Arrival Time Prediction for Public Transport Using LSTM-Based Neural Networks

Cristian DINU, Mihai DOINEA
Bucharest University of Economic Studies, Romania
cristian.dinu96@gmail.com, mihai.doinea@ie.ase.ro

Neural networks have recently found widespread application across various domains within IT software infrastructure. This study focuses on specific neural network architectures incorporating Long Short-Term Memory (LSTM) layers and their variations, aiming to effectively capture the dynamic, nonlinear nature of traffic data. LSTM networks are well-suited for learning long-term dependencies in sequential data, making them particularly effective for time series prediction tasks. To evaluate the performance of different LSTM-based architectures, we utilize a dataset comprising bus logs from New York City collected over a four-month period. The experimental results indicate that LSTM architectures demonstrate strong predictive capabilities and are well-suited for modeling complex temporal patterns in traffic data.

Keywords*: LSTM, Neural Networks, Machine Learning, Time estimation*
DOI: 10.24818/issn14531305/29.3.2025.03

1 Introduction

The global push toward developing smart cities has significantly increased the demand for intelligent, data-driven public transportation systems. A critical component of this transformation is the ability to accurately estimate vehicle arrival times, enabling dynamic scheduling and real-time updates. Accurate time estimation allows commuters to better plan their journeys, leading to more efficient use of time and improved overall quality of life in urban environments.

In this context, Long Short-Term Memory (LSTM) neural networks have emerged as a powerful tool for time series analysis due to their ability to capture long-range dependencies in sequential data. Prior research has demonstrated the effectiveness of LSTM-based models in forecasting traffic speed and predicting transportation patterns [1]. Other studies have explored alternative approaches, such as fuzzy neural models (FNM) for urban traffic flow prediction [2], and have shown that LSTM networks can outperform traditional recurrent neural networks (RNNs) in handling predictable time series data [3]. More recently, advanced LSTM-based architectures have been proposed that surpass earlier state-of-the-art models in time series classification tasks [4].

This study investigates several LSTM-based neural network architectures with the goal of enhancing arrival time predictions for public transport vehicles. By improving the accuracy of these predictions, such models can contribute to smarter transportation systems and, ultimately, to the broader objectives of smart city development.

2 Arrival Time in a Smart City Context

In the context of a smart city, the need of good estimation regarding the arrival time of public transport vehicle has made its appearance. The public transport is one of the key parts of a developed city, or even an in developing city. In my opinion, the public transport infrastructure is one of the main possibilities that can launch a city on a fast-forward trend regarding the development. It has implications for each and every person that lives or travels in the city, because exact time estimations is the key for investing the most important resource, time, in the developing of the entire collective.

Everyone wants to know exactly how long it takes to the travel from one location to another, no matter how traffic jams will influence the journey. By tacking into

considerations all types of factors that may or may not affect the route to which a public transport vehicle is running, there should be a possibility to provide good time estimations for the road ahead. And if this problem is solved, then more and more people will choose the public transport in the detriment of the private vehicle, because they will be sure that the planned schedule will be mostly on time.

To find a solution for estimating the time for a route, in a specific line, some methods must be researched in order to do a time series analysis. From the statistical point of view, there are multiple methods of analyzing and estimating time series, based on the linearity of the process (linear or non-linear process). By thinking at what influences a travel route, we can find multiple independent factors: *Time of day, Day of week, Month and season, Population density, Population activities, Holidays, Weather conditions, Political events, Previous traffic characteristics.*

By looking at these possible factors, it is clearly that this is a non-linear process and advanced methods should be used for estimating future time intervals.

One paper from 2015 [1], proposes 2 approaches that could be used for estimating in the context of a time series similar with the one analyzed in this paper:

- "Parametric approach", that uses key characteristics as parameters from the traffic, like speed, density and flow. For this method, the most widely used method is Autoregressive Integrated Moving Average (ARIMA). This approach is fitted to time series data, by providing a better understanding of the data and that can predict specific future points.
- "Nonparametric approaches", that takes into consideration the whole state of the process and estimates some weights used in the future estimation. Based on this approach, there are multiple techniques, that includes the following:
    - Kalman filter – The main idea is to find the optimal solution by minimizing the variance and thus exhibiting the superior capability for online calibrations and learning.
    - Support vector machine (SVM) – This method essence is to perform linear regression within a high-dimensional space where the data was mapped through non-linear relationships.
    - Artificial neural network (ANN) – Multiple advantages recommends this method as a popular one for estimating the traffic times. Some of those advantages are the flexibility of model structure, handling multi-dimensional data, the ability to learn and adapt, and the strong generalization. Still, there are present some disadvantages, at least for the traditional RNNs. They weren't "able to train on time series with long time lags", because, usually, "They rely on the predetermined time lags to learn the temporal sequence processing, but it is difficult to find the optimal time window size in an automatic way" [1].

Although the traditional RNNs have some disadvantages regarding time series, they were highly developed and improved in order to be able to handle this type of data. The LSTM NN is a result of this improvement that manages to overcome all of its previous generations neural networks disadvantages. The main advantage is that an LSTM layer is composed from cells that can maintain the state overtime and thus influencing future states.

3 Neural Network Solutions

Based on the choice of approach for estimating the time for a public transport route, neural networks, the need of a programming language that is highly oriented on this type of calculations and optimizations is required in order to obtain good results. The widely choice for scientific computing [5], is the programming language named Python.

Having Python as the programming language, we can benefit from the external modules to implement neural networks, Figure 1.

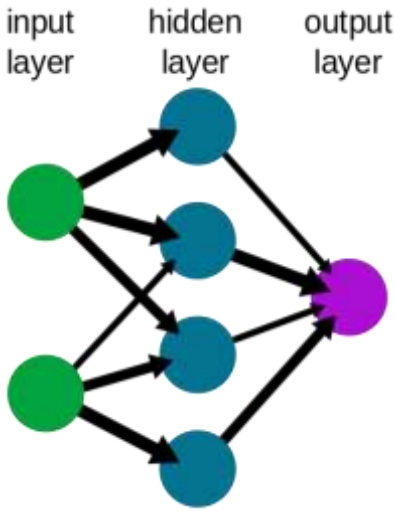

Fig. 1. Simplified view of a feedforward artificial neural network. Source Wikimedia Commons

For a long time, it has been the first choice in scientific papers regarding neural networks or machine learning algorithms, because of its simplified syntax and the already existent external libraries. It is safe to say that Python is the most used language for proof of concept regarding IT academic research.

A neural network, also named artificial neural network, is mainly a circuit of neurons inspired from the biological neural network. In the network are present a lot of connections between these neurons and these are modeled by weights that categorizes the stimulatory data into 2 branches: excitatory connection (if the weight is positive) and inhibitory connection (if the weight is negative). The activity that takes place into the network is referred as a linear combination for which output is controlled by the activation function. Al these characteristics are merged into layers that can be stacked for solving Artificial Intelligence (AI) problems.

Although there are multiple types of neural networks developed, this research will be based on LSTM NN, using an Rectified Linear Unit (ReLU) activation function. These types are used in different domains, because of the key characteristics that they provide. One of it is that they manage datasets with long period of times really well. In a paper from 2016 [6], the LSTM network is used for named entity recognition to "*automatically detect word and character-lever features using a hybrid bidirectional LSTM and CNN architecture*". Activity recognition time series classification is another type of work in which this type of neural network has been used, as described in [7]. Another paper, [1], uses the same type of architecture to predict traffic speeds, while a study from 2015 uses convolutional LSTM networks "*to predict the future rainfall intensity*" [8].

A good definition of long short-term memory neural network is provided by Jason Brownlee in his work [7]: "*LSTM network models are a type of recurrent neural network that are able to learn and remember over long sequences of input data. They are intended for use with data that is comprised of long sequences of data, up to 200 to 400 time steps*". One important characteristic about the artificial neural networks is that they are stochastic, meaning that for multiple trainings on the same data and the same configuration, a different model can be created.

3.1 Dataset

The dataset is the base resource required in order for any neural network to obtain acceptable results. If the provided dataset is correctly structured, it enables the possibility of the created neural network architecture to optimize the weights for providing the best results. For this research, the dataset consists in the logs provided by the NYC MTA buses data stream service and it consists in various details about the locations, Figure 2, schedules and other information transmitted and registered live. In any training, the dataset is surely to require a preprocessing step before it can be feed into the network. In this way, new composed features may appear, or, by applying some conversions, the features can be normalized. In this way they can describe in an improved manner the process.

Fig. 2. B46-SBS Route. Inspired from http://web.mta.info/ and https://www.openstreetmap.org/

The found dataset is provided on Kaggle [9]. There are 17 provided columns for each entry and the whole dataset contains data for 242 lines and for over 170 different vehicles. In total there are over 20 million reported entries that can be analyzed. From all the provided data, only one line will be selected in order to apply a processing step and, in the end, to determine the improved features. The line is B46-SBS.

After the preprocessing steps, the final dataset is described as follows:

- *RecordedDayOfWeek* – The day of the week according to ISO 8601.
- *RecordedDayMinutes* - The minutes of the day when the entry was reported.
- *DirectionRef* – The direction of the vehicle (0 – Dekalb Av Via Utica, 1 – Kings Plaza Via Utica).
- *RemainingRoad* – The percentage of the entire route that remains to be completed.
- *LateMinutes* – The latency based on the scheduled arrival to the last station.
- *RouteStartTime* – The minutes elapsed from the start of the current route.
- *ScheduledArrivalIn* – The scheduled arrival to the end of the route.
- *RouteEndTime* – The minutes until the actual arrival time to the end of the route, extrapolated from the available data.

3.2 LSTM

The LSTM NN architecture is defined as a classic LSTM layer, followed by a Dense layer. The activation layer is represented by the rectified linear unit, for the first layer, and the linear activation function for the last layer in order to match the loss function applied. The input shape is represented by 2 timesteps and 7 features.

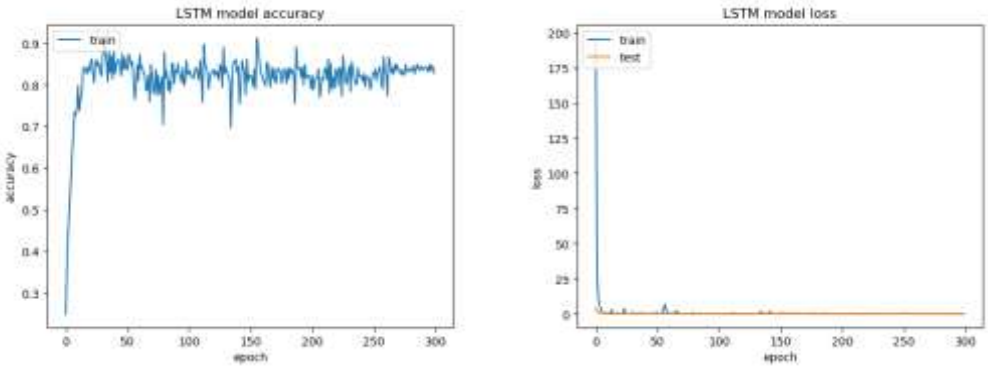

Fig. 3. LSTM accuracy / loss preview

After the training has finished, the model was evaluated on approximately 49000 entries with the following results: Accuracy 83.20%, Mean square error 0.1345, Validation mean squared error 0.0949, Figure 3.

3.3 BLSTM

The BLSTM NN was designated as similar as possible with the LSTM NN in order to gain comparable results only based on the difference of the core layer. So, the

architecture is represented by a Bidirectional Long Short-Term Memory layer, followed by a Dense one. The activation functions are the same as for the LSTM example: rectified linear unit plus linear for the last layer.

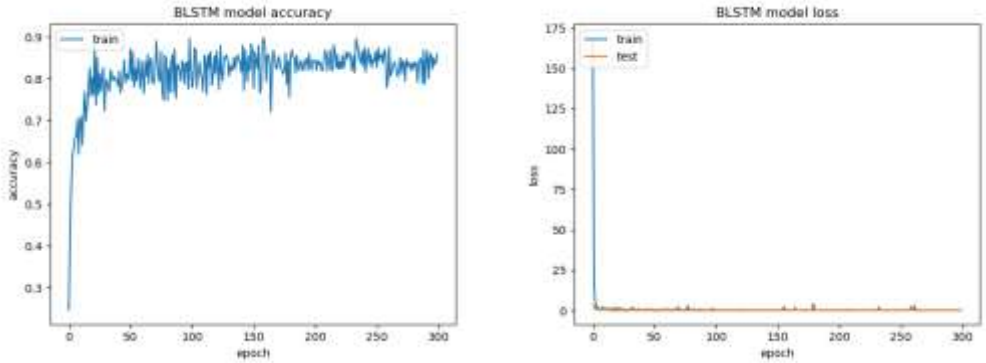

Fig. 4. BLSTM accuracy / loss preview

After the training has finished, the model was evaluated with the following results: Accuracy 85.82%, Mean squared error 0.1219, Validation mean squared error 0.1426, Figure 4.

3.4 Stacked LSTM

A Stacked LSTM NN implies the existence of multiple Long Short-Term Memory layers into the network. As a result, this type of architecture becomes a Deep Neural Network. The current example will have 3 LSTM layers. The first layer will receive the input as three-dimensional entries and will output the values in the same structure, until the last specific layer. The last layer is represented by a Dense layer and the most used activation function is the rectified linear unit.

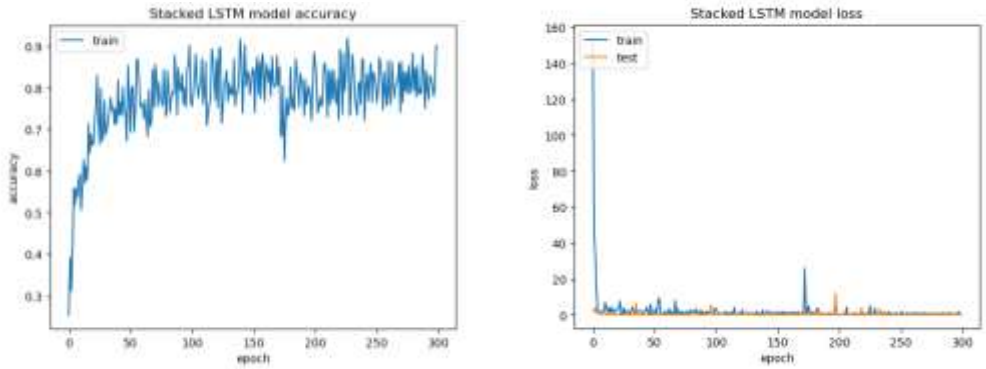

Fig. 5. Stacked LSTM accuracy/ loss preview

The model has been evaluated and the results were as follows: Accuracy 89.82%, Mean squared error 0.1349, Validation mean squared error 0.0976, Figure 5.

3.5 Stacked BLSTM

Like for the Stacked LSTM, the Stacked Bidirectional Long Short-Term Memory Neural Network is composed from multiple layers. The increase has been gradual with one layer per training until the optimal solution has been found. For the current dataset, the tested architecture with best results contains 3 layers of BLSTM, followed by one Dense layer.

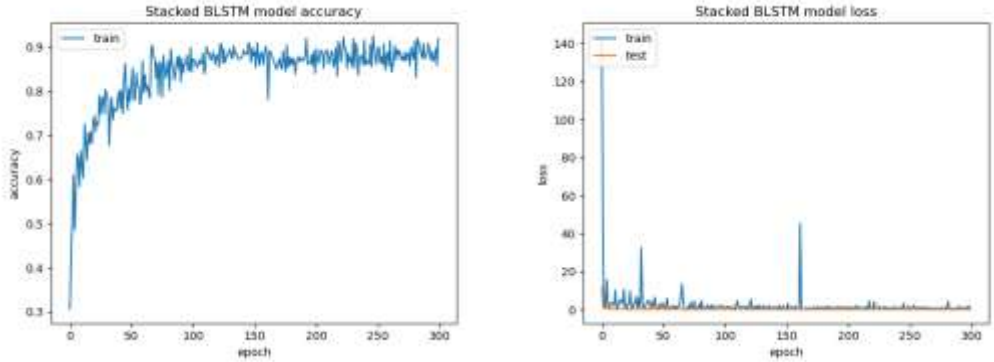

Fig. 6. Stacked BLSTM accuracy / loss preview

After the training, the results are as follows: Accuracy 91.84%, Mean squared error 0.1761, Validation mean squared error 0.0936, Figure 6.

3.6 Mixed Deep Neural Network

The mixed deep neural network, for this example, is composed from multiple layers of both LSTM and BLSTM types. They are stacked one over the other, and as the final layer, a Dense one. After analyzing multiple architectures, the one with best results is composed from one bidirectional long short-term memory layer, on top of it a simple layer, and then another bidirectional one. For the first three layers is used the rectified linear unit activation function.

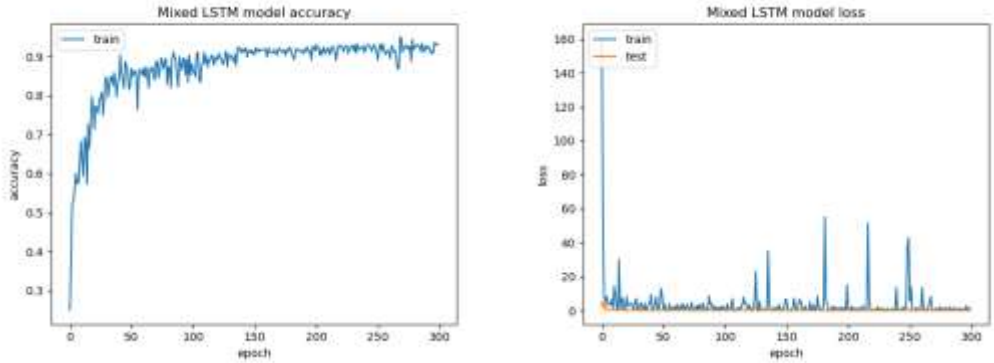

Fig. 7. Mixed LSTM accuracy / loss preview

This model results are Accuracy 93.05%, Mean squared error 0.1078, Validation mean squared error 0.1056, Figure 7.

4 Results

When comparing the *LSTM* and the *BLSTM* architectures, both surpassed a threshold of 0.8. Both networks were trained for 300 epochs with a batch size of 200 and 25% validation data. The first architecture had a decrease in the accuracy around the 50^{th} epoch. After that step it remained relatively constant. On the other hand, the bidirectional long short-term memory neural network with a single layer, rapidly increased to an accuracy of 0.8, and then continued to improve with a steadier peace for the whole process. Looking at the loss graphics, we can say that both of them managed to train based on the current dataset. Both network types were created as similar as possible, with the same configuration. The single difference being the core layer. As a result, the Bidirectional Long Short-Term Memory Neural Network outperformed the other network with 2.62% for the accuracy and a better loss.

Looking at the *Stacked LSTM vs Stacked BLSTM*, the accuracy in the long short-term memory network has an increased variation, then in the bidirectional neural network. In

this case the bidirectional network reaches an accuracy level above 0.8 and continued with a steady peace. Seems like both networks went through a short drop in the accuracy around the 150th epoch, and they are in concordance with the loss graphic. Regarding the mean squared error for both architectures, it was tending to the value zero. Both networks were modeled to receive an input shape of 2 timesteps and 7 features. Like in the previous case, the single difference of these networks are the core layers. For this example, the architecture that prevailed is the simple Stacked BLSTM NN, which had an accuracy greater with 2.02% than the Stacked LSTM NN.

In the overall results, the mixed deep neural network, composed from layers of both types, bidirectional and simple LSTM, has the best result. It reached a level of 93.05% accuracy and a mean squared error of 0.1078, figure 8.

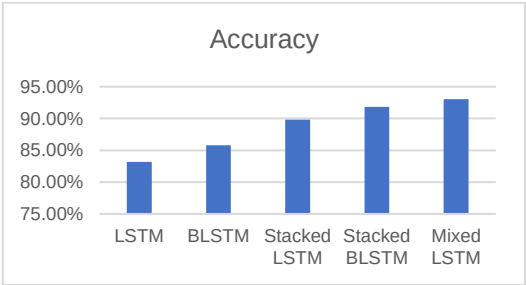

Fig. 8. Neural networks accuracy comparison

As can be seen in the accuracy comparison graphic, the mixed long short-term memory neural network has the best results. All of the trained networks have exceeded the threshold of 80% accuracy and the simple neural networks are the last based on the accuracy results. Most certainly because one single layer can't fully learn the hidden bias from the dataset.

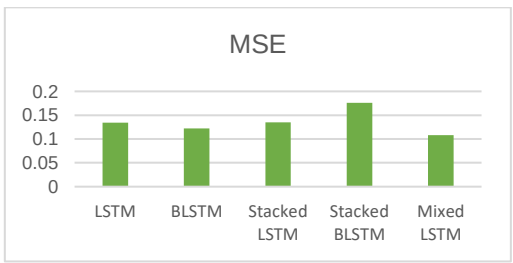

Fig. 9. Neural networks mean squared error comparison

The mean squared error, Figure 9, used as the cost function for the training model is one of the most important indicators for regression problems. The mixed long short-term memory neural network has the lowest value, 0.1078 and it is in concordance with the accuracy.

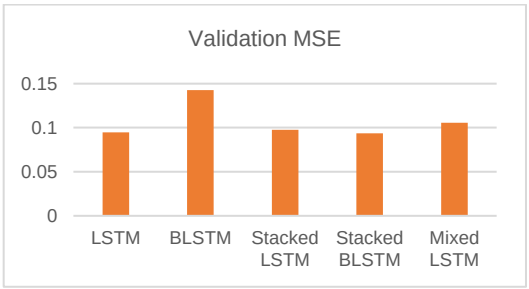

Fig. 10. Neural networks validation mean squared error comparison

Taking a look at the graphical review of the validation cost function, figure 10, we can see that for all the trained models they were almost zero. The lowest value is for the Mixed LSTM. The value close to 0 for the cost function means that the trainings has converged to a good result.

5 Conclusion

This research demonstrates the effectiveness of Long Short-Term Memory (LSTM) neural network architectures for estimating public transport arrival times based on time series data. Among the evaluated models, the mixed deep neural network combining both standard and bidirectional LSTM layers achieved the highest performance. All proposed architectures successfully converged and produced reliable results, even when using a limited dataset. The findings suggest that further improvements could be achieved by enhancing the dataset quality and diversity. Currently, the input features are limited to the day of the week and time of day. Expanding the dataset to include additional context such as season, weather conditions, holidays, and special events could significantly improve model accuracy and generalization. Future work should also focus on integrating real-time data streams and testing the scalability of these models in larger, more complex urban environments.

References

[1] M. Xiaolei, T. Zhimin, W. Yinhai, Y. Haiyang and W. Yunpeng, "Long short-term memory neural network for traffic speed prediction using remote microwave sensor data," Transportation Research Part C: Emerging Technologies, vol. 54, pp. 187-197, 2015.

[2] Y. Hongbin, W. S.C., X. Jianmin and W. C.K., "Urban traffic flow prediction using a fuzzy-neural approach," Transportation Research Part C: Emerging Techologies, vol. 10, no. 2, pp. 58-98, 2002.

[3] F. A. Gers, D. Eck and J. Schmidhuber, "Applying LSTM to Time Series Predictable Through Time-Window Approaches," in Neural Nets WIRN Vietri-01. Perspectives in Neural Computing, London, Springer, 2002, pp. 193-200.

[4] F. Karim, S. Majumdar, H. Darabi and S. Harford, "Multivariate LSTM-FCNs for time series classification," Neural Networks, vol. 116, pp. 237-245, 2019.

[5] T. E. Oliphant, "Python for Scientific Computing," Computing in Schience & Engineering, vol. 9, no. 3, pp. 10-20, 2007.

[6] J. P. Chiu and E. Nicholas, "Named Entity Recognition with Bidirectional LSTM-CNNs," The MIT Press Journals, vol. 4, pp. 357-370, 2016.

[7] J. Brownlee, "How to Develop RNN Models for Human Activity Recognition Time Series Classification," 24 09 2018. [Online]. Available: https://machinelearningmastery.com/how-to-develop-rnn-models-for-human-activity-recognition-time-series-classification/. [Accessed 2020 03 14].

[8] S. Xingjian, C. Zhourong, W. Hao, Y. Dit-Yan and W. Wai-Kin, "Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting," 2015.

[9] M. Stone, "New York City Bus Data," 05 18 2018. [Online]. Available: https://www.kaggle.com/stoney71/new-york-city-transport-statistics. [Accessed 15 04 2020].

Mihai DOINEA holds a Ph.D. in Economic Informatics, awarded in 2011 for his thesis titled Security Optimization in Distributed Applications. He is currently an Associate Professor at the Faculty of Cybernetics, Statistics, and Economic Informatics. Dr. Doinea has published over 70 scientific papers in internationally recognized journals and conferences, focusing on security-related topics. His research interests include security, data structures, mobile applications, IoT, biometrics, and Java card technology — areas in which his work has been validated by the broader scientific community.

Cristian DINU graduated in 2018 from the Faculty of Cybernetics, Statistics, and Economic Informatics. He holds a master's degree in IT&C Security from the Bucharest University of Economic Studies, faculty of Cybernetics, Statistics, and Economic Informatics. His professional experience includes hands-on involvement in software architecture and development for projects focused on document digitization and human resources solutions.