# Prediction of Store Demands by Decision Trees and Recurrent Neural Networks Ensemble with Transfer Learning

Nikica Perić[1][a], Naomi-Frida Munitić[1][b], Ivana Bašljan[1][c] and Vinko Lešić[1][d]

*Laboratory for Renewable Energy Systems, Faculty of Electrical Engineering and Computing,*
*University of Zagreb, Zagreb, Croatia*

Keywords: Multi Period VRP, Prediction of Delivery Capacities, Gradient Boosting Decision Trees, Recurrent Neural Networks, Transfer Learning.

Abstract: Simple vehicle routing problem (VRP) algorithms today achieve near-optimal solution and solve problems with a large number of nodes. Recently, these algorithms are upgraded with additional constraints to respect an increasing number of real-world conditions and, further on, adding a predictive character to the optimization. A distinctive contribution lies in taking into account the predictions of orders that are yet to occur. Such problems fall under time series approaches that are most often obtained using statistical methods or historical data heuristics. Machine learning methods have proven to be superior to statistical methods in most of the literature. In this paper, machine learning techniques for predicting the mass of total daily orders for individual stores are further elaborated and tested on historical data of a local retail company. Among the tested methods are Gradient Boosting Decision Tree methods (XGBoost and LightGBM) and methods of Recurrent Neural Networks (LSTM, GRU and their variations using transfer learning). Finally, an ensemble of these methods is performed, which provides the highest prediction accuracy. The final models use the information on historical order quantities and time-related slack variables.

## 1 INTRODUCTION

The Vehicle Routing Problem (VRP) is one of the most studied topics when it comes to combinatorial optimization and operations research. The goal of a VRP is to enable orders to be delivered to the desired locations in a desired time with the lowest possible costs. Since the problem is very complex computationally (complexity is $O(n!)$, where $n$ is the total number of locations), it is impossible to find an optimal solution for problems with a large number of locations in a limited time. Therefore, various methods are developed to find a suboptimal solution. Heuristic and meta-heuristic methods based on local search are most often used (Gillett and Miller, 1974), (Rochat and Taillard, 1995), (Nagata and Bräysy, 2009). More recently a number of algorithms use deep reinforcement learning (Nazari et al., 2018).

There are different variants of a VRP depending on what requirements it should meet during optimization. Among established variants are Capacitated Vehicle Routing Problem (CVRP) introduced in (Dantzig and Ramser, 1959) and Vehicle Routing Problem with Time Windows (VRPTW) elaborated in (Solomon, 1987). Recent research often analyzes other variants such as Multidepot Vehicle Routing Problem (MDVRP), (Lau et al., 2010), Three-dimensional Loading Capacitated Vehicle Routing Problem (3L-CVRP,) (Tarantilis et al., 2009) or Pickup and Delivery Vehicle Routing Problem (PDVRP), (Chen and Fang, 2019). Additional variants are often combined with CVRP and VRPTW to better incorporate real-world requirements as e.g. in (Wang et al., 2016). Despite many requirements that are being considered in optimization, little attention is paid to the predictive component.

The Time Dependent Vehicle Routing Problem (TDVRP) takes into account the time component to avoid traffic jams. The TDVRP consideres 3.06% of VRP papers from the 2009-2015 period according to (Braekers et al., 2016). The predictive component that combines current orders with anticipated orders for other days or shifts is used in even fewer VRP algorithms. This variant is called Multi Period Vehicle

[a] https://orcid.org/0000-0002-2476-6305
[b] https://orcid.org/0000-0002-8337-6251
[c] https://orcid.org/0000-0001-9495-6342
[d] https://orcid.org/0000-0003-1595-6016

Perić, N., Munitić, N., Bašljan, I. and Lešić, V.
Prediction of Store Demands by Decision Trees and Recurrent Neural Networks Ensemble with Transfer Learning.
DOI: 10.5220/0010802500003116
In *Proceedings of the 14th International Conference on Agents and Artificial Intelligence (ICAART 2022) - Volume 3*, pages 218-225
ISBN: 978-989-758-547-0; ISSN: 2184-433X

Routing Problem (MPVRP) and can be found in (Wen et al., 2010) or (Mancini, 2016). The authors identify variables where predictions should improve the existing state of VRP algorithms, especially in the last-mile delivery. These are: *i*) prediction of the delivery point time matrix (used for TDVRP), *ii*) prediction of store (delivery point) activity, and *iii*) prediction of the quantity of goods to be delivered (can be divided into mass prediction and volume prediction). The TDVRP implies variable time matrices at different times of the day, which also takes into account realistic traffic phenomena. Using these matrices, it is possible to avoid traffic jams in cities by sending vehicles mostly to the outskirts of the city during heavy traffic, and to the city center when there is less traffic. Predictions of store (delivery point) activity allow the algorithm to better schedule locations visits so that a delivery vehicle visits a specific location fewer times. These predictions are beneficial for a VRP with a time horizon, and if orders can be postponed for other days. Prediction of the quantity of goods to be delivered is also essential in this case. Estimating the quantity needed to be delivered over the time horizon is important for determination of how many vehicles and what types of vehicles are needed to deliver all orders. If necessary, it is possible to postpone deliveries to another day or to deliver in advance, which reduces the need for borrowing external vehicles or overtime work and thus the total operating costs.

In this paper, methods for predicting the mass of goods to be delivered are compared. Statistical models such as Autoregressive Integrated Moving Average (ARIMA) and some of its more advanced variations are mostly used for similar time series predictions (Zhang, 2003), (Pavlyuk, 2017). With the growing popularity of machine learning, new methods have been developed that are superior in accuracy to such statistical methods. (Siami-Namini et al., 2018). Machine learning methods are used for similar applications in: (Anzar, 2021), (Mackenzie et al., 2019) and (Fu et al., 2016). For retail sales series that are often found as non-linear problems, due to the seasonality, basic statistical models and linear models of machine learning are unable to solve such problems with a higher accuracy. Therefore, it is recommended to experiment with advanced forecasting methods such as Neural Networks or Gradient Boosting Methods (Wanchoo, 2019). In further research, it is also worth to notice that those two models mostly perform better than other time series and regression techniques, including cases related to store demands (Hodžić et al., 2019).

The methods tested are separated into two groups: Gradient Boosting Decision Tree (GBDT) and Recurrent Neural Networks (RNN). Among the GBDT methods, Extreme Gradient Boosting (XGBoost) and Light Gradient Boosting Method (LightGBM) are considered. Among the RNN methods, Long Short-Term Memory (LSTM), Gated Recurrent unit (GRU) are tested, and these two methods with transfer learning are considered as a special cases. The same methods can be used after re-tuning of hyperparameters to predict the time matrix and predict the activity of the stores.

This paper first briefly presents all the methods used in Section II. Section III describes the observed dataset. Utilized approaches are presented in Section IV. Data pre-processing, common to all models, is described first, followed by the selection of hyperparameters and post-processing. Section V presents the results. In Section VI, a conclusion is given.

## 2 METHODOLOGY

The GBDT and RNN methods listed in the introduction are used to predict the mass of goods to be delivered. These methods are described below.

### 2.1 Gradient Boosting Decision Tree

The GBDT is a widely used machine learning algorithm thanks to its efficiency and interpretability. The model consists of an ensemble of weak models (decision trees) that through the epochs with the use of previous models become more accurate and together give better predictions than individual models. These predictions, due to a large number of ”weak” models, give robustness to the common model. Extreme Gradient Boosting (Chen and Guestrin, 2016) and LightGBM Method (Ke et al., 2017) are currently among the most successful GBDT implementations. Compared to standard GBDT, XGBoost provides a parallel tree boosting to increase speed, uses regularized model and implements Dropouts meet multiple Additive Regression Trees (DART) to reduce overfitting, Newton Boosting to converge faster, etc. The LightGBM algorithm has some changes in addition to those XGBoost has. The changes are: *i*) the Histogram-Based Gradient Boosting algorithm which increases the execution speed, *ii*) leaf-wise (best-first) tree growth instead of level-wise tree growth in XGBoost, *iii*) support for categorical features, etc. The disadvantage is a large number of hyperparameters, which makes it difficult to do a detailed grid search.

### 2.2 Recurrent Neural Networks

The application of Recurrent Neural Networks has increased significantly with the increase in computer processing power in recent years. The RNNs are selected here because of the chain structure corresponding to time series predictions. The main disadvantage of classical RNNs is the vanishing gradient problem, which is mitigated by recently established methods such as LSTM (Hochreiter and Schmidhuber, 1997) and GRU (Chung et al., 2014). The LSTM networks solve this problem using three regulators (gates): forget, input and output. The GRU networks are introduced as a variation of LSTM in which the number of gates is reduced to 2: reset and update. This results in higher speed and fewer neurons in a single cell, making the GRU networks easier to prevent overfitting.

### 2.3 Transfer learning

Transfer learning is a method where the model is first trained using one, usually larger dataset, and then reused and adapted to another, usually smaller dataset. Although more commonly used in classification problems, transfer learning has some applications in time series prediction (He et al., 2019), (Chaurasia and Pal, 2020). According to (Tan et al., 2018), transfer learning is separated into four categories: instances-based, mapping-based, network-based and adversarial-based. The approach used in this paper is network-based transfer learning. This is applicable when multiple similar datasets relating to individual units (stores in this paper) are available. First, a common base model is generated that is trained on all datasets. Most of the hidden layers of the neural network remain from this model, and the last layers are then re-trained on individual sets. In this way, each individual set has own prediction model while the initial layers are the same for all, and the last layers are specific to each of the individual sets. The described approach is shown in Fig. 1.

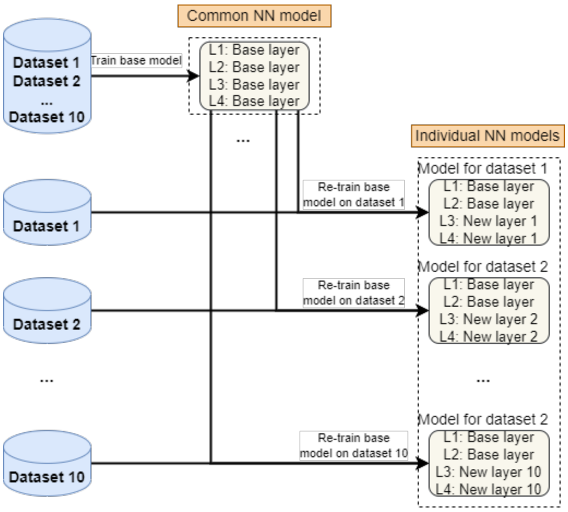

Figure 1: Network based transfer learning structure for 10 individual sets.

## 3 DATASET

The presented predictions are generated from a historical dataset of a retail company that owns over 1000 delivery locations, of which over 200 are in the considered city. Most locations relate to stores and warehouses. These stores are separated into large, medium and small categories based on the daily turnover of goods and the type of goods in the store. The company also owns a heterogeneous fleet of delivery vehicles. These vehicles transport goods from warehouses to stores. Due to the access restrictions of some trucks and with the aim of quality planning of the MDVRP, it is important that the stores are observed individually. Also, groups of stores differ in some ordering habits, so most large stores have orders every work day, while small ones usually order demands 2-3 times a week. This is evident from Fig. 2, which on the left shows mass distribution of demands for small store 1 over the days of the week, and for large store 1 on the right. Therefore, 10 small, 10 medium and 10 large stores are selected to test the machine learning models. For small stores, data processing, tables and predictions are presented in detail for elaboration and presentation purpose and while for medium and large stores, only the final prediction results are shown for methods scale-up purpose. The data processing procedure is the same for all 3 types of stores.

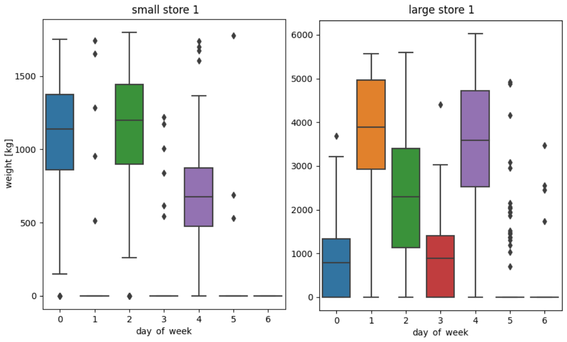

Figure 2: Distribution of mass in kilograms by days of week for small store 1 and big store 1.

The original dataset refers to 2018 and 2019 (730 days) and is structured as a list of orders. Each order corresponds to a specific date and mass of goods.

The dataset is restructured into a 2D array where the columns represent different stores, and each row corresponds to a single day. The distribution by mass of daily orders for each small store is shown in Fig. 3. The x-axis shows small stores sorted by indices (from 1 to 10), and the y-axis shows the mass that stores ordered. The wider part of the graph for a particular store indicates a larger number of orders of similar mass over the observed time period. For example, store 5 has fewer days with order deliveries compared to store 8. Store 1 has an approximately normal distribution if we exclude non-delivery days, compared to store 7.

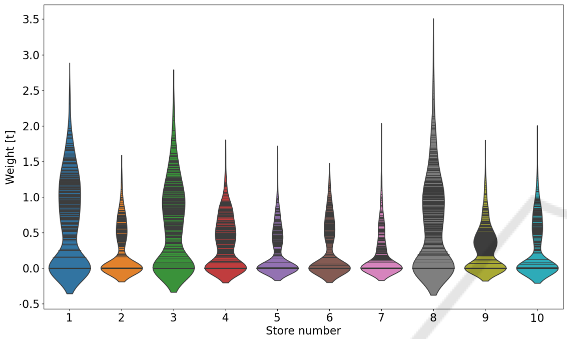

Figure 3: Distribution of mass in tons by daily orders for each small store.

# 4 UTILIZED APPROACHES

In order to increase the quality of predictions, the dataset is preprocessed and inputs obtained by *feature engineering* are tested. The already mentioned 6 models with appropriate hyperparameters and an ensemble of them is tested, and then the results are post processed. In the sequel, *common models* notation is used for transfer learning models from the upper part of Fig. 1 and *individual models* refers to XGBoost, LightGBM, LSTM, GRU, and the lower part of Fig. 1.

## 4.1 Preprocessing and Feature Engineering

As mentioned before, the dataset is separated into three smaller datasets containing orders for 10 small, 10 medium and 10 large stores from 2018 and 2019. The reason for the separation into categories is that common models use data similar to individual models because stores of the same size have similar habits of ordering goods. The input data for all 6 models are the same and consist of the variables listed in Table I. The common models for first store $s_1$ have $s_{i,j}$, $d_j$ and $w_j$ as the inputs, and the inputs of the individuals model are $s_{1,j}$, $d_j$ and $w_j$. The inputs contain data from the last 14 days ($t_s \cdot h$), by which the model predicts demands for the next day ($t_s \cdot f$). The variable $s$ is used to designate days in a week, with demands for all 10 stores.

Table 1: Models inputs, outputs and common hyperparameters.

| Variable and values | Description |
|---|---|
| $s_i$ | store designation, $i \in\{1,...,10\}$ |
| $s_{i,j}\in\{1,...,m_{max}\}$ | historical store demands,<br>$i \in\{1,...,10\}$, $j \in\{1,...,h\}$ |
| $d_j\in\{1,...,7\}$ | day of week, $j \in\{1,...,h\}$ |
| $w_j\in\{0, 1\}$ | working or non-working day,<br>$j \in\{1,...,h\}$ |
| $p_{i,j}\in\{1,...,m_{max}\}$ | predictions of store demands,<br>$i \in\{1,...,10\}$, $j \in\{1,...,f\}$ |
| $m \geqslant 0$ | Store demands mass in kg |
| $t_s$ = 1 day | Predictions and data resolution |
| $h$ | Amount of historical input values |
| $f$ | Amount of future values to predict |

It is observed that stores of similar size have similar behavior. Small stores usually have 1-3 orders per week, while large ones have orders every working day. In some stores, a change in the customer habits is noticeable, e.g., a change in the usual ordering days from Tuesday and Thursday to Monday and Friday. In some, an increase or decrease in the number of orders is evident during the observed period, which is caused by a change in the turnover of the store, and consequently by the number of workers in it. Among the noticeable deviations are also single change of order day, different behavior before the holidays, etc. Since individual models often do not have information to learn such behavior changes well enough, common models are introduced in which such behavior changes are learned from other stores. In order not to create bias, common models for small, medium and large stores are separated according to the average mass of demands in one day:

$$s_{i,avg} = \frac{\sum_{j=1}^{730} s_{i,j}}{730}, \tag{1}$$

as shown in Table II.

Table 2: Separation of stores by size.

| Stores size | daily demands mass [kg] | | |
|---|---|---|---|
| | min | max | average |
| small | 0 | 500 | 242.9 |
| medium | 500 | 1250 | 824.6 |
| large | 1250 | - | 3704.4 |

The correlation matrix of input variables is shown in Fig. 4 for the category of small stores. The order of the variables was chosen according to the correlation value to make clusters of similar stores more noticeable. As expected, $w$ has a positive and $d$ negative correlation with all stores. Very high correlations should be noted: $s_2$, $s_5$ and $s_{10}$ group, $s_3$, $s_4$ and $s_8$ group and $s_6$ to $s_7$. These groups of stores have common ordering habits (mostly ordering on the same day of the week), which makes the common model more adjusted to them. The variables $d$ and $w$ are mainly used to identify the days with orders, which is evident from their correlations (both have similar correlations to each of the stores, especially $w$).

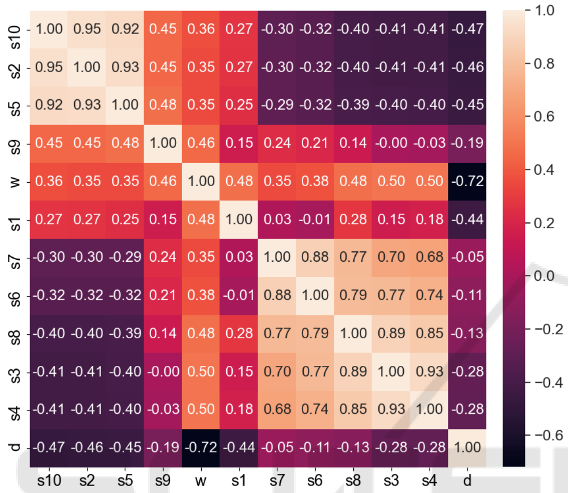

Figure 4: Correlation matrix of all models inputs.

The insertion of a rolling average on the input and a slack variable that gives 1 on days when there is a demand, and 0 when there is no demand, is also tested. However, these variables did not prove to be beneficial for the models.

In Fig. 3, several outliers can be seen. They are corrected using the 2σ rule. When calculating outliers, days without deliveries are not considered, and too big values of demands are corrected using the formula:

$$s_{i,j} = \max(s_{i,j}, s_{i,avgp} + 2 \cdot s_{i,stdp} + 0.1 \cdot s_{i,j}), \qquad (2)$$

where $s_{i,avgp}$ and $s_{i,stdp}$ are average and standard deviation of positive $s_{i,j}$ values. In this way, an average of 3 outliers per store is corrected.

Datasets in RNN models are separated chronologically into parts for training, validation and test in the ratio of approximately 60%-20%-20%, and in GBDT methods they are separated into parts for training and test in the ratio 80%-20%. For RNN methods, MinMax scaler with [0, 1] limits is used.

## 4.2 Hyperparameters Selection

After selecting inputs, prediction models are created. Grid search is applied to all models. The best obtained hyperparameter values for the XGBoost and LightGBM models are shown in Table III. Table IV shows parameters for LSTM, GRU, LSTM with transfer learning and GRU with transfer learning.

Table 3: XGBoost and LightGBM hyperparameters.

| **XGBoost** | **LightGBM** |
|---|---|
| reg_lambda = 0.15 | min_data_in_leaf = 20 |
| reg_alpha = 0.004 | num_leaves = 10 |
| subsample = 0.4 | subsample = 0.7 |
| colsample_bytree = 0.4 | subsample_freq = 5 |
| max_depth = 4 | max_depth = 5 |
| n_estimators = 500 | colsample_bytree = 0.7 |
| learning_rate = 0.01 | learning_rate = 0.04 |

Table 4: RNN methods hyperparameters.

| **Hyper-parameter** | **LSTM** | **GRU** | LSTM **transfer** | **GRU transfer** |
|---|---|---|---|---|
| Layers | LSTM(40)<br>Dropout(0.2)<br>LSTM(40)<br>Dropout(0.2)<br>LSTM(1) | GRU(40)<br>Dropout(0.1)<br>GRU(25)<br>Dropout(0.1)<br>GRU(1) | LSTM(40)<br>Dropout(0.2)<br>LSTM(40)<br>Dropout(0.2)<br>LSTM(1) | GRU(40)<br>Dropout(0.2)<br>GRU(40)<br>Dropout(0.2)<br>GRU(1) |
| Transfer learning layers | / | / | LSTM(25)<br>LSTM(1) | GRU(25)<br>GRU(1) |
| Loss functions | RMSE<br>MAE* | RMSE<br>MAE* | RMSE<br>MAE* | RMSE<br>MAE* |
| Optimizer | Adam | Adam | Adam | Adam |
| Learning rate | 0.003 | 0.003 | 0.002<br>0.001** | 0.002<br>0.001** |
| Epochs | 200 | 200 | 200<br>30** | 200<br>30** |
| Batch size | 50 | 50 | 50 | 50 |

Used for evaluation only, not in training

Refers to the learning of an individual part of a model

## 4.3 Post-processing and Ensemble Model

After selecting the hyperparameters, 6 models are trained and predictions are obtained. RNN prediction values are scaled back to the original range to be comparable to GBDT methods. After that, all predictions values less than a quarter of the mean value are saturated to 0. An ensemble of the four most accurate models has been created, which gives the weighted average of the output of these four models at the output. The predictions of the ensemble model are calculated by:

$$p_{i,j(e)} = 0.1 p_{i,j(1)} + 0.2 p_{i,j(4)} + 0.25 p_{i,j(5)} + 0.45 p_{i,j(6)}, \qquad (3)$$

where $p_{i,j(e)}$ is ensemble prediction, $p_{i,j(1)}$ is XGBoost prediction, $p_{i,j(4)}$ is GRU prediction, $p_{i,j(5)}$ is LSTM with transfer learning prediction and $p_{i,j(6)}$ is GRU with transfer learning prediction.

# 5 EXPERIMENT AND RESULTS

## 5.1 Small Stores

All models are tested on test sets for 10 small stores. Table V shows the results. The upper values for each store refer to Root Mean Squared Error (RMSE) and the lower values to Mean Absolute Error (MAE). RMSE is observed as the main metric. In addition to the results for all 10 stores individually, average errors and the time in seconds required to learn the model for all 10 stores are added.

Table 5: Comparison of prediction accuracy for small stores category.

| store | XG-Boost | Light-GBM | LSTM | GRU | LSTM transfer | GRU transfer | ens-amble |
|---|---|---|---|---|---|---|---|
| s1 | 265.0<br>151.8 | 271.2<br>151.8 | 256.6<br>140.1 | 277.2<br>136.2 | 258.1<br>140.2 | 264.0<br>142.3 | 252.2<br>135.6 |
| s2 | 140.2<br>67.2 | 145.3<br>69.7 | 142.1<br>58.3 | 136.1<br>55.3 | 134.4<br>61.0 | 126.0<br>56.8 | 127.1<br>57.9 |
| s3 | 341.7<br>198.2 | 352.8<br>204.0 | 329.2<br>187.7 | 335.8<br>177.9 | 338.5<br>178.2 | 305.2<br>174.3 | 307.3<br>168.8 |
| s4 | 188.8<br>101.9 | 196.9<br>109.3 | 197.9<br>127.3 | 181.0<br>95.8 | 172.5<br>94.1 | 174.2<br>98.2 | 166.9<br>92.8 |
| s5 | 119.5<br>53.1 | 129.1<br>58.0 | 127.0<br>52.9 | 121.8<br>55.2 | 93.9<br>43.1 | 102.5<br>48.4 | 98.3<br>46.1 |
| s6 | 171.7<br>82.2 | 184.4<br>89.8 | 224.5<br>124.7 | 168.8<br>82.7 | 169.5<br>75.8 | 161.3<br>81.0 | 156.0<br>77.6 |
| s7 | 108.9<br>44.8 | 117.2<br>50.7 | 141.1<br>88.3 | 94.7<br>45.5 | 98.6<br>43.2 | 89.3<br>42.3 | 88.9<br>41.7 |
| s8 | 307.7<br>153.6 | 345.6<br>176.1 | 287.1<br>144.9 | 303.6<br>158.4 | 354.4<br>179.5 | 302.1<br>158.0 | 301.7<br>157.9 |
| s9 | 175.7<br>95.8 | 179.3<br>95.1 | 189.4<br>103.3 | 171.4<br>101.9 | 175.3<br>95.0 | 159.7<br>86.1 | 160.6<br>89.6 |
| s10 | 150.2<br>72.7 | 151.5<br>74.8 | 159.1<br>68.0 | 151.0<br>73.4 | 129.4<br>57.8 | 134.2<br>64.9 | 132.6<br>62.9 |
| avg. | 197.0<br>102.1 | 207.3<br>107.9 | 206.4<br>109.6 | 194.1<br>98.2 | 192.5<br>96.8 | 181.9<br>95.2 | 179.2<br>93.1 |
| Time | 4.9 | 1.8 | 255.6 | 196.7 | 169.7 | 140.7 | 512.0 |

Upper values in the rows denote RMSE, lower are for MAE

Table shows that the XGBoost method gives better accuracy than LightGBM, and the GRU is more accurate than LSTM, according to RMSE and MAE. Two models using transfer learning compared to the same methods without transfer learning achieve significant progress: 9% for LSTM and 5% for GRU. Transfer learning brings the greatest progress in $s_5$ and $s_{10}$ stores, which together with $s_2$ make up the group of stores with the highest correlations (Fig. 4). It is concluded that, by increasing the number of observed stores, transfer learning could bring an additional advantage in the accuracy of predictions. In that case, instead of separating stores according to the number of deliveries, it would be good to use some more advanced form of clustering. The ensemble of 4 best methods provides the best results as expected, 2% better than GRU with transfer learning.

Part of the predictions from Table V are also shown in Fig. 5. The figure shows the 45 days of predictions given by the algorithm on the test set of the $s3$ store. Predictions from the figure omit described post-processing (saturation to zero) for the purpose of better illustration. During the usual, steady-state, schedule of order days, all models give similar predictions. A higher difference in accuracy occurs on days when the order schedule changes rapidly. The schedule has changed on 8.10.2019., which greatly influenced the predictions for the following days. Models with XGBoost and LightGBM do not change the behavior much, LSTM partially changes the behavior, and the other 3 models adapt more to the new behavior, especially the GRU model with transfer learning. Precisely, such situations are the biggest advantage of transfer learning methods. The disadvantage of these methods may be the learning speed at which XGBoost and LightGBM are far better. Nevertheless, all tested algorithms are fast enough for this application. Applications such as TDVRP in which predictions are made for each of the two store combinations should also consider the speed component.

## 5.2 All Stores

The algorithms are tested on a set of 10 medium and a set of 10 large stores. Table VI lists RMSE and MAE for all 3 types of stores. The results for medium and large stores are similar to those for the small ones. Predictions with GBDT and RNN methods have similar results, and transfer learning brings advances in models with an emphasis on smaller stores. The best results are obtained using an ensemble of a few methods. The results of the ensemble of methods are compared with a model that copies occurrences from 7 days ago for working days and predicts 0 for non-working days. A 57% lower loss is obtained for small stores, 60% lower for medium, and 54% for large ones. Such persistence model gives RMSE = 412.4kg and MAE = 221.8kg for all 10 observed test datasets of 4 months for the small stores, i.e. the ensemble approach provides 57% improvement in prediction accuracy. For the medium and the large stores, ensemble provides 60% and 54% improvement, respectively, while the most evident improvement is during rapid changes in stores demand. This is of great benefit to Multi Period VRP because prediction of total weight of order directly affects the number of vehicles, overtime hours, but also the total cost of delivery.

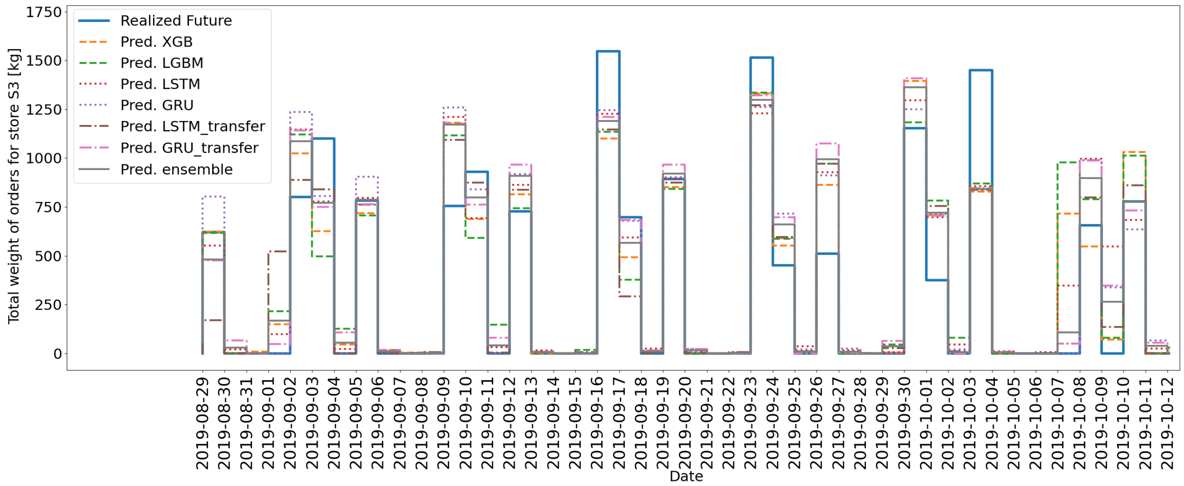

Figure 5: Comparison of prediction accuracy for all tested methods.

Table 6: Comparison of prediction accuracy for all stores categories.

| stores size | XG-Boost | Light-GBM | LSTM | GRU | LSTM transfer | GRU transfer | ens-amble |
|---|---|---|---|---|---|---|---|
| small | 197.0 | 207.3 | 206.4 | 194.1 | 192.5 | 181.9 | 179.2 |
| | 102.1 | 107.9 | 109.6 | 98.2 | 96.8 | 95.2 | 93.1 |
| medium | 645.1 | 660.1 | 688.5 | 639.6 | 654.1 | 638.7 | 616.8 |
| | 346.5 | 345.4 | 355.6 | 330.9 | 338.3 | 329.3 | 322.2 |
| large | 2398 | 2385 | 2470 | 2467 | 2621 | 2368 | 2339 |
| | 1670 | 1647 | 1692 | 1699 | 1832 | 1660 | 1647 |

Upper values in the rows denote RMSE, lower are for MAE

## 6 CONCLUSION

In this paper, machine learning methods for prediction of delivery capacities in last-mile logistics are tested. These predictions are important to enable the use of Multi Period VRP. Predictions are generated using Gradient Boosting Decision Tree methods (XGBoost and LightGBM) and methods of Recurrent Neural Networks (LSTM, GRU and their variations using transfer learning). Real historical datasets are used, divided into 3 categories according to store size. At the inputs of all algorithms are the historical mass values of the store order and slack variables depicting working days and day of the week. Preprocessing and post-processing are applied. Among the mentioned methods, GRU with transfer learning proves to be the most accurate. Transfer learning generally brings an improvement over the same metrics without transfer learning, GRU is more accurate than LSTM, and XGBoost is more accurate than LightGBM. RNN methods are more accurate than GBDT methods for small and medium-sized stores where orders are more volatile, and GBDT methods are more accurate in large stores. Eventually an ensemble of these methods is generated which, as expected, gives the most accurate predictions (2% compared to GRU with transfer learning and 57% compared to persistence model).

## ACKNOWLEDGEMENTS

This work has been supported by the European Union from the European Regional Development Fund via Operational Programme Competitiveness and Cohesion 2014-2020 for Croatia through the project Research and development of a unified system for logistic and transport optimisation - Collaborative Elastic and Green Logistics - CEGLog (grant KK.01.2.1.02.0081). The authors would like to thank GDi Group for their collaboration.

## REFERENCES

Anzar, T. (2021). Forecasting of Daily Demand's Order Using Gradient Boosting Regressor BT - Progress in Advanced Computing and Intelligent Engineering. pages 177–186, Singapore. Springer Singapore.

Braekers, K., Ramaekers, K., and Van Nieuwenhuyse, I. (2016). The vehicle routing problem: State of the art classification and review. *Computers & Industrial Engineering*, 99:300–313.

Chaurasia, V. and Pal, S. (2020). Application of machine learning time series analysis for prediction COVID-19 pandemic. *Research on Biomedical Engineering*.

Chen, R. M. and Fang, P. J. (2019). Solving Vehicle Routing Problem with Simultaneous Pickups and Deliveries Based on A Two-Layer Particle Swarm optimization. In *2019 20th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Net-*

*working and Parallel/Distributed Computing (SNPD)*, pages 212–216.

Chen, T. and Guestrin, C. (2016). XGBoost: A scalable tree boosting system. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 13-17-August-2016:785–794.

Chung, J., Gulcehre, C., Cho, K., and Bengio, Y. (2014). Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. pages 1–9.

Dantzig, G. B. and Ramser, J. H. (1959). The Truck Dispatching Problem. *Management Science*, 6(1):80–91.

Fu, R., Zhang, Z., and Li, L. (2016). Using LSTM and GRU neural network methods for traffic flow prediction. In *2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC)*, pages 324–328.

Gillett, B. E. and Miller, L. R. (1974). A Heuristic Algorithm for the Vehicle-Dispatch Problem. *Operations Research*, 22(2):340–349.

He, Q., Pang, P. C.-I., and Si, Y. W. (2019). Transfer Learning for Financial Time Series Forecasting. pages 24–36.

Hochreiter, S. and Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

Hodžić, K., Hasić, H., Cogo, E., and Jurić, Ž. (2019). Warehouse Demand Forecasting based on Long Short-Term Memory neural networks. In *2019 XXVII International Conference on Information, Communication and Automation Technologies (ICAT)*, pages 1–6.

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., and Liu, T. Y. (2017). LightGBM: A highly efficient gradient boosting decision tree. *Advances in Neural Information Processing Systems*, 2017-December(Nips):3147–3155.

Lau, H. C. W., Chan, T. M., Tsui, W. T., and Pang, W. K. (2010). Application of Genetic Algorithms to Solve the Multidepot Vehicle Routing Problem. *IEEE Transactions on Automation Science and Engineering*, 7(2):383–392.

Mackenzie, J., Roddick, J. F., and Zito, R. (2019). An Evaluation of HTM and LSTM for Short-Term Arterial Traffic Flow Prediction. *IEEE Transactions on Intelligent Transportation Systems*, 20(5):1847–1857.

Mancini, S. (2016). A real-life Multi Depot Multi Period Vehicle Routing Problem with a Heterogeneous Fleet: Formulation and Adaptive Large Neighborhood Search based Matheuristic. *Transportation Research Part C: Emerging Technologies*, 70:100–112.

Nagata, Y. and Bräysy, O. (2009). Edge Assembly based Memetic Algorithm for the Capacitated Vehicle Routing Problem. *Networks*, 54:205–215.

Nazari, M., Oroojlooy, A., Takáč, M., and Snyder, L. V. (2018). Reinforcement learning for solving the vehicle routing problem. *Advances in Neural Information Processing Systems*, 2018-Decem:9839–9849.

Pavlyuk, D. (2017). Short-term Traffic Forecasting Using Multivariate Autoregressive Models. *Procedia Engineering*, 178:57–66.

Rochat, Y. and Taillard, É. D. (1995). Probabilistic diversification and intensification in local search for vehicle routing. *Journal of Heuristics*, 1(1):147–167.

Siami-Namini, S., Tavakoli, N., and Namin, A. S. (2018). A Comparison of ARIMA and LSTM in Forecasting Time Series. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 1394–1401.

Solomon, M. M. (1987). Algorithms for the Vehicle Routing and Scheduling Problems with Time Window Constraints. *Operations Research*, 35(2):254–265.

Tan, C., Sun, F., Kong, T., Zhang, W., Yang, C., and Liu, C. (2018). A Survey on Deep Transfer Learning BT - Artificial Neural Networks and Machine Learning – ICANN 2018. pages 270–279, Cham. Springer International Publishing.

Tarantilis, C. D., Zachariadis, E. E., and Kiranoudis, C. T. (2009). A Hybrid Metaheuristic Algorithm for the Integrated Vehicle Routing and Three-Dimensional Container-Loading Problem. *IEEE Transactions on Intelligent Transportation Systems*, 10(2):255–271.

Wanchoo, K. (2019). Retail Demand Forecasting: a Comparison between Deep Neural Network and Gradient Boosting Method for Univariate Time Series. In *2019 IEEE 5th International Conference for Convergence in Technology (I2CT)*, pages 1–5.

Wang, J., Zhou, Y., Wang, Y., Zhang, J., Chen, C. L. P., and Zheng, Z. (2016). Multiobjective Vehicle Routing Problems With Simultaneous Delivery and Pickup and Time Windows: Formulation, Instances, and Algorithms. *IEEE Transactions on Cybernetics*, 46(3):582–594.

Wen, M., Cordeau, J.-F., Laporte, G., and Larsen, J. (2010). The dynamic multi-period vehicle routing problem. *Computers & Operations Research*, 37(9):1615–1623.

Zhang, G. (2003). Time series forecasting using a hybrid ARIMA and neural network model. *Neurocomputing*, 50:159–175.