UDC 004.415.2
DOI https://doi.org/10.32782/2663-5941/2024.1.1/40

Oleshchenko L.M.
National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute"

INTERCITY PASSENGER FLOW FORECASTING AND MTE BUSES OPTIMAL OPERATION USING LSTM NEURAL NETWORK

The article presents an analysis of existing software and software platforms for intelligent data analysis that are used to forecast passenger flow between cities. The main statistical methods used for forecasting passenger flow between cities are given. In the research, a software method of data collection and passenger flow forecasting on intercity routes of Ukraine using LSTM neural network is proposed. Data on passenger flows between the cities of Chernihiv and Kyiv from 2011 to 2015 were used for learning and training the neural network. The Python programming language and the Flask web framework were used to develop the software, in particular, the prediction module. Studies have shown the effectiveness of using the LSTM neural network in comparison with such statistical forecasting methods as linear trend, logarithmic trend, moving average, Holt-Winters exponential smoothing. The proposed software makes it possible to make an optimal plan for the use of the buses based on the forecast value of the passenger flow and the mathematical programming model for the given structure of the rolling stock of the MTE. The LSTM model can be effective for medium-term passenger flow forecasting on long-distance routes because it can take into account long-term time series dependencies such as seasonality and trends. However, it is important to note that LSTM can tend to overfit on small amounts of data, and this can be computationally expensive, especially with large data sizes. In addition, LSTM may not perform effectively under unstable or variable conditions that may occur in the field of passenger flow forecasting.

Key words: *software, Python programming language, machine learning, LSTM neural network, forecasting, data mining software platforms, passenger flow, data processing, intercity passenger transportation, MTE, optimization, buses, mathematical programming.*

Problem statement. The forecasting of intercity passenger flow involves a combination of mathematical and software methods to analyze historical data, identify patterns, and predict future trends. One mathematical method commonly employed is time series analysis, where historical passenger data is examined to understand patterns and variations over time. This involves techniques such as moving averages, exponential smoothing, and autoregressive integrated moving average (ARIMA) models. Additionally, machine learning algorithms, including neural networks like the Long Short-Term Memory (LSTM) model, are increasingly used for their ability to capture complex patterns and dependencies in data. On the software side, forecasting tools and platforms equipped with advanced analytics and machine learning (ML) capabilities provide a user-friendly interface for transportation analysts and planners to apply these mathematical methods effectively. The integration of these mathematical and software approaches allows for accurate predictions, facilitating optimized planning and resource allocation in intercity passenger transportation systems of the motor transport enterprise (MTE).

The purpose of this research is to analyze and improve existing methods of intercity passenger flow forecasting and optimizing MTE buses utilization using LSTM neural network.

Related research and existing software solutions analysis for forecasting passenger flow between cities. Forecasting passenger flow between cities involves a combination of statistical methods and machine learning algorithms, depending on the complexity of the data and the accuracy required [1–7]. ARIMA models are widely used for time series forecasting. They can capture trends, seasonality, and other patterns in historical passenger flow data. Simple linear regression can be used to establish relationships between passenger flow and various predictor variables, such as economic indicators, population growth, or events.

Holt-Winters exponential smoothing method is suitable for time series data with trend and seasonality. It considers three components: level, trend, and seasonality. By assigning weights to recent observations, this method adapts to changing patterns over time. The level represents the average value, the trend captures the direction, and seasonality accounts for repeating patterns. The method involves updating these components iteratively to make accurate predictions, making it particularly effective for time series forecasting, including passenger flow in transportation systems.

Bayesian approaches can incorporate prior knowledge and update predictions based on new data. Bayesian models are useful when dealing with uncertainty in passenger flow forecasts. Random Forests can handle non-linear relationships and interactions between variables. They are robust and effective for forecasting when there are multiple factors influencing passenger flow.

LSTM networks are a type of recurrent neural network (RNN) designed for sequence data. They can capture long-term dependencies and are suitable for time series forecasting (Fig. 1).

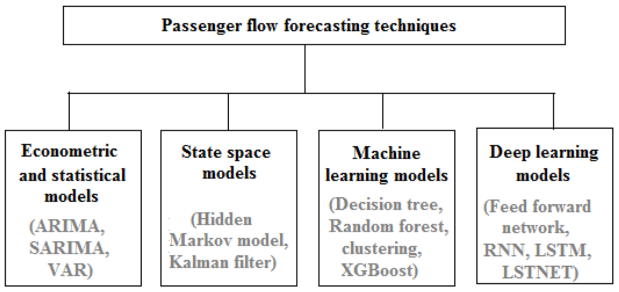

Passenger flow forecasting techniques
Econometric and statistical models
(ARIMA, SARIMA, VAR)
State space models
(Hidden Markov model, Kalman filter)
Machine learning models
(Decision tree, Random forest, clustering, XGBoost)
Deep learning models
(Feed forward network, RNN, LSTM, LSTNET)

Fig. 1. Passenger flow forecasting techniques [1–7]

SVR (Support Vector Regression) is effective for regression tasks, including forecasting. It works well in high-dimensional spaces and is robust to outliers. XGBoost and LightGBM are gradient boosting algorithms that can handle complex relationships and non-linearities. They often perform well in forecasting tasks. K-Nearest Neighbors (KNN) can be used for regression tasks by predicting the average or weighted average of the *k*-nearest neighbors. It's a non-parametric algorithm suitable for various data distributions.

Combining multiple models through ensemble methods can improve overall forecasting accuracy. For instance, combining the predictions of different machine learning algorithms. Clustering can be applied to identify patterns in passenger behavior or to group cities with similar traffic characteristics. A hybrid approach, combining statistical methods with ML algorithms, is often employed for more accurate and robust predictions. Software solutions exist for forecasting passenger flow between cities, offering advanced analytical tools and models to aid in decision-making for transportation planners. These solutions integrate mathematical models and ML algorithms.

TransCAD is a transportation planning software that integrates GIS (Geographic Information System) with transportation modeling. It allows for the analysis of intercity passenger flows, considering factors like demographics, land use, and transportation infrastructure (Fig. 2).

TensorFlow and PyTorch are popular open-source ML libraries that can be used for building predictive models. Data scientists and transportation analysts can leverage these platforms to implement ML algorithms, including neural networks like LSTM, for passenger flow forecasting.

TensorFlow is designed to scale seamlessly from mobile devices to large-scale distributed systems. This scalability is crucial for handling large datasets and complex models in passenger forecasting scenarios.

PyTorch's dynamic computational graph allows for more intuitive model development and easier debugging. This flexibility is advantageous when experimenting with different model architectures for passenger forecasting. PyTorch provides TorchServe, a flexible

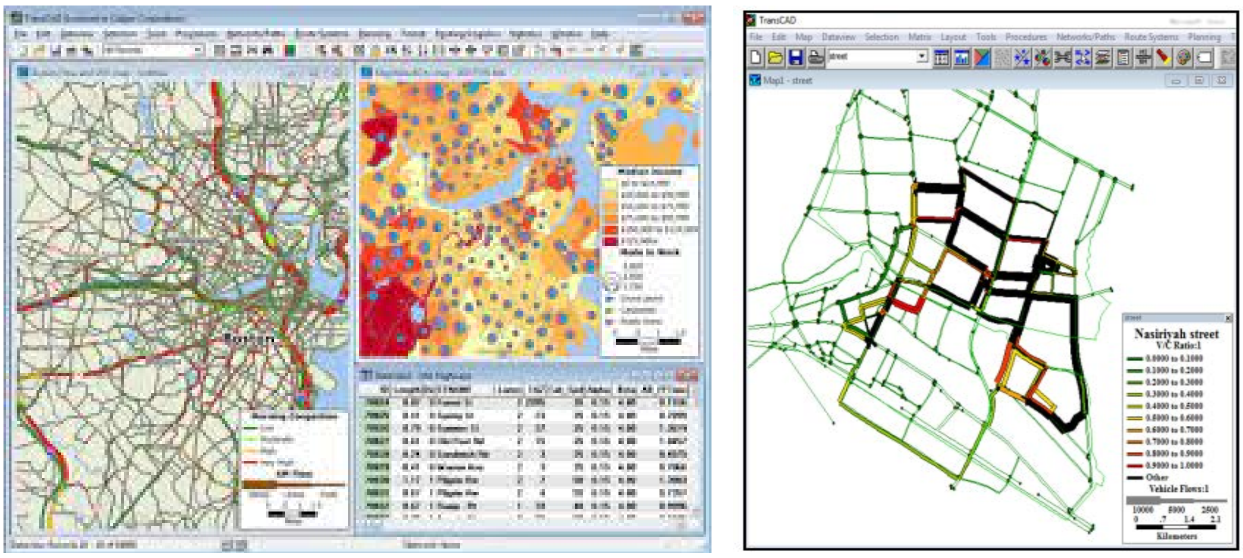


Fig. 2. TransCAD transportation planning software [8]

and easy-to-use library for serving PyTorch models in production environments. This can aid in deploying and managing passenger forecasting models at scale.

RapidMiner is a predictive analytics platform that offers a visual interface for designing, testing, and deploying predictive models. It supports ML algorithms and can be used for forecasting passenger flow based on historical data (Fig. 3).

RapidMiner allows the integration of various data sources, enabling to combine historical passenger data with other relevant information such as weather, holidays, events, or transportation schedules. This comprehensive dataset can improve the accuracy of passenger forecasting models. The platform provides tools for cleaning, transforming, and pre-processing data. This is crucial for passenger forecasting as it helps in handling missing values, outliers, and ensuring data quality, which is essential for building accurate predictive models.

RapidMiner supports a wide range of ML algorithms, including regression, classification, and time series analysis to build models for passenger demand prediction based on historical patterns, allowing transportation providers to optimize resource allocation and improve service efficiency.

RapidMiner offers AutoML capabilities, which can automate the process of selecting and tuning ML models. This is particularly useful for those who may not have extensive data science expertise but still want to harness the power of advanced analytics for passenger forecasting. Given that passenger forecasting often involves time-dependent data, RapidMiner's capabilities in time series analysis can be beneficial. It allows for the identification of temporal patterns and trends that can be used for more accurate predictions.

RapidMiner supports ensemble learning techniques, where multiple models are combined to improve predictive performance. This can be useful in passenger forecasting scenarios where different factors may influence demand, and combining multiple models can provide more robust predictions.

The platform provides tools for visualizing data and model outputs. Clear visualizations can help transportation providers understand the patterns and trends in passenger data, facilitating informed decision-making.

RapidMiner is designed to handle large datasets and can scale to meet the demands of forecasting for transportation systems with significant passenger volumes. Depending on the specific requirements, RapidMiner can be used to deploy models for real-time predictions. This is crucial for dynamic situations where passenger demand patterns may change rapidly.

While not dedicated forecasting tools, data visualization platforms like Tableau and Power BI can be integrated with predictive models. They enable users to create interactive dashboards and visualize forecasted passenger flow trends, making it easier for stakeholders to understand the insights.

In general the passenger flow forecasting process is shown in Fig. 4.

While mainly used for traffic simulation, VISSIM can be employed to model and simulate intercity

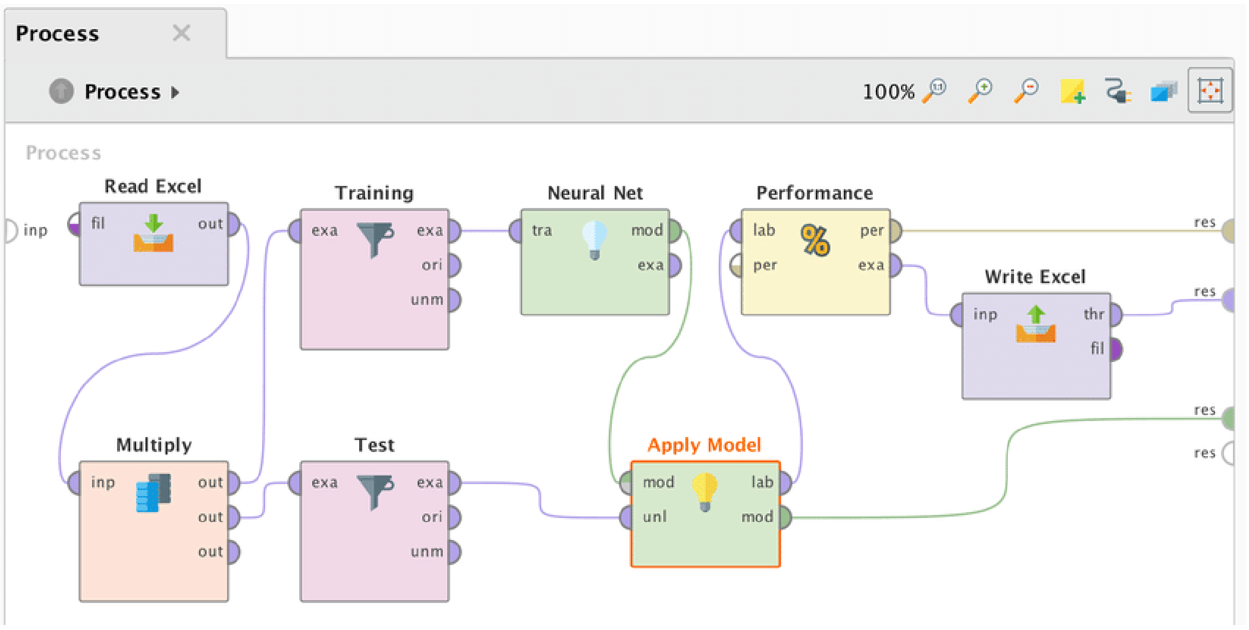


Fig. 3. RapidMiner process model for forecasting

passenger flow scenarios. It helps in understanding the impact of different factors on traffic flow and can aid in forecasting based on simulation results. Analysts often use statistical analysis tools like R and Python to perform in-depth statistical analysis on historical data. The *statsmodels* library in Python provides statistical models suitable for time series analysis and forecasting. These software solutions offer a range of capabilities, from basic statistical analysis to advanced ML models, providing transportation planners with valuable insights for forecasting intercity passenger flow.

Intercity passenger flow forecasting using LSTM neural network. To forecast time series data on passenger flow between Chernihiv and Kyiv cities of Ukraine, we first process the passenger data stored on a secure cloud service. The synchronized data is then formatted into a .xls file. Research experiment uses time series data from 2011 to 2015 (data about passenger flow for every hour). For LSTM-based passenger flow prediction, we follow next steps.

1. Collect training data.
2. Prepare and organize the data (emoving outliers, replacing missing values, smoothing noisy data, correcting inconsistent data).
3. Choose the neural network topology (layers and feedback). Analyze the characteristics of passenger flow data. Understand the patterns, trends, and any potential seasonality. Determine the appropriate sequence length for time series data. Experiment with different sequence lengths to find the optimal one. Choose the number of LSTM layers and hidden units in each layer. This depends on the complexity of the data. More complex patterns may require a deeper network with more hidden units.
4. Experimentally select network characteristics.
5. Empirically determine training parameters.
6. Train the neural network.
7. Validate training adequacy.
8. Adjust parameters based on validation.

Python, with the Keras library, is used for forecasting implementation. Passenger data input depends on lot size, time steps, and hidden size (Fig. 5).

The output is directed to a layer *TimeDistributed*. The final layer employs a *Softmax* activation. This result is compared to training data in batches, initiating error computation and backpropagation gradient flow. Training data, represented by inputs *x*, are processed one time step at each instance, predicting the subsequent sequence value. This occurs at every step, aligning the output layer with the same number of time steps as the input layer.

To format predictions suitably for an LSTM model, each unique value in the dataset is assigned a distinct integer index. The corpus is then reordered with integer identifiers in place of values using functions such as *read_appointment*, *build_appointment*, and *file_to_ids*. This assigns a unique integer to each distinct value. Ultimately, the output file is transformed into a list of these unique integers, enabling its utilization in a neural network.

The initial layer in the network, depicted in the architecture diagram, is the embedding layer. This layer transforms integer references (entries) into meaningful contribution vectors. Monitoring the tensor shape in the network is crucial. For the input layer, it's (*batch_size, num_steps*), and the output is (*batch_size, num_steps, hidden_size*). In the Sequential model, the batch size is consistently the first dimension. The batch size is obtained from the fitting function (like *fit_generator*), making it less frequently specified.

The subsequent layer is the first of two LSTM layers. To configure an LSTM layer, we specify the number of nodes in the hidden layers within the LSTM cell (e.g., cells in the forgotten gate step). The code also includes

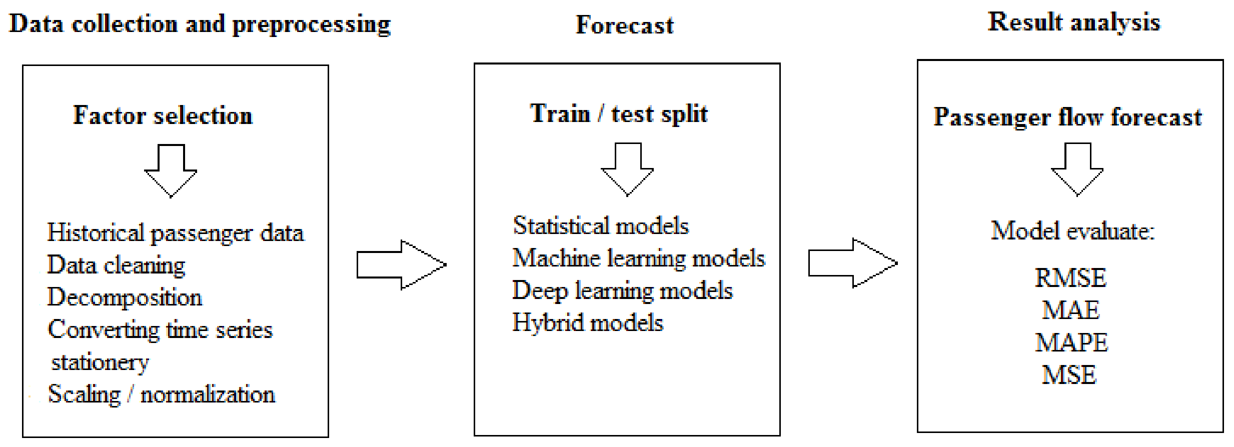


Fig. 4. Passenger flow forecasting process

the *return_sequences=True argument*, ensuring the LSTM cell produces outputs from all time steps. Without this argument, only the output from the last time step is returned. The LSTM client layer processes input in two layers, and the output is compared to training data in batches. To ensure correct sequencing for predictions, unique whole indices are assigned to each value in the data. The re-indexed output facilitates neural network forecasting. During training, data is fed in small batches using the *fit_generator* function from Keras. The first network layer converts integer references to weight vectors, taking the number of passengers and resulting vector size as arguments. The subsequent parameter is the number of iterations per training epoch. Setting it as *len(train_data) // (batch_size * num_steps)* ensures that the entire dataset is processed through the model during each epoch. Likewise, the generator is invoked for the smaller validation dataset with the same iteration argument.

After each epoch, the model assesses the validation data for accuracy. The previously described checkpoint type callback is then incorporated via the callbacks parameter in the *fit_generator function*. To achieve satisfactory results, the model should undergo numerous epochs, and it needs to exhibit a considerable level of complexity. Running it on a CPU might take a substantial amount of time, so it is advisable to utilize a machine with a robust GPU. If a GPU-equipped machine is unavailable, an alternative is to create an Amazon EC2 instance.

Prior to making forecasting, it is essential to label the initial time series. The commonly employed approach for this task is the window sliding algorithm. This algorithm operates on the principle of utilizing the data at time step (*t*+1) as the label for the data at time step *t*, thereby generating a corresponding set of labels.

Following the LSTM layer in neural network, there is a precautionary filter layer. Subsequently, there is a specialized layer for periodic neural networks known as *TimeDistributed*. This layer allocates a distinct layer for each stage in the periodic model. For instance, if the model has 10 time steps, a *TimeDistributed* layer operating on the dense layer will generate 10 separate dense layers, one for each time step. The activation for these dense layers is configured as *Softmax* in the concluding layer of our LSTM model.

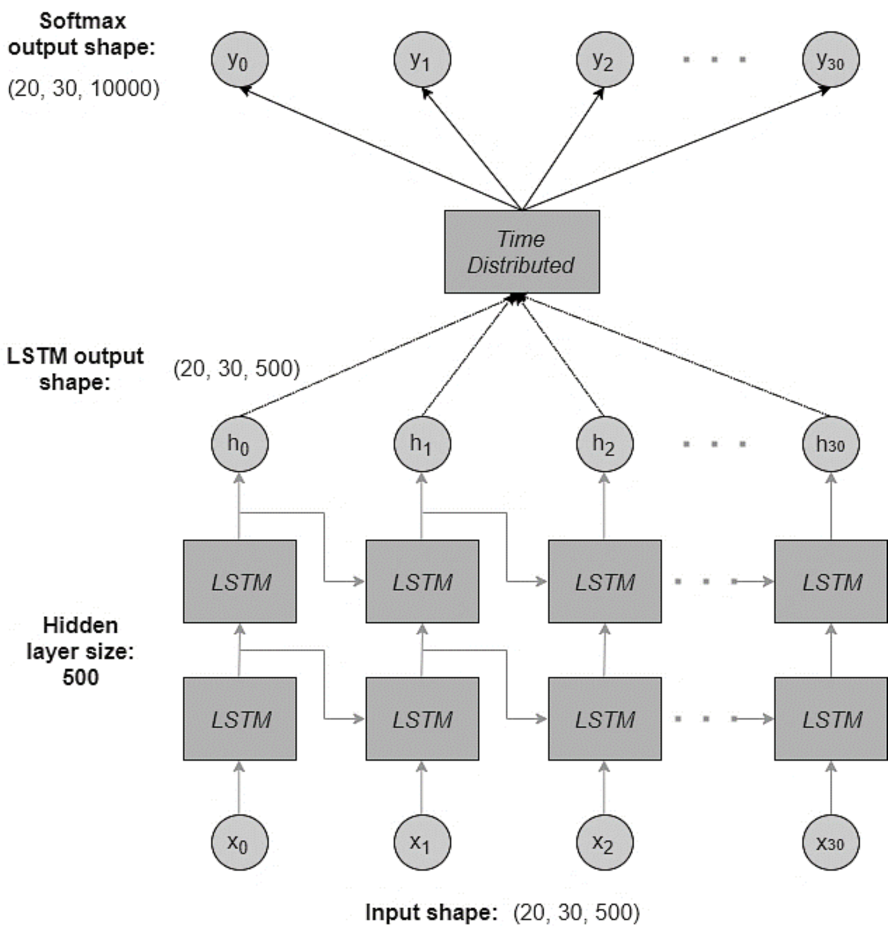


Fig. 5. LSTM neural network model

Ran the model up to 40 epochs and got some initial results. Model parameters for the results below are:

• *num_steps=30*

• *batch_size=20*

• *hidden_size=500*

After 40 epochs, the accuracy of the training dataset was about 98%, and the accuracy of the validation set was about 96%.

To test the LSTM model, we compare projected and actual values in both training and testing sets. The model is reinitialized with trained data, and fictitious data is extracted from the generator to assess predictive accuracy. The relative forecasting error during adaptation is less than 5%, as shown in Table 1 comparing actual and predicted values [9]. The learning algorithm's completion fine-tunes synoptic weighted connections to establish a dependable association between inputs and outputs. During the testing and validation phase, these established connections remain constant, while new inputs are introduced to generate a sequence of outputs. These outputs are then compared with a set of test data representing the actual results.

Table 1

Results of forecasting

Actually values	Predicted values (100 epochs)	Predicted values (400 epochs)
20	20	20
32	28	32
50	46	48
60	62	60
100	100	100
34	70	32
38	38	38

If the disparity between the actual output and the test output exceeds a predefined error threshold, it becomes necessary to adjust the training and reconfigure the neural network. We assess the effectiveness of forecasting methods, including linear trend, logarithmic trend, moving average, Holt-Winters exponential smoothing, and LSTM, in predicting passenger flow. Table 2 illustrates the forecast accuracy achieved by these methods [9].

Table 2

Metrics measuring the precision of predictions employing various methods

Forecasting method	Forecast accuracy
Moving average	0.961
Holt-Winters Exponential Smoothing	0.933
Linear trend	0.930
Logarithmic trend	0.901
Long short-term memory (LSTM)	0.983

Software method of optimizing buses utilization. To improve the quality of passenger service and the competitiveness of the MTE, there is a need to create feedback from passengers. In addition to the main goal – the transportation of passengers, ensuring a comfortable trip for passengers is also a key issue of profit for MTE. Providing comfortable transportation and efficient planning of buses is important. Traditional approaches are based on fixed schedules. Using this software method to process real-time intercity passenger flow data will provide new opportunities and data-driven approaches to meet passenger demand.

When developing this software system, it was assumed that the end user (MTE manager) has the following capabilities:

• add, edit and delete trip records;

• add, edit information about drivers;

• view and edit the schedule;

• predict the number of passengers for the required time, date and route;

• to simulate the most convenient and most profitable rolling stock.

Functional requirements of proposed software are next.

1. The software should provide the prediction result in the corresponding window.

2. The software should provide an opportunity to enter the necessary data using the fields in the corresponding window.

3. The user must be able to select the required date, time and route using the selectors in the corresponding window.

4. Training data should be read from a csv file.

5. In the schedule window, the system should show all planned and executed trips.

6. When clicking on an empty window, a trip should be created and the number of passengers can be predicted at the moment.

The architecture of proposed software system is shown in Fig. 6.

The following components can be distinguished in the software architecture:

1) server for working with records, database and neural network;

2) user interface – a component used by the user to create trips and forecast;

3) neural network – a component of the method that is called from the server for predicting passenger flow.

Flask web framework for Python, which is used for software development, offers numerous benefits for web development. Its simplicity and minimalism make it easy to learn and quick to set up, allowing developers to focus on building features rather than navigating through a complex structure.

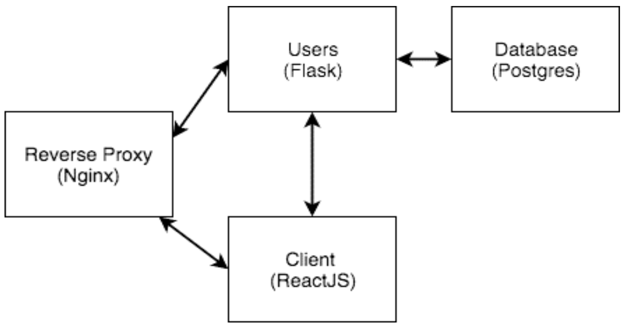
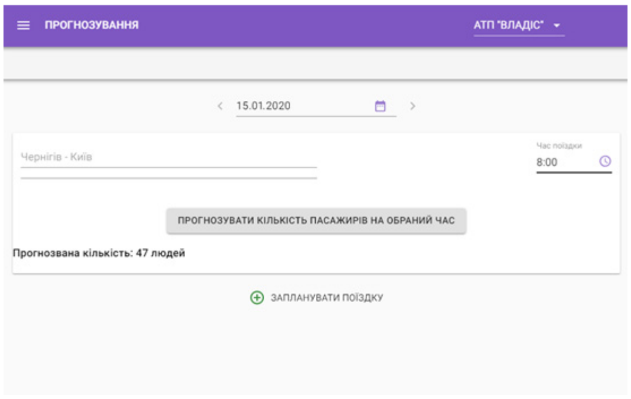


Fig. 6. The architecture of the software system

Flask follows the WSGI standard, ensuring compatibility with various web servers and deployment options. Its modular design encourages the use of extensions, allowing developers to integrate only the components they need for their specific project, promoting scalability and maintainability. Additionally, Flask supports Jinja2 templating, facilitating the creation of dynamic and interactive web applications. Overall, Flask's flexibility, ease of use, and community support make it an excellent choice for developing web applications ranging from small projects to more complex, scalable solutions.

The React JS framework was used to create the client part of the software (Fig. 7). The unidirectional data flow in React simplifies state management, enhancing predictability and maintainability of the codebase. Its extensive ecosystem and a large community contribute to the availability of numerous pre-built components and libraries, streamlining development and reducing the need to build functionalities from scratch. Furthermore, React's ability to seamlessly integrate with other libraries and frameworks and its support for server-side rendering enhance its versatility for building scalable and interactive web applications.



15.01.2020

Чернігів - Київ

Час поїздки
8:00

Прогнозвана кількість: 47 людей

Fig. 7. Forecast window view

PostgreSQL, often referred to as Postgres, is a powerful open-source relational database management system (RDBMS) known for its robust features, extensibility, and adherence to SQL standards. The predicted passenger flow values obtained as a result of using the LSTM model are further used for optimal use of the MTE buses.

The expenses associated with MTE are categorized into two main types: permanent costs, which are fixed and cannot be eliminated (e.g., land lease, taxes), and variable costs linked to the production of goods and services.

The profit of an MTE is calculated as the disparity between total income and the combined expenses of both permanent and variable costs:

$$P = D - V_1 - V_2 , \qquad (1)$$

where D is the amount of income related to passenger flow and the cost of tickets, V_1 is the cost of vehicle maintenance (including drivers' salaries), depending on the choice of buses, V_2 is general production costs (the content of the control apparatus, production premises, social deductions, etc. further), which do not depend on the types of buses operating on the route.

Then the task of maximizing profits ($P \to \max$) is reduced to the problem of minimizing expenses (V_1) for maintenance of the buses, which can be represented as:

$$V_1 = \sum_{j=1}^{4} \left(\left(\frac{s_0}{\tau_0} \right) \cdot \gamma_j + z \right) \cdot K_{1j} \to \min \qquad (2)$$

and restrictions on the number oftransported passengers per hour and the total number of seats in buses:

$$F_1 \le \sum_{j=1}^{4} \left(K_{1j} \cdot N_j \right) , \qquad (3)$$

the average time interval $\overline{\tau}_1$ for buses shall not be greater than the specified τ_1^+:

$$\overline{\tau}_1 = \frac{1}{K_1} \le \tau_1^+ , \qquad (4)$$

where γ_j is the direct material costs (UAH/km), z is the drivers salary (UAH/hour), K_{1j} and N_j ($j = 1,2,3,4$) – accordingly, the number of buses and seats in them, $K_1 = \sum_{j=1}^{4} \left(K_{1j} \right)$ – the types of buses, τ_0 and s_0 are hours and the length of the route.

The act of inputting symbols into the system $a_j = \frac{s_0}{\tau_0} + z$, $x_j = K_{1j}$, signifies the initiation of a linear programming problem. In this context, the task can be formulated with a specific target function:

$$V_1 = a_1 x_1 + a_2 x_2 + a_3 x_3 + a_4 x_4 \to \min , \qquad (5)$$

with the limitations:

$$\begin{cases} c_1 x_1 + c_2 x_2 + c_3 x_3 + c_4 x_4 \ge F_1, \\ x_1 + x_2 + x_3 + x_4 \ge \dfrac{1}{\tau_1^+}, \\ 0 \le x_j \le \mu_j, \, j = 1, 2, 3, 4, \end{cases} \qquad (6)$$

where μ_j – restriction on the availability of the relevant type of buses.

The average bus occupancy rate is determined by the ratios $\overline{\varepsilon}_1$:

$$F_1 = \sum_{j=1}^{4}\left(\varepsilon_{1j} \cdot K_{1j} \cdot N_j\right), \ \overline{\varepsilon_1} = \frac{F_1}{\sum_{j=1}^{4}\left(K_{1j} \cdot N_j\right)} \quad (0 < \overline{\varepsilon_1} \le 1) \ (7)$$

where ε_{1j} – fillfactor for j-th bus. To incorporate this model seamlessly into information technology for the purpose of establishing the most efficient arrangement of buses in MTE, the following scheme is suggested (Fig. 8) [10].

The software employs an LSTM model to predict passenger flow for a chosen route on a given day and recommends an optimal vehicle schedule for the specified hour (Fig. 9).

Conclusions and future work. This research proposes a software method for data collection and forecasting of passenger flow on intercity routes of Ukraine using the LSTM neural network. Studies have shown the effectiveness of using the LSTM neural network in comparison with known statistical forecasting methods. The proposed software system allows to create an optimal plan for the use of the car fleet based on the forecast value of the passenger flow. The LSTM model can be effective for medium-term passenger flow forecasting on long-distance routes because it can take into account long-term dependencies in time series, such as seasonality and trends. However, it is important to note that LSTM can have a tendency to overtrain on small amounts of data, and it can be computationally expensive, especially with large data sizes. In addition, it may not work effectively under unstable or changing conditions that may occur in the field of transportation forecasting.

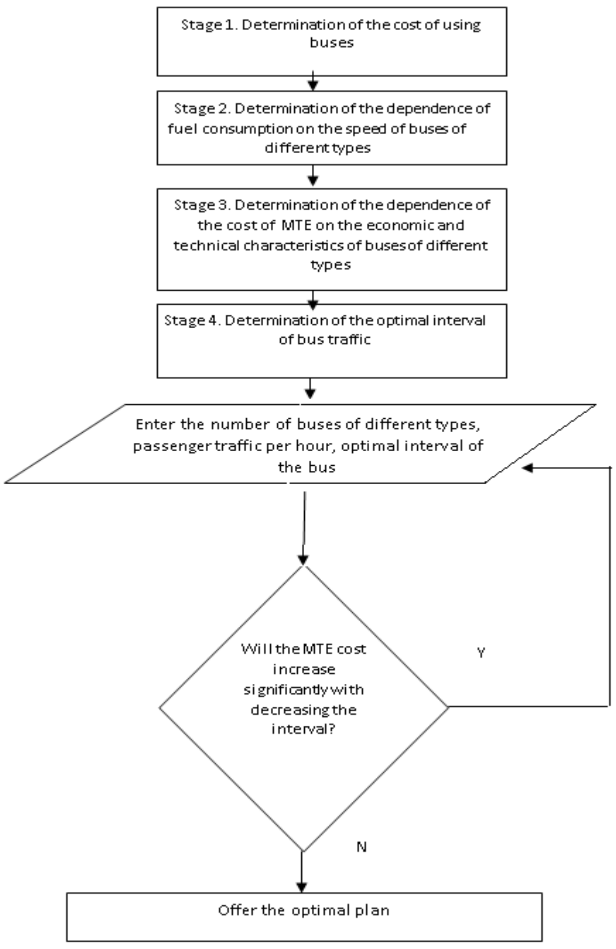
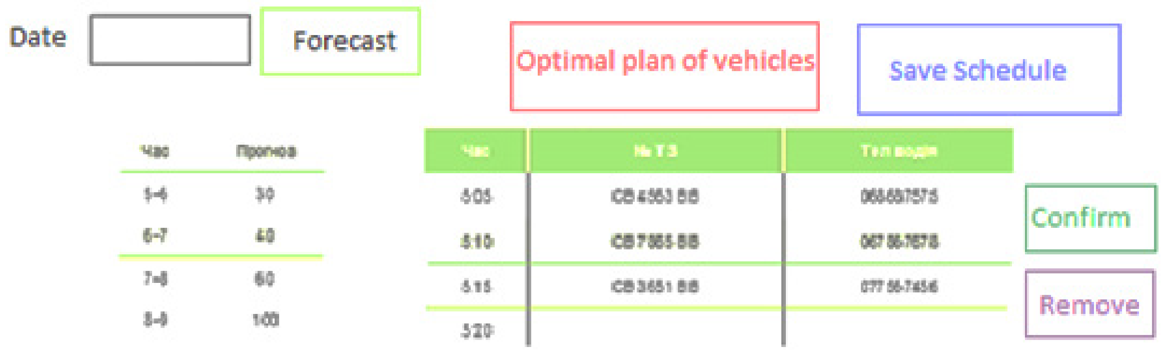

Stage 1. Determination of the cost of using buses
Stage 2. Determination of the dependence of fuel consumption on the speed of buses of different types
Stage 3. Determination of the dependence of the cost of MTE on the economic and technical characteristics of buses of different types
Stage 4. Determination of the optimal interval of bus traffic
Enter the number of buses of different types, passenger traffic per hour, optimal interval of the bus
Will the MTE cost increase significantly with decreasing the interval?
Y
N
Offer the optimal plan

Fig. 8. Scheme of information technology offering the optimal structure of vehicles



Час	Прогноз
5-6	30
6-7	40
7-8	60
8-9	100

Час	№ ТЗ	Тел водія
5:05	CB 4353 BB	0666657575
5:10	CB 7885 BB	067 55 7878
5:15	CB 3651 BB	077 55 7456
5:20		

Fig. 9. Finding the forecast value of the passenger flow on the selected date and selecting the optimal vehicle plan for the selected hour

Bibliography:

1. Bandara K., Bergmeir C., Smyl S. Forecasting across time series databases using recurrent neural networks on groups of similar series: a clustering approach. *Expert Syst Appl*. 2020. Vol. 140, № 3. P. 112896. https://doi.org/10.1016/j.eswa.2019.112896.
2. Banerjee N., Morton A., Akartunalı K. Passenger demand forecasting in scheduled transportation. *European Journal of Operational Research*. 2020. Vol. 286, № 3. P. 797–810. https://doi.org/10.1016/J.EJOR.2019.10.032.

3. Bontempi G., Ben Taieb S., Le Borgne Y.A. Machine learning strategies for time series forecasting. *Lecture Notes in Business Information Processing*. 2013. Vol. 138. P. 62–77. https://doi.org/10.1007/978-3-642-36318-4_3.

4. Chen J.H., Wei H.Y.H., Chen C.L., Wei H.Y.H., Chen Y.P., Ye Z. A practical approach to determining critical macroeconomic factors in air-traffic volume based on K-means clustering and decision-tree classification. *Air Transp. Manag.* 2020. № 82. P. 101743. https://doi.org/10.1016/j.jairtraman.2019.101743.

5. Gunter U., Zekan B. Forecasting air passenger numbers with a GVAR model. *Annals of Tourism Research*. 2021. № 89. P. 103252. https://doi.org/10.1016/j.annals.2021.103252.

6. Kim S. Forecasting short-term air passenger demand using big data from search engine queries. *Autom Constr.* 2016. Vol. 70. P. 98–108. https://doi.org/10.1016/J.AUTCON.2016.06.009.

7. Milenkovic M., Švadlenka L., Melichar V., Bojovic N., Avramovic Z. SARIMA modelling approach for railway passenger flow forecasting. *Transport*. 2016. Vol. 33. P.1113–1120. https://doi.org/10.3846/16484142.2016.1139623.

8. Zaynab Qasim, Abdul-Razzak Ziboon and Khaldoon Falih. TransCad analysis and GIS techniques to evaluate transportation network in Nasiriyah city. *MATEC Web of Conferences.* 2018. Vol. 162, Article 03029. https://doi.org/10.1051/matecconf/201816203029.

9. Hu Z., Dychka I., Oleshchenko L., Kukharyev S. Applying Recurrent Neural Network for Passenger Traffic Forecasting. *Advances in Intelligent Systems and Computing*. 2020. Vol. 938. P. 68–77. https://doi.org/10.1007/978-3-030-16621-2_7.

10. Medvedev M.G., Oleschenko L.M. The optimal control models of interurban bus transport. *Electronics and control systems*. 2014. Vol. 1 (39). P. 85-90.

Олещенко Л.М. ПРОГНОЗУВАННЯ МІЖМІСЬКОГО ПАСАЖИРОПОТОКУ ТА ОПТИМАЛЬНЕ ВИКОРИСТАННЯ АВТОБУСІВ АТП ЗА ДОПОМОГОЮ НЕЙРОННОЇ МЕРЕЖІ LSTM

У статті представлено аналіз наявного програмного забезпечення та програмних платформ інтелектуального аналізу даних, які використовуються для прогнозування пасажиропотоку між містами. Наведено основні статистичні методи, які використовуються для прогнозування пасажиропотоку між містами.

У цьому дослідженні запропоновано програмний метод збору даних та прогнозування пасажиропотоку на міжміських маршрутах України з використанням нейронної мережі LSTM. Для навчання та тренування нейронної мережі використано дані про пасажиропотоки між містами Чернігів та Київ з 2011 по 2015 роки. Для розроблення програмного забезпечення, зокрема, модуля прогнозування, використано мову програмування Python та вебфреймворк Flask. Дослідження показали ефективність використання нейронної мережі LSTM у порівнянні з такими статистичними методами прогнозування як лінійний тренд, логарифмічний тренд, ковзне середнє, експоненціальне згладжування Холта-Вінтерса. Запропоноване програмне забезпечення дозволяє скласти оптимальний план використання автопарку на основі прогнозного значення пасажиропотоку та моделі математичного програмування для заданої структури рухомого складу АТП. Модель LSTM може бути ефективною для середньострокового прогнозування пасажиропотоку на міжміських маршрутах, оскільки вона може враховувати довгострокові залежності часових рядів, такі як сезонність і тенденції. Однак важливо зазначити, що LSTM може мати тенденцію до перенавчання на малих обсягах даних, що може бути дорогим з точки зору обчислень, особливо з великими розмірами даних. Крім того, LSTM може працювати не ефективно за нестабільних або мінливих умов, які можуть мати місце у сфері прогнозування пасажирських перевезень.

Ключові слова: *програмне забезпечення, мова програмування Python, машинне навчання, нейронна мережа LSTM, прогнозування, програмні платформи інтелектуального аналізу даних, пасажиропотік, оброблення даних, міжміські пасажирські перевезення, АТП, оптимізація, автобуси, математичне програмування.*