blundell, Timothy P. Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. *CoRR*, abs/1606.04080, 2016.

[37] S. Prince W. Zi, L. S. Ghoraie. Few-shot learning and meta-learning.

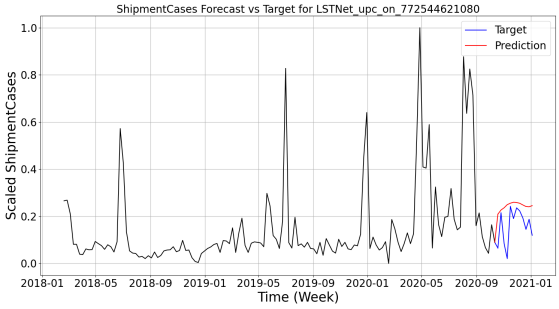
[38] Yunxiao Wang, Zheng Liu, Di Hu, and Mian Zhang. Multivariate time series prediction based on optimized temporal convolutional networks with stacked auto-encoders. In *Asian Conference on Machine Learning*, pages 157–172. PMLR, 2019.

[39] Zhenyu Yuan, Yuxin Jiang, Jingjing Li, and Handong Huang. Hybrid-dnns: Hybrid deep neural networks for mixed inputs, 2020.

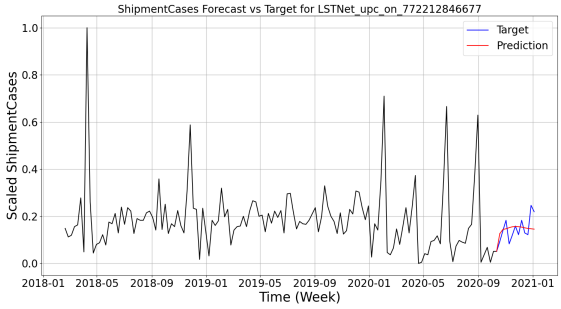
[40] Miodrag Zivkovic, Nebojsa Bacanin, K. Venkatachalam, Anand Nayyar, Aleksandar Djordjevic, Ivana Strumberger, and Fadi Al-Turjman. Covid-19 cases prediction by using hybrid machine learning and beetle antennae search approach. *Sustainable Cities and Society*, 66:102669, 2021.

A 12 Step Forecast Plots for LSTNet

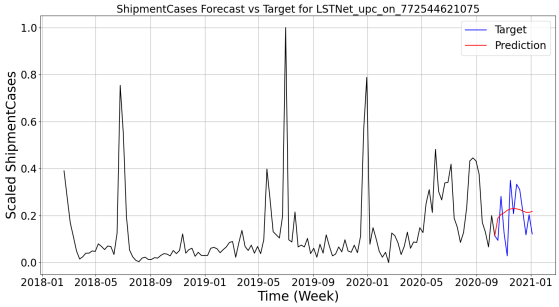
A.1 LSTNet on case UPCs



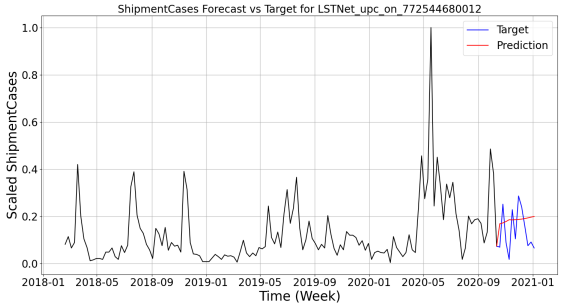
(a) LSTNet 12 week forecast for case UPC 772544621080



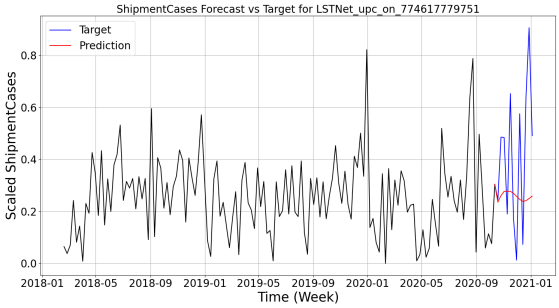
(b) LSTNet 12 week forecast for case UPC 772212846677



(c) LSTNet 12 week forecast for case UPC 772544621075



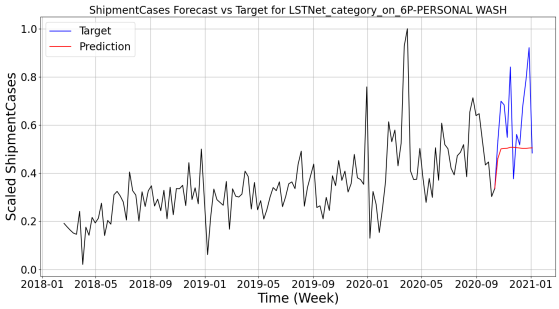
(d) LSTNet 12 week forecast for case UPC 772212846677



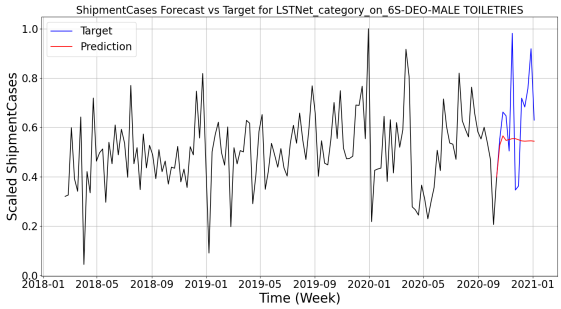
(e) LSTNet 12 week forecast for case UPC 772544621075

Figure 19: LSTNet 12 week forecasts on 5 case UPC time series

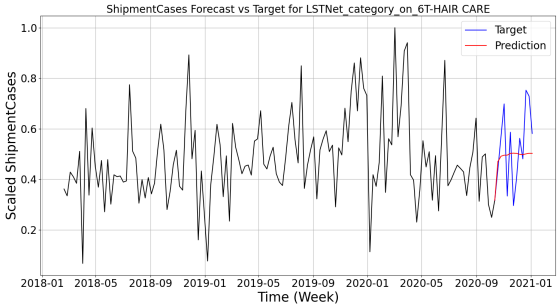
A.2 LSTNet on categories



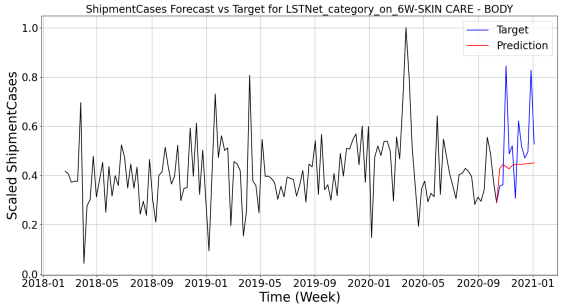
(a) LSTNet 12 week forecast for category 6P-PERSONAL WASH



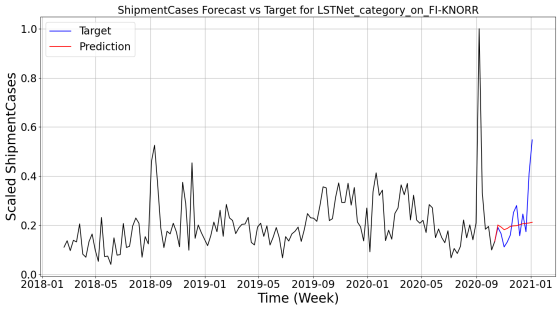
(b) LSTNet 12 week forecast for category 6S-DEO-MALE TOILETRIES



(c) LSTNet 12 week forecast for category 6T-HAIR CARE



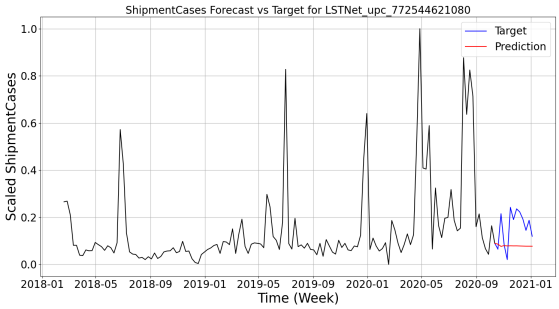
(d) LSTNet 12 week forecast for category 6W-SKIN CARE - BODY



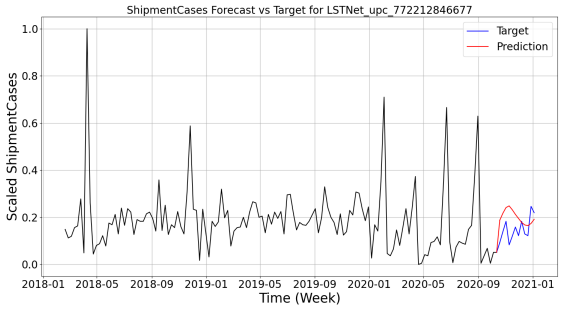
(e) LSTNet 12 week forecast for category FI-KNORR

Figure 20: LSTNet 12 week forecasts on 5 category time series

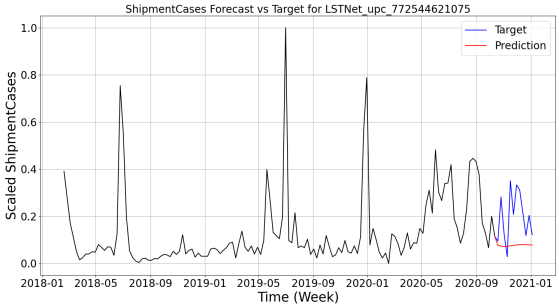
A.3 LSTNet-S on case UPCs



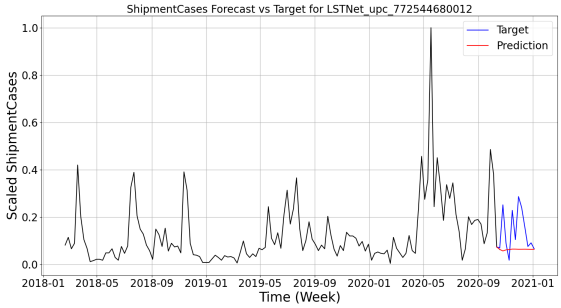
(a) LSTNet-S 12 week forecast for case UPC 772544621080



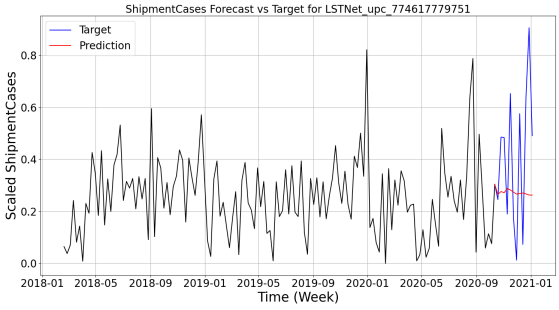
(b) LSTNet-S 12 week forecast for case UPC 772212846677



(c) LSTNet-S 12 week forecast for case UPC 772544621075



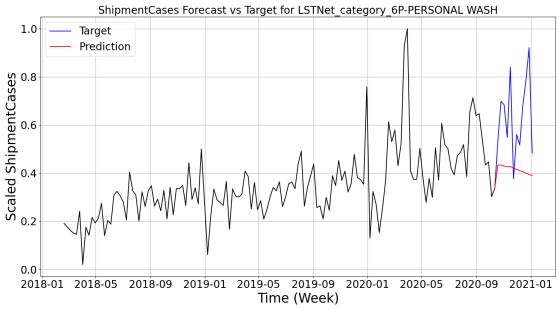
(d) LSTNet-S 12 week forecast for case UPC 772212846677



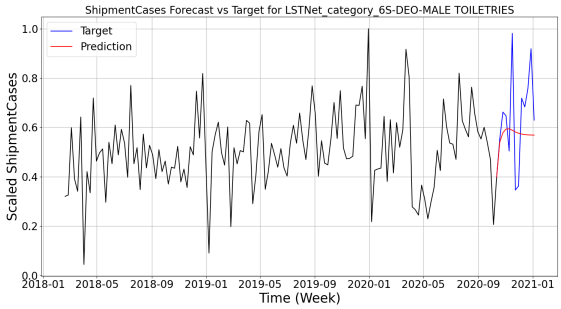
(e) LSTNet-S 12 week forecast for case UPC 772544621075

Figure 21: LSTNet-S 12 week forecasts on 5 case UPC time series

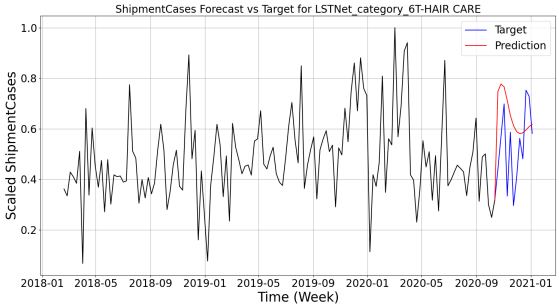
A.4 LSTNet-S on categories



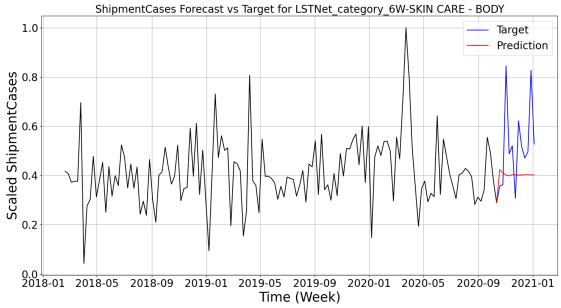
(a) LSTNet-S 12 week forecast for category 6P-PERSONAL WASH



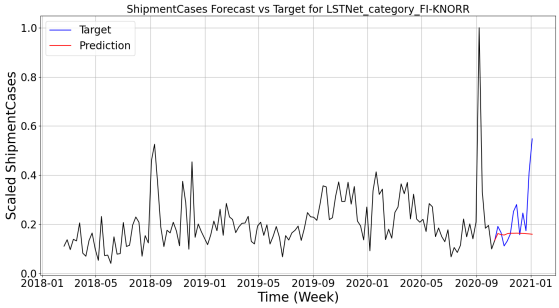
(b) LSTNet-S 12 week forecast for category 6S-DEO-MALE TOILETRIES



(c) LSTNet-S 12 week forecast for category 6T-HAIR CARE



(d) LSTNet-S 12 week forecast for category 6W-SKIN CARE - BODY

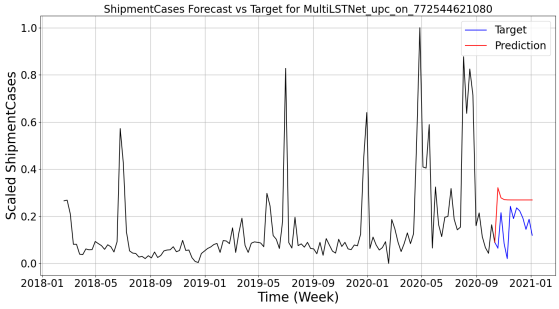


(e) LSTNet-S 12 week forecast for category FI-KNORR

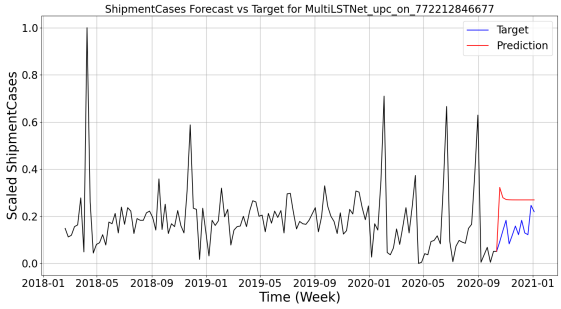
Figure 22: LSTNet-S 12 week forecasts on 5 category time series

B 12 Step Forecast Plots for MultiLSTNet

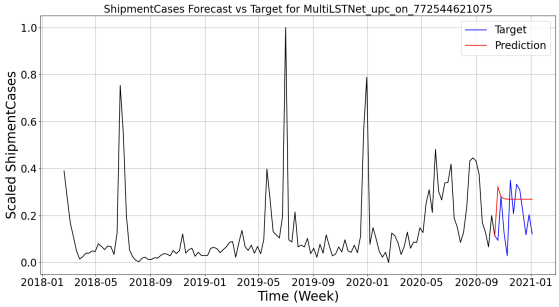
B.1 MultiLSTNet on case UPCs



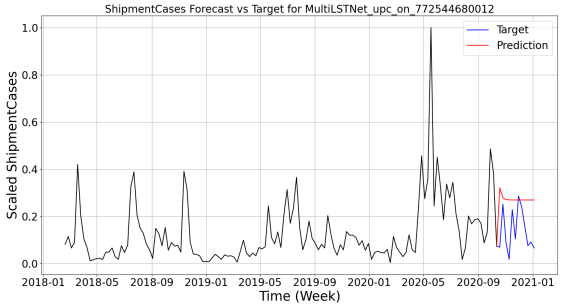
(a) MultiLSTNet 12 week forecast for case UPC 772544621080



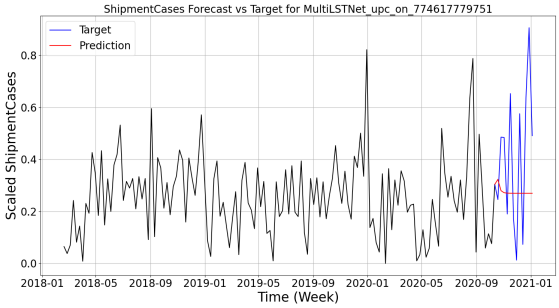
(b) MultiLSTNet 12 week forecast for case UPC 772212846677



(c) MultiLSTNet 12 week forecast for case UPC 772544621075



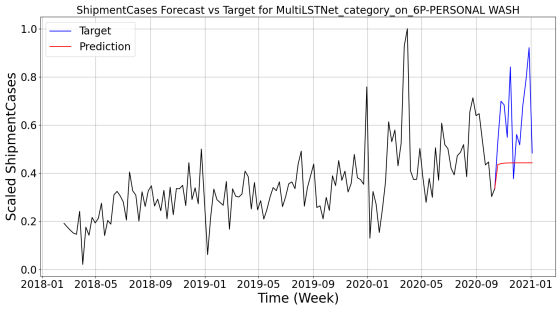
(d) MultiLSTNet 12 week forecast for case UPC 772212846677



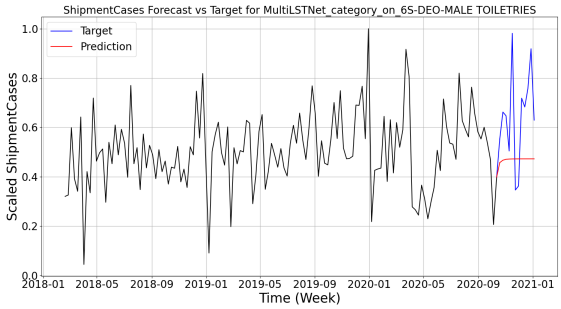
(e) MultiLSTNet 12 week forecast for case UPC 772544621075

Figure 23: MultiLSTNet 12 week forecasts on 5 case UPC time series

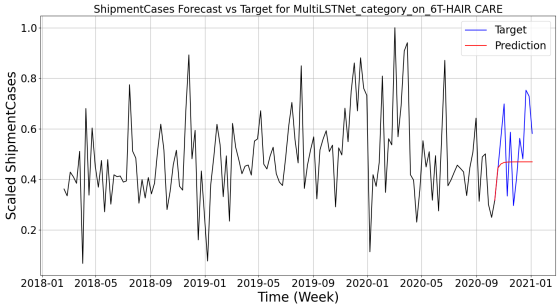
B.2 MultiLSTNet on categories



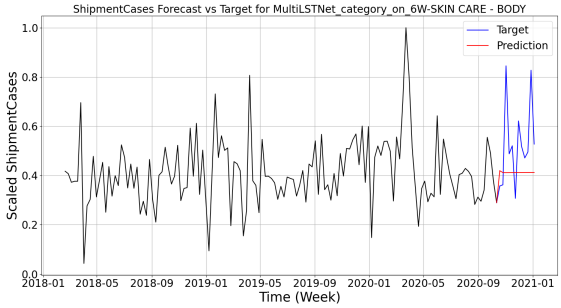
(a) MultiLSTNet 12 week forecast for category 6P-PERSONAL WASH



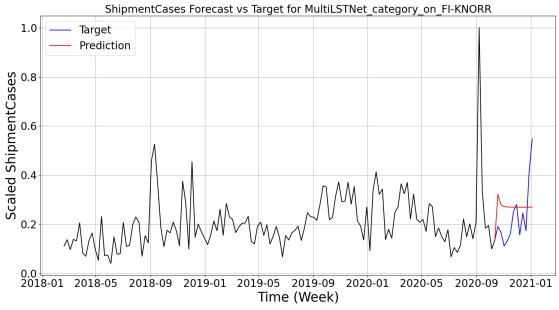
(b) MultiLSTNet 12 week forecast for category 6S-DEO-MALE TOILETRIES



(c) MultiLSTNet 12 week forecast for category 6T-HAIR CARE



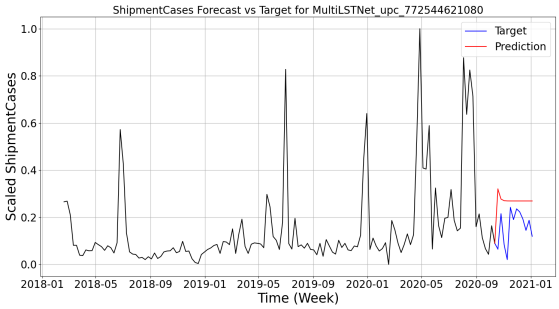
(d) MultiLSTNet 12 week forecast for category 6W-SKIN CARE - BODY



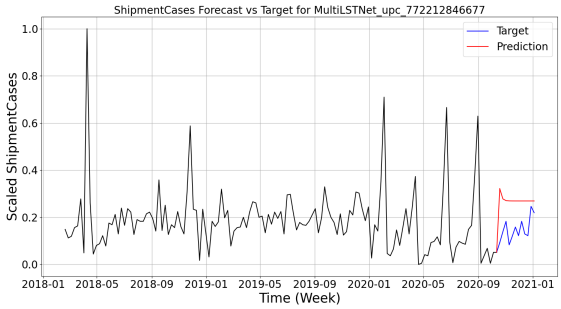
(e) MultiLSTNet 12 week forecast for category FI-KNORR

Figure 24: MultiLSTNet 12 week forecasts on 5 category time series

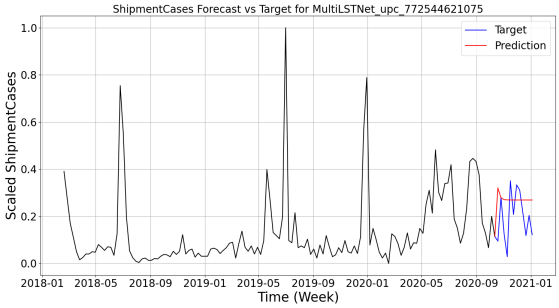
B.3 MultiLSTNet-S on case UPCs



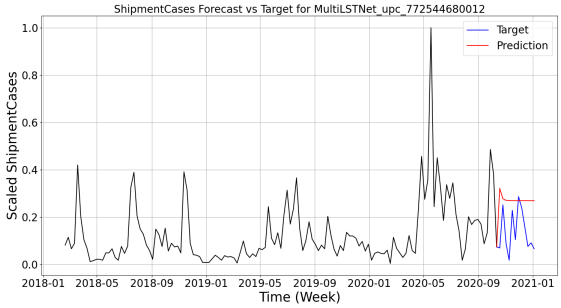
(a) MultiLSTNet-S 12 week forecast for case UPC 772544621080



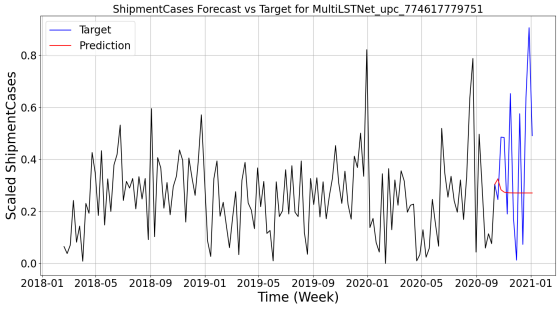
(b) MultiLSTNet-S 12 week forecast for case UPC 772212846677



(c) MultiLSTNet-S 12 week forecast for case UPC 772544621075



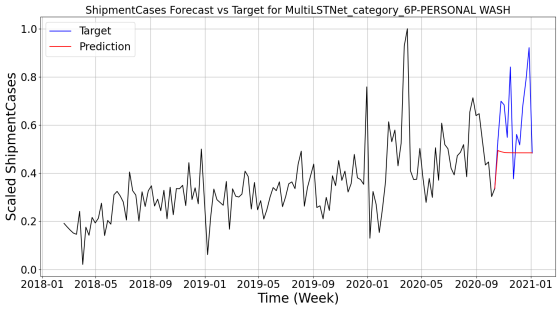
(d) MultiLSTNet-S 12 week forecast for case UPC 772212846677



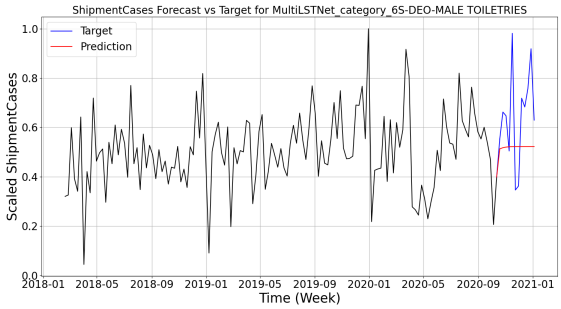
(e) MultiLSTNet-S 12 week forecast for case UPC 772544621075

Figure 25: MultiLSTNet-S 12 week forecasts on 5 case UPC time series

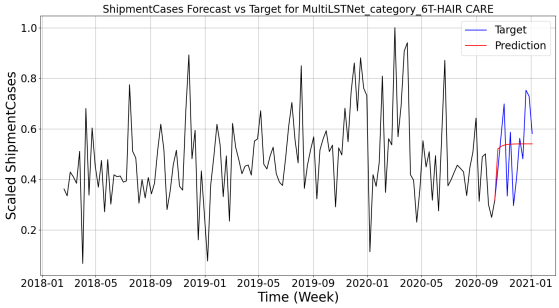
B.4 MultiLSTNet-S on categories



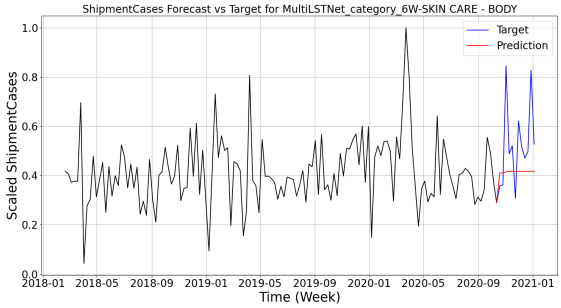
(a) MultiLSTNet-S 12 week forecast for category 6P-PERSONAL WASH



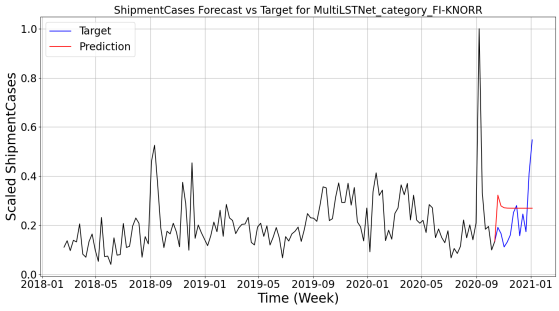
(b) MultiLSTNet-S 12 week forecast for category 6S-DEO-MALE TOILETRIES



(c) MultiLSTNet-S 12 week forecast for category 6T-HAIR CARE



(d) MultiLSTNet-S 12 week forecast for category 6W-SKIN CARE - BODY

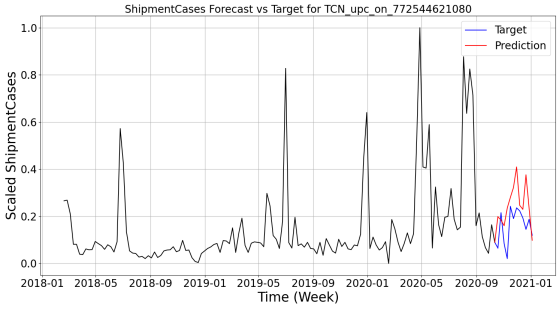


(e) MultiLSTNet-S 12 week forecast for category FI-KNORR

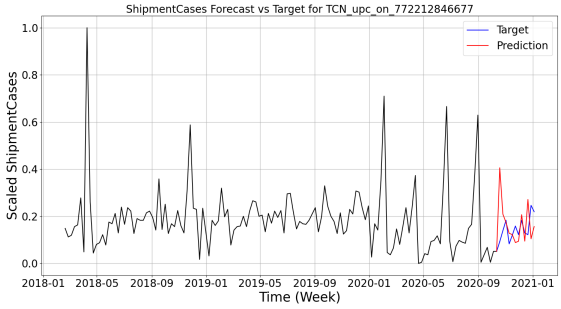
Figure 26: MultiLSTNet-S 12 week forecasts on 5 category time series

C 12 Step Forecast Plots for TCN

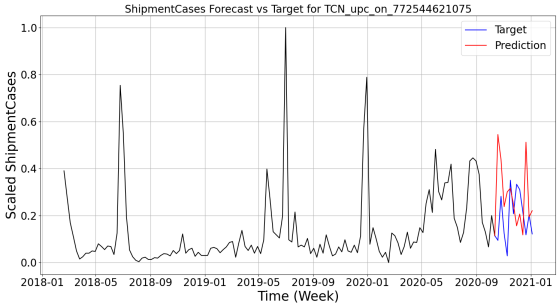
C.1 TCN on case UPCs



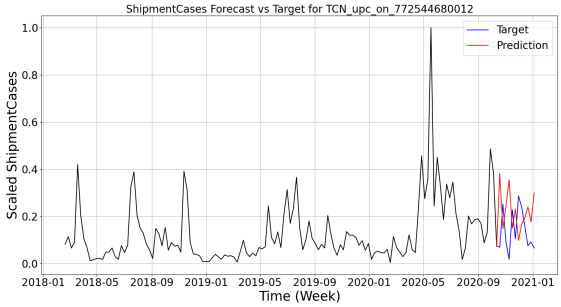
(a) TCN 12 week forecast for case UPC 772544621080



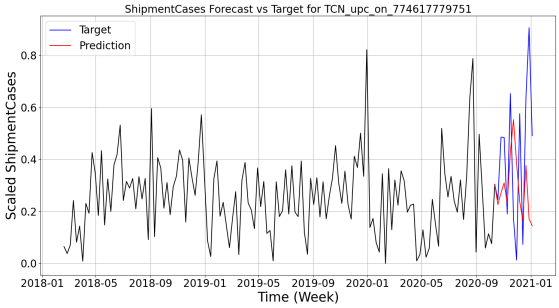
(b) TCN 12 week forecast for case UPC 772212846677



(c) TCN 12 week forecast for case UPC 772544621075



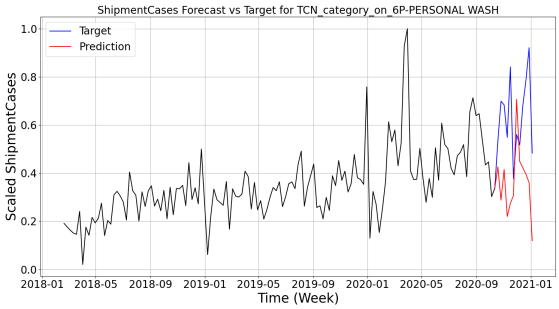
(d) TCN 12 week forecast for case UPC 772212846677



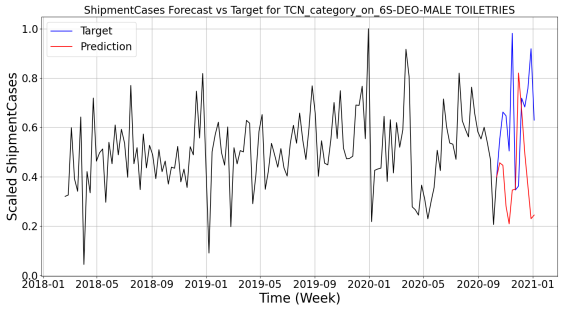
(e) TCN 12 week forecast for case UPC 772544621075

Figure 27: TCN 12 week forecasts on 5 case UPC time series

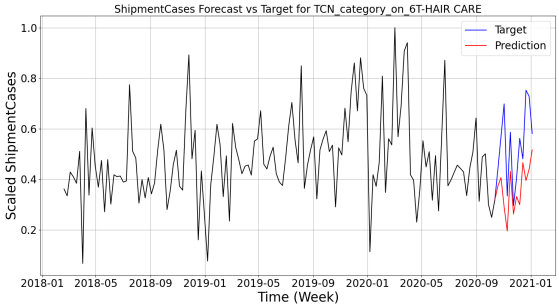
C.2 TCN on categories



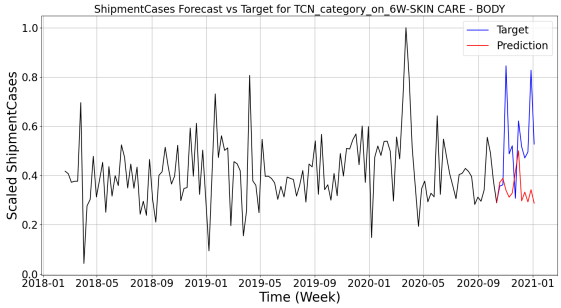
(a) TCN 12 week forecast for category 6P-PERSONAL WASH



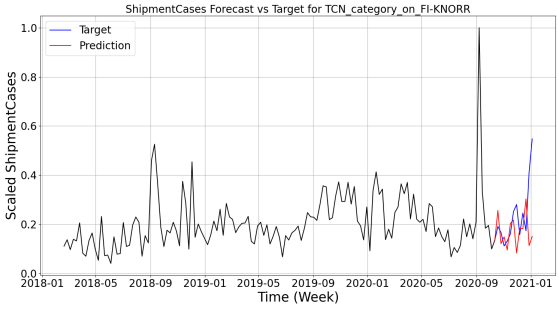
(b) TCN 12 week forecast for category 6S-DEO-MALE TOILETRIES



(c) TCN 12 week forecast for category 6T-HAIR CARE



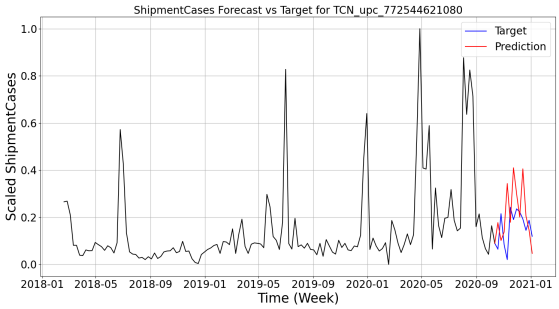
(d) TCN 12 week forecast for category 6W-SKIN CARE - BODY



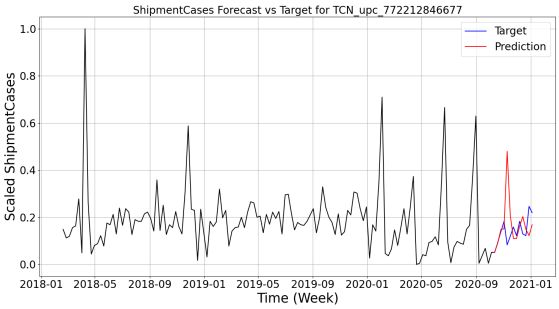
(e) TCN 12 week forecast for category FI-KNORR

Figure 28: TCN 12 week forecasts on 5 category time series

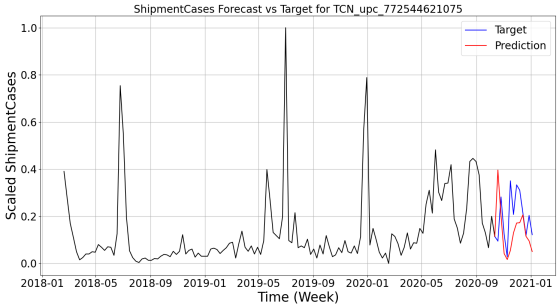
C.3 TCN-S on case UPCs



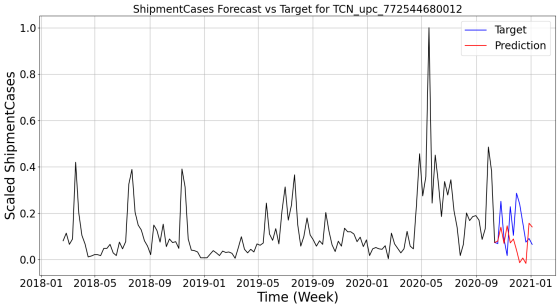
(a) TCN-S 12 week forecast for case UPC 772544621080



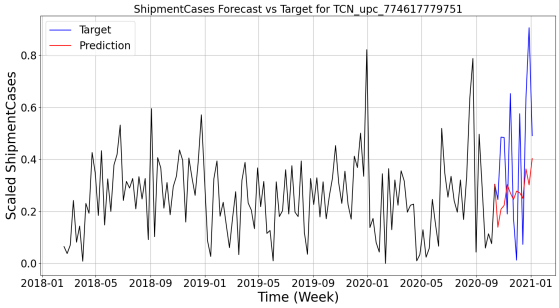
(b) TCN-S 12 week forecast for case UPC 772212846677



(c) TCN-S 12 week forecast for case UPC 772544621075



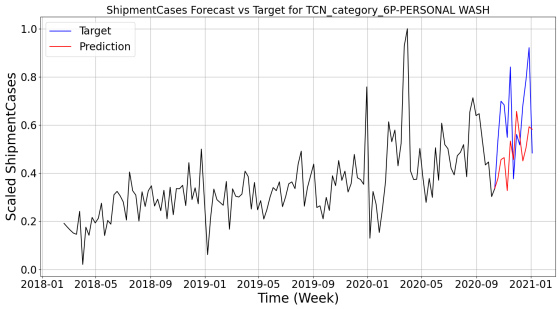
(d) TCN-S 12 week forecast for case UPC 772212846677



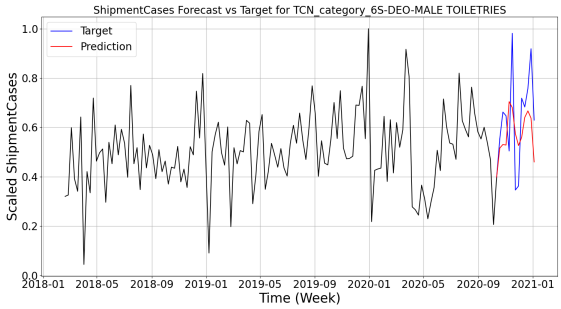
(e) TCN-S 12 week forecast for case UPC 772544621075

Figure 29: TCN-S 12 week forecasts on 5 case UPC time series

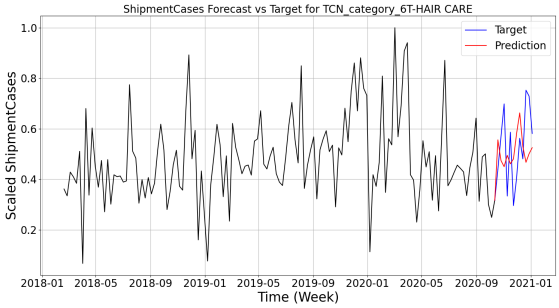
C.4 TCN-S on categories



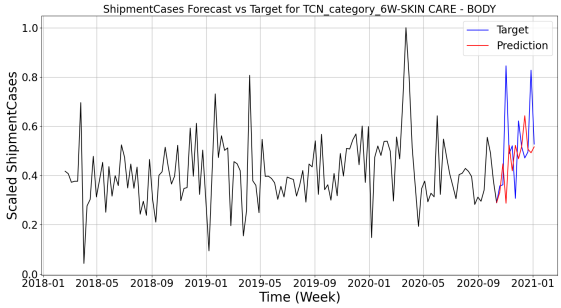
(a) TCN-S 12 week forecast for category 6P-PERSONAL WASH



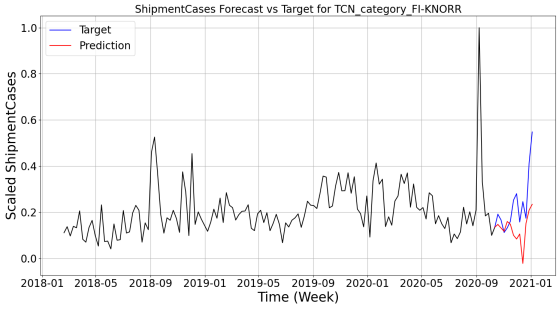
(b) TCN-S 12 week forecast for category 6S-DEO-MALE TOILETRIES



(c) TCN-S 12 week forecast for category 6T-HAIR CARE



(d) TCN-S 12 week forecast for category 6W-SKIN CARE - BODY

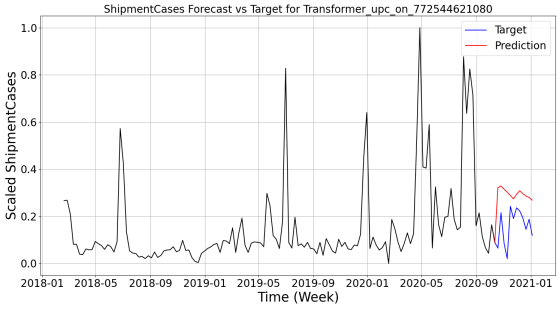


(e) TCN-S 12 week forecast for category FI-KNORR

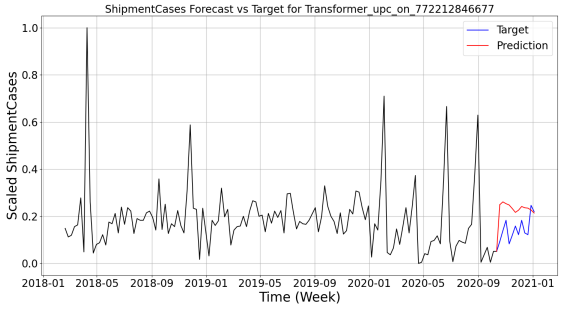
Figure 30: TCN-S 12 week forecasts on 5 category time series

D 12 Step Forecast Plots for Transformer

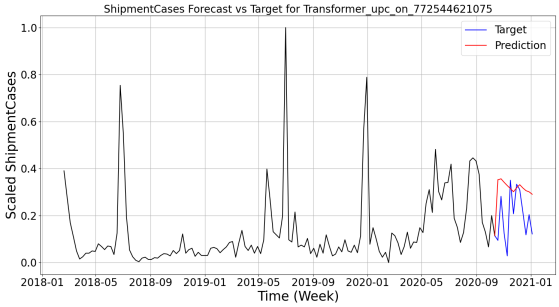
D.1 Transformer on case UPCs



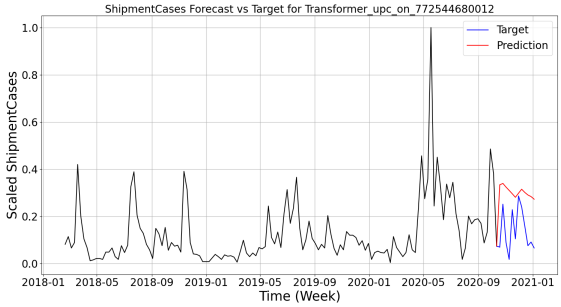
(a) Transformer 12 week forecast for case UPC 772544621080



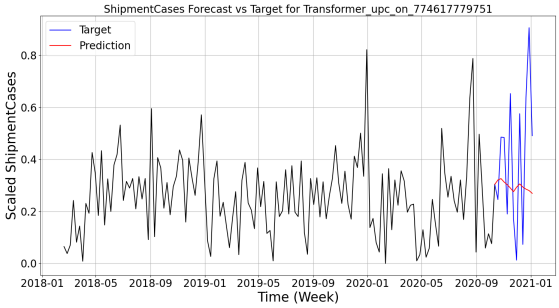
(b) Transformer 12 week forecast for case UPC 772212846677



(c) Transformer 12 week forecast for case UPC 772544621075



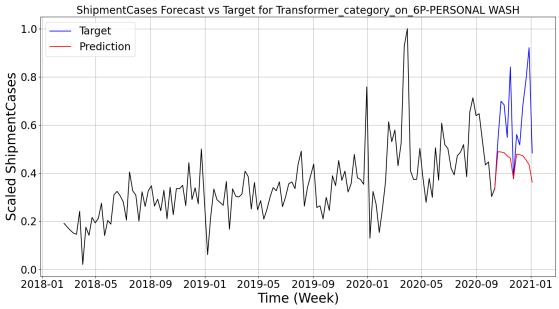
(d) Transformer 12 week forecast for case UPC 772212846677



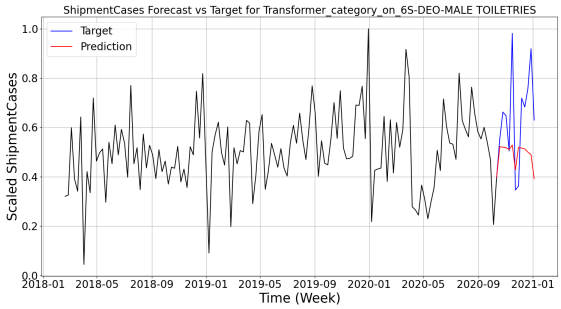
(e) Transformer 12 week forecast for case UPC 772544621075

Figure 31: Transformer 12 week forecasts on 5 case UPC time series

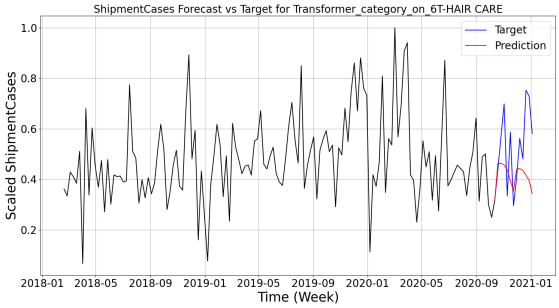
D.2 Transformer on categories



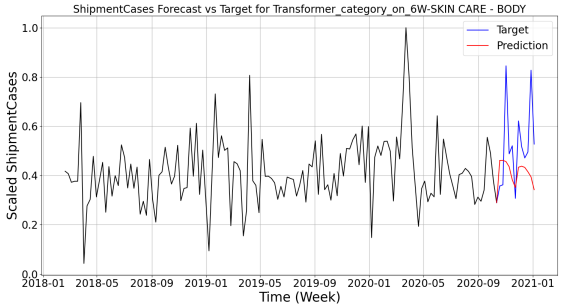
(a) Transformer 12 week forecast for category 6P-PERSONAL WASH



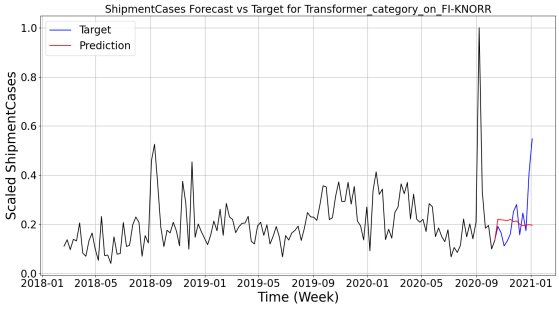
(b) Transformer 12 week forecast for category 6S-DEO-MALE TOILETRIES



(c) Transformer 12 week forecast for category 6T-HAIR CARE



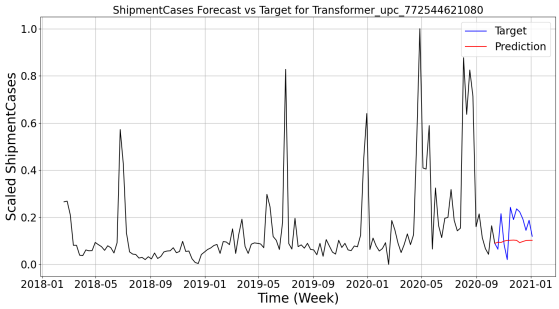
(d) Transformer 12 week forecast for category 6W-SKIN CARE - BODY



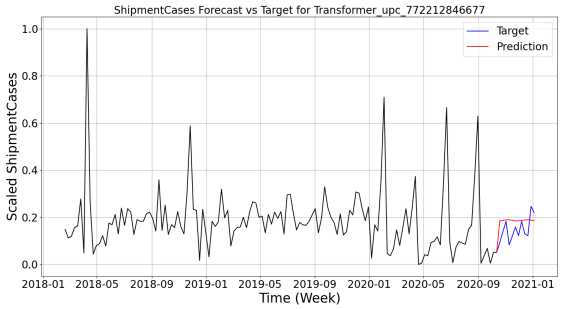
(e) Transformer 12 week forecast for category FI-KNORR

Figure 32: Transformer 12 week forecasts on 5 category time series

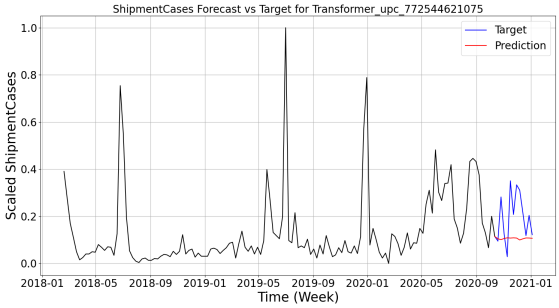
D.3 Transformer-S on case UPCs



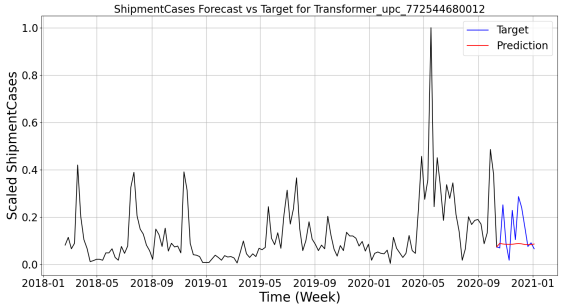
(a) Transformer-S 12 week forecast for case UPC 772544621080



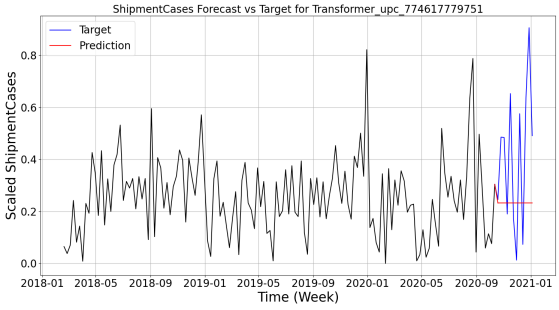
(b) Transformer-S 12 week forecast for case UPC 772212846677



(c) Transformer-S 12 week forecast for case UPC 772544621075



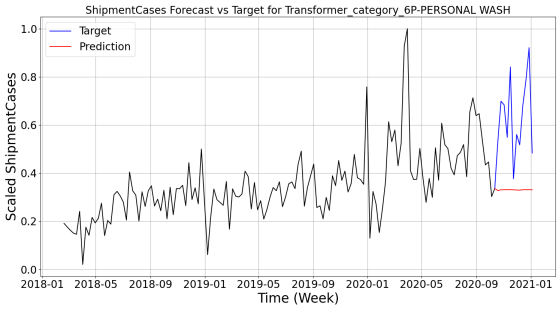
(d) Transformer-S 12 week forecast for case UPC 772212846677



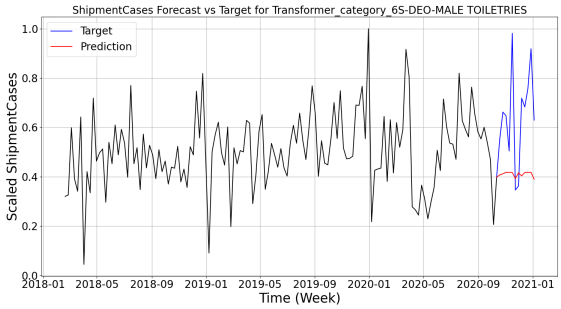
(e) Transformer-S 12 week forecast for case UPC 772544621075

Figure 33: Transformer-S 12 week forecasts on 5 case UPC time series

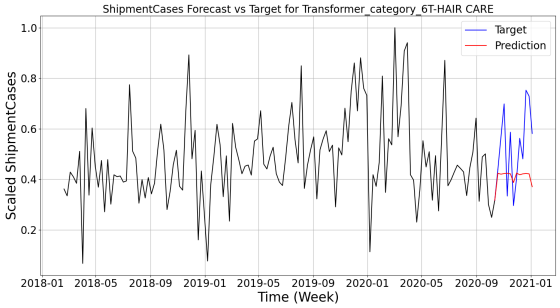
D.4 Transformer-S on categories



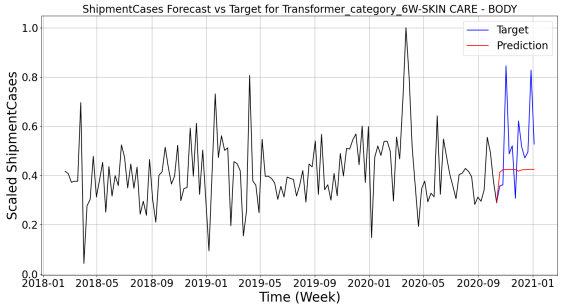
(a) Transformer-S 12 week forecast for category 6P-PERSONAL WASH



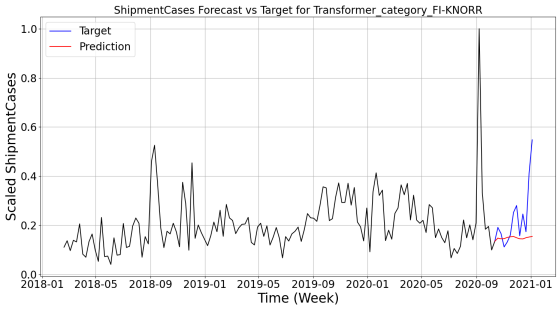
(b) Transformer-S 12 week forecast for category 6S-DEO-MALE TOILETRIES



(c) Transformer-S 12 week forecast for category 6T-HAIR CARE



(d) Transformer-S 12 week forecast for category 6W-SKIN CARE - BODY

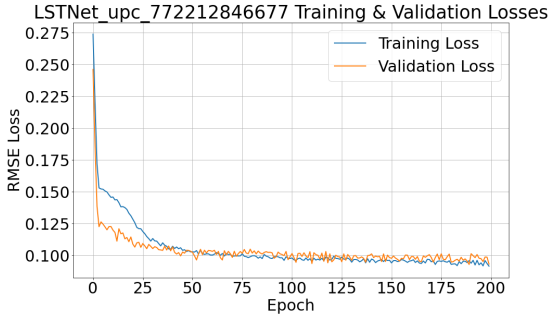


(e) Transformer-S 12 week forecast for category FI-KNORR

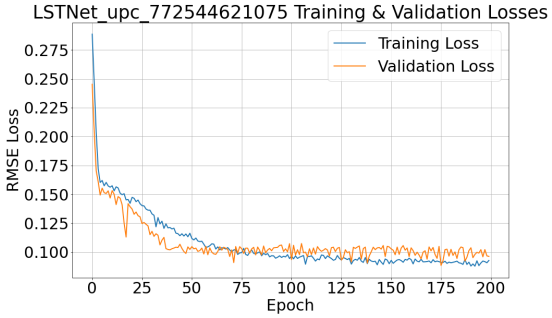
Figure 34: Transformer-S 12 week forecasts on 5 category time series

E Training Loss Plots for LSTNet-S

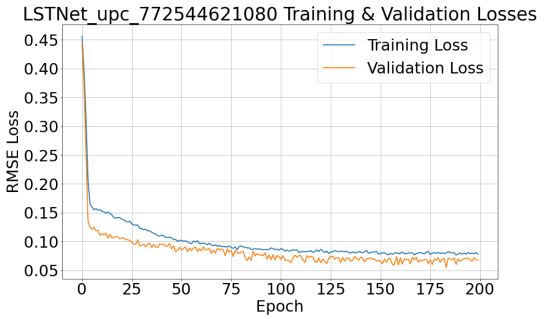
E.1 LSTNet-S losses on case UPCs



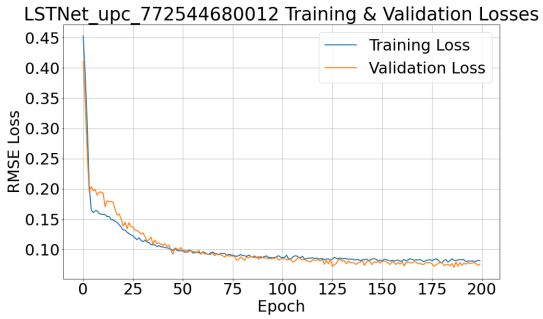
(a) Training and validation losses for the LSTNet-S model trained on case UPC 772212846677



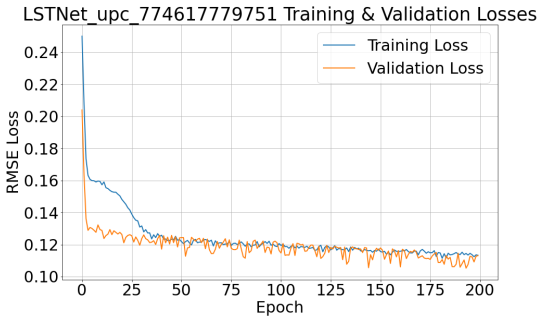
(b) Training and validation losses for the LSTNet-S model trained on case UPC 772544621075



(c) Training and validation losses for the LSTNet-S model trained on case UPC 772544621080



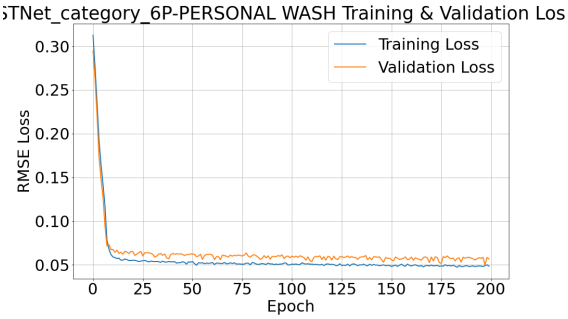
(d) Training and validation losses for the LSTNet-S model trained on case UPC 772544680012



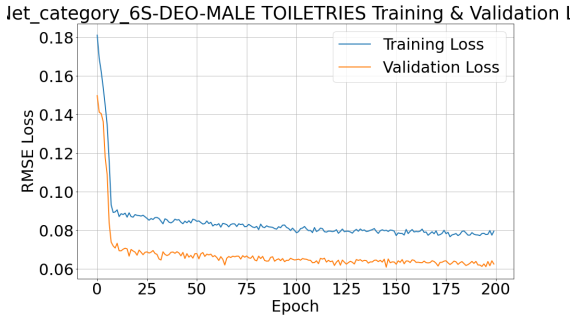
(e) Training and validation losses for the LSTNet-S model trained on case UPC 774617779751

Figure 35: Training and validation loss curves for the LSTNet-S model trained on case UPCs

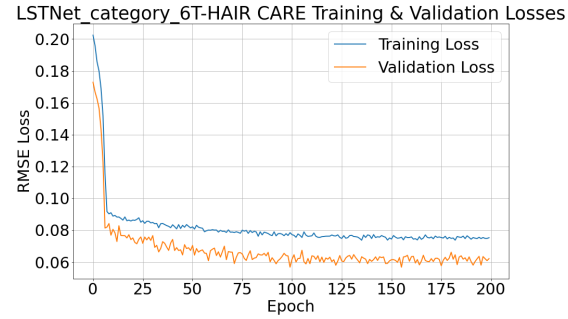
E.2 LSTNet-S losses on Categories



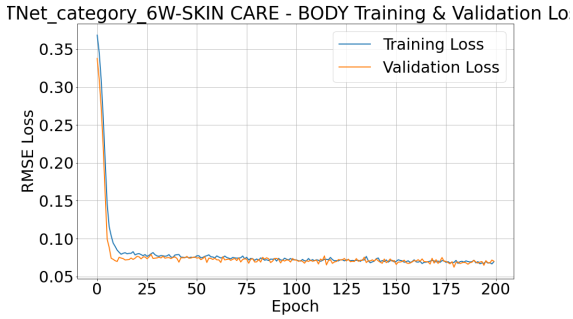
(a) Training and validation losses for the LSTNet-S model trained on category 6P-PERSONAL WASH



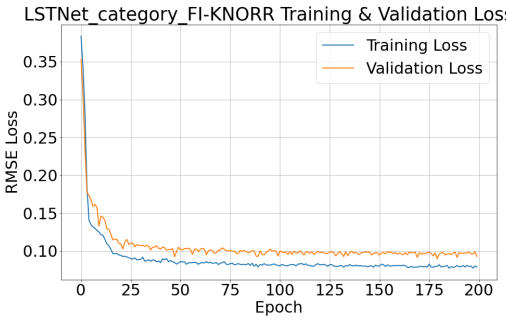
(b) Training and validation losses for the LSTNet-S model trained on category 6S-DEO-MALE TOILETRIES



(c) Training and validation losses for the LSTNet-S model trained on category 6T-HAIR CARE



(d) Training and validation losses for the LSTNet-S model trained on category 6W-SKIN CARE - BODY

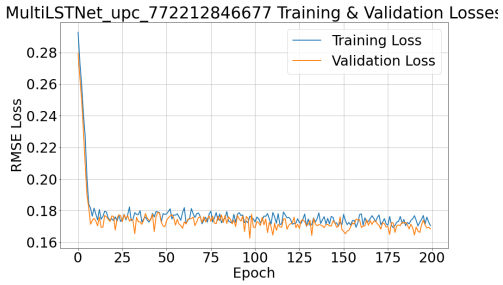


(e) Training and validation losses for the LSTNet-S model trained on category FI-KNORR

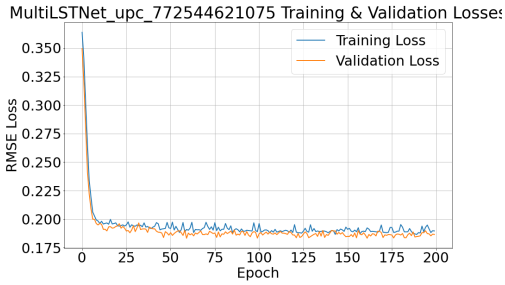
Figure 36: Training and validation loss curves for the LSTNet-S model trained on categories

F Training Loss Plots for MultiLSTNet-S

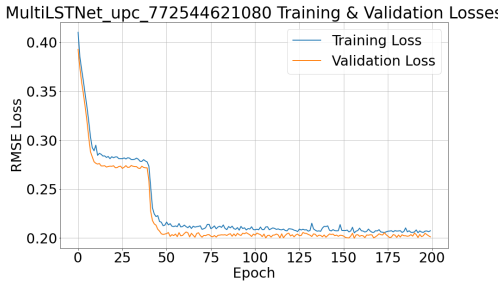
F.1 MultiLSTNet-S losses on case UPCs



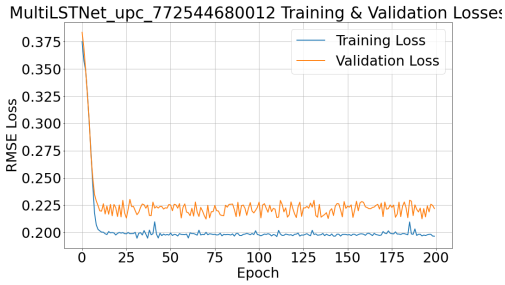
(a) Training and validation losses for the MultiLSTNet-S model trained on case UPC 772212846677



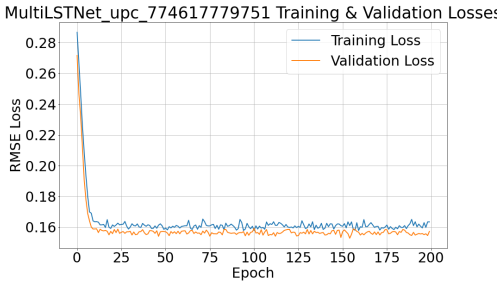
(b) Training and validation losses for the MultiLSTNet-S model trained on case UPC 772544621075



(c) Training and validation losses for the MultiLSTNet-S model trained on case UPC 772544621080



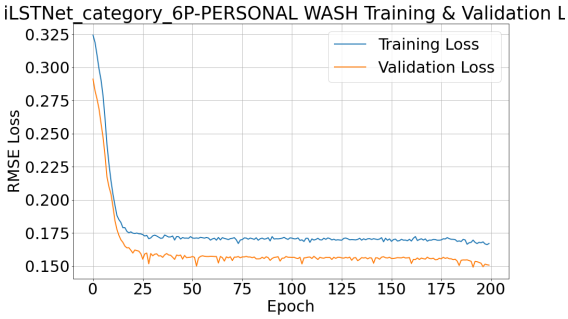
(d) Training and validation losses for the MultiLSTNet-S model trained on case UPC 772544680012



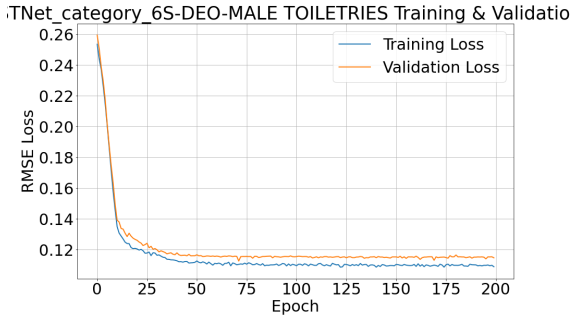
(e) Training and validation losses for the MultiLSTNet-S model trained on case UPC 774617779751

Figure 37: Training and validation loss curves for the MultiLSTNet-S model trained on case UPCs

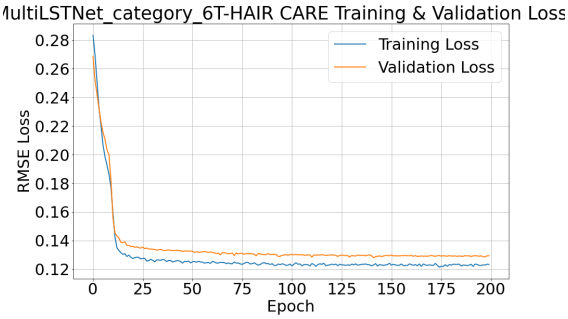
F.2 MultiLSTNet-S losses on Categories



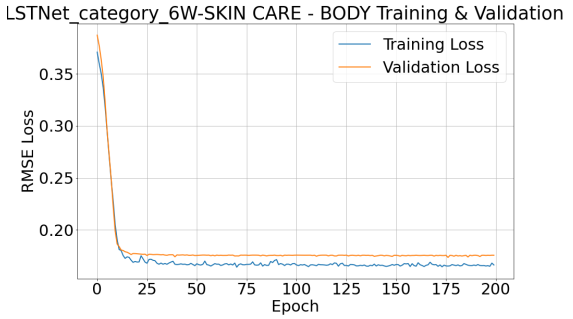
(a) Training and validation losses for the MultiLSTNet-S model trained on category 6P-PERSONAL WASH



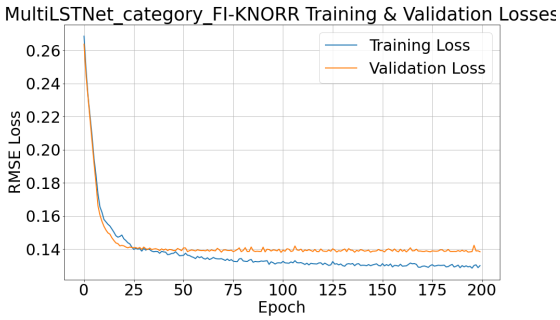
(b) Training and validation losses for the MultiLSTNet-S model trained on category 6S-DEO-MALE TOILETRIES



(c) Training and validation losses for the MultiLSTNet-S model trained on category 6T-HAIR CARE



(d) Training and validation losses for the MultiLSTNet-S model trained on category 6W-SKIN CARE - BODY

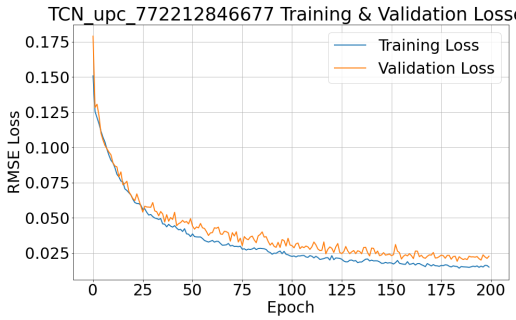


(e) Training and validation losses for the MultiLSTNet-S model trained on category FI-KNORR

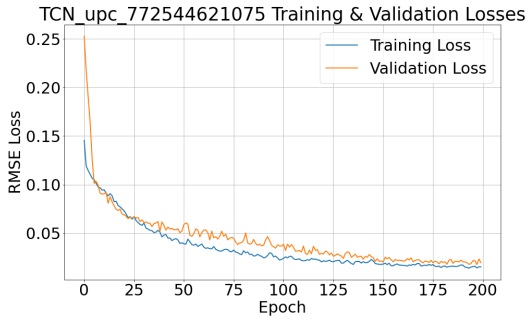
Figure 38: Training and validation loss curves for the MultiLSTNet-S model trained on categories

G Training Loss Plots for TCN-S

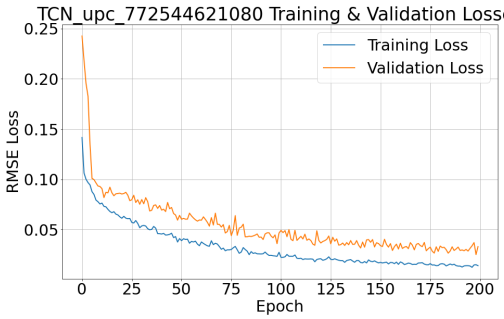
G.1 TCN-S losses on case UPCs



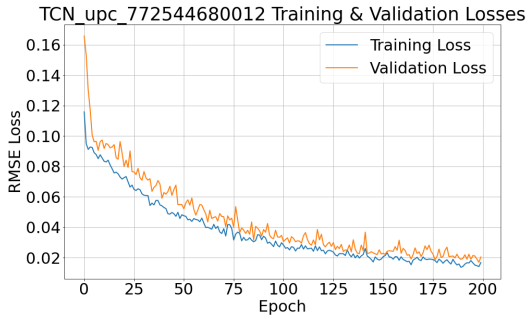
(a) Training and validation losses for the TCN-S model trained on case UPC 772212846677



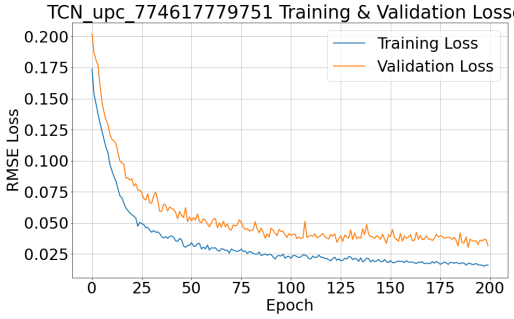
(b) Training and validation losses for the TCN-S model trained on case UPC 772544621075



(c) Training and validation losses for the TCN-S model trained on case UPC 772544621080



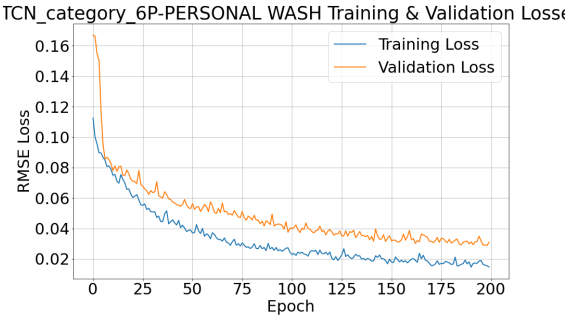
(d) Training and validation losses for the TCN-S model trained on case UPC 772544680012



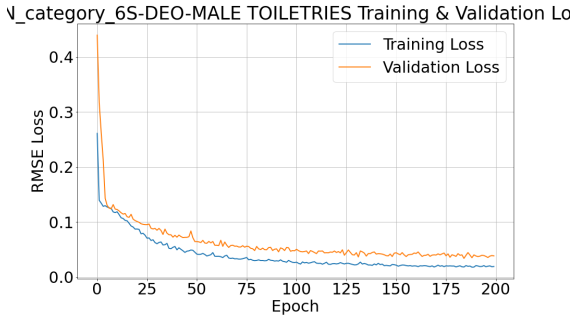
(e) Training and validation losses for the TCN-S model trained on case UPC 774617779751

Figure 39: Training and validation loss curves for the TCN-S model trained on case UPCs

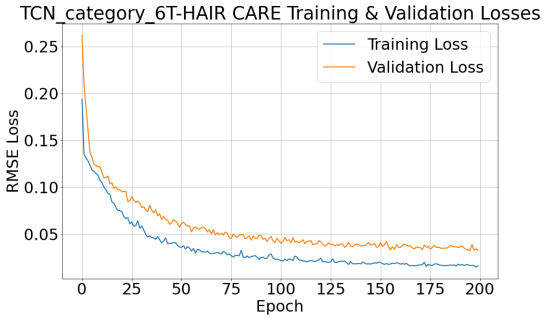
G.2 TCN-S losses on Categories



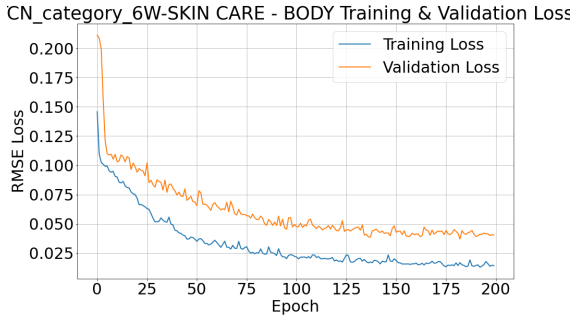
(a) Training and validation losses for the TCN-S model trained on category 6P-PERSONAL WASH



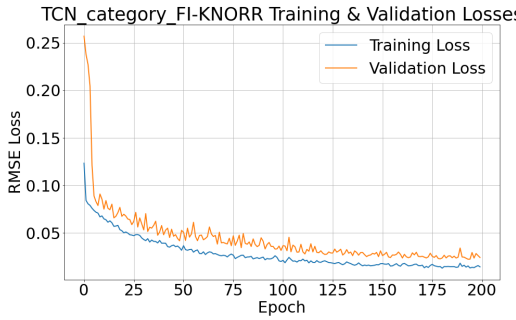
(b) Training and validation losses for the TCN-S model trained on category 6S-DEO-MALE TOILETRIES



(c) Training and validation losses for the TCN-S model trained on category 6T-HAIR CARE



(d) Training and validation losses for the TCN-S model trained on category 6W-SKIN CARE - BODY

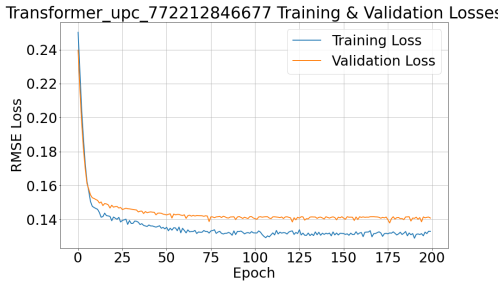


(e) Training and validation losses for the TCN-S model trained on category FI-KNORR

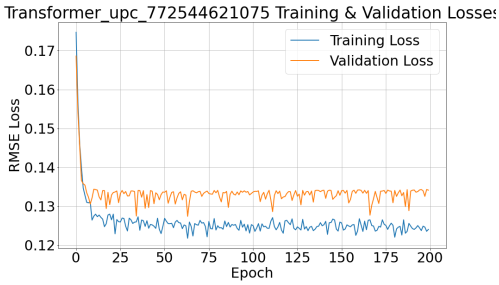
Figure 40: Training and validation loss curves for the TCN-S model trained on categories

H Training Loss Plots for Transformer-S

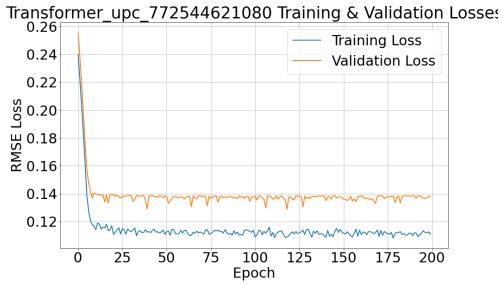
H.1 Transformer-S losses on case UPCs



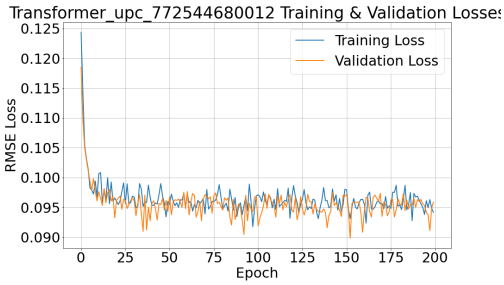
(a) Training and validation losses for the Transformer-S model trained on case UPC 772212846677



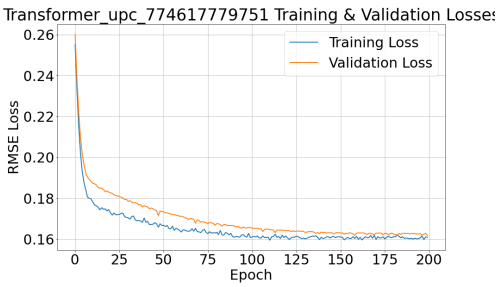
(b) Training and validation losses for the Transformer-S model trained on case UPC 772544621075



(c) Training and validation losses for the Transformer-S model trained on case UPC 772544621080



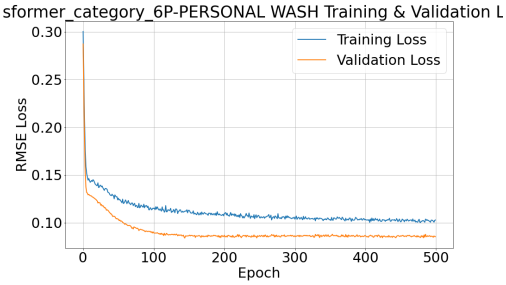
(d) Training and validation losses for the Transformer-S model trained on case UPC 772544680012



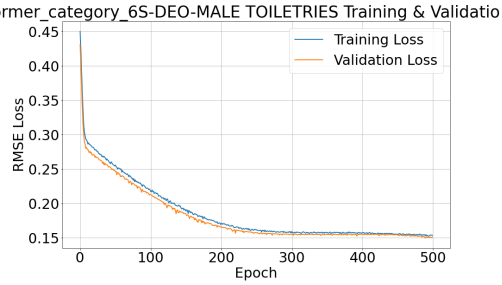
(e) Training and validation losses for the Transformer-S model trained on case UPC 774617779751

Figure 41: Training and validation loss curves for the Transformer-S model trained on case UPCs

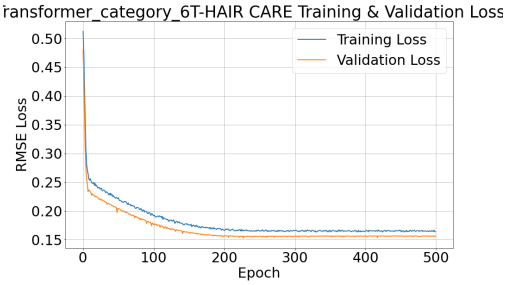
H.2 Transformer-S losses on Categories



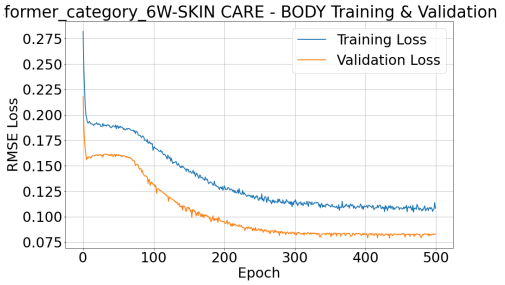
(a) Training and validation losses for the Transformer-S model trained on category 6P-PERSONAL WASH



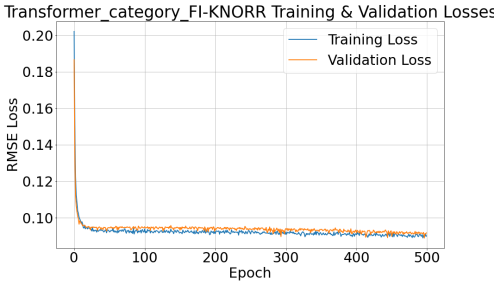
(b) Training and validation losses for the Transformer-S model trained on category 6S-DEO-MALE TOILETRIES



(c) Training and validation losses for the Transformer-S model trained on category 6T-HAIR CARE



(d) Training and validation losses for the Transformer-S model trained on category 6W-SKIN CARE - BODY



(e) Training and validation losses for the Transformer-S model trained on category FI-KNORR

Figure 42: Training and validation loss curves for the Transformer-S model trained on categories