

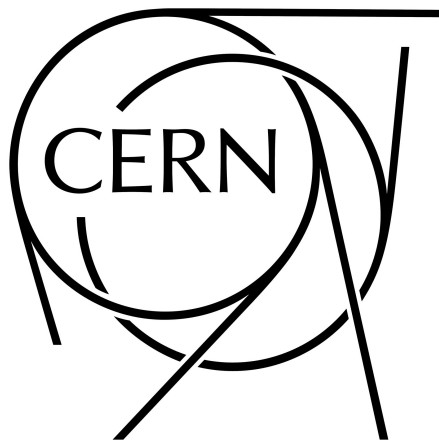
# Network Traffic Prediction with Deep Learning-Based Encoder-Decoder Algorithms to Improve the Network Controller NOTED

**Elisabetta Schneider**

Supervisor: Carmen Misa Moreira

Co-Supervisor: Edoardo Martelli

A Report Presented as Part of the  
CERN Summer Student Program 2024



CERN - IT Department  
European Council for Nuclear Research  
Conseil Européen pour la Recherche Nucléaire  
Switzerland

## Acknowledgements

I would like to express my deepest gratitude to CERN for providing me with the opportunity to participate in the Summer Student Program. This experience has been incredibly enriching, both personally and professionally. My sincere thanks to the entire Summer Student Program team for their dedication and support, especially Ms. Preobrazenska, whose guidance was invaluable throughout my time at CERN.

I am particularly grateful to my supervisor, Carmen Misa Moreira, for her mentorship and encouragement and to Edoardo Martelli for his continuous support. Their expertise and guidance were instrumental in the success of my project, and I am truly thankful for the opportunity to work alongside such talented professionals.

A heartfelt thank you to the entire IT department at CERN, whose welcoming spirit and knowledge-sharing made this experience truly exceptional. I am also deeply grateful to my parents, Yulia and Reinhard, for their unwavering support. Last, but not least, a special thanks to my sister, Alina, and my boyfriend, Liam, for always believing in me. Thank you!

This journey would not have been possible without the support and love of all these incredible individuals, and I am profoundly thankful for their contributions to my success.

## Abstract

During my summer internship at CERN, I contributed to the Network Optimised Transfer of Experimental Data (NOTED) project within the IT department. My work focused on enhancing the accuracy and computational efficiency of network traffic forecasting by implementing advanced Encoder-Decoder machine learning algorithms, including Seq2Seq models, Autoencoders, and Transformers. These algorithms were tested for their ability to predict network traffic and optimize data transfers across the LHCONE (Large Hadron Collider Open Network Environment) and LHCOPN (Large Hadron Collider Optical Private Network) links. My contributions helped improve NOTED's ability to forecast traffic more accurately and efficiently, thus supporting CERN's broader goal of optimizing data transfers for high-energy physics research.

# 1 Introduction

The Large Hadron Collider (LHC) at CERN is one of the most complex scientific instruments ever built, producing vast amounts of data that are crucial for high-energy physics research. This data, generated by various experiments, is transferred across a global network that connects CERN with research centers worldwide. However, the sheer volume of data can lead to network congestion, particularly during peak transfer times, which poses significant challenges to maintaining efficient and reliable data flows. This congestion can result in underutilized alternative network paths while others are overwhelmed, leading to inefficiencies that can delay scientific discoveries.

To address these challenges, CERN launched the Network Optimised Transfer of Experimental Data (NOTED) project in 2020. The primary goal of NOTED is to optimize network utilization by dynamically reallocating bandwidth and managing data transfers in real-time. Accurate network traffic forecasting is central to this objective, as it enables the prediction of congestion patterns and the efficient rerouting of data. However, traffic forecasting within the NOTED framework presents challenges, especially in predicting short bursts of congestion that may not warrant significant rerouting but can still impact data transfer efficiency [1].

My summer internship at CERN's IT department focused on enhancing the predictive capabilities of NOTED by integrating advanced machine learning algorithms. Previous research has already demonstrated the potential of Long Short-Term Memory (LSTM) networks, CNNs (Convolutional Neural Networks) and hybrid models such as Conv-LSTM and CNN-LSTM, in improving traffic forecasting [2]. A next step would be to explore different architectures based on the previous work to potentially increase efficiency and accuracy. My work centered on implementing various LSTM and other neural network based Encoder-Decoder architectures, including Seq2Seq models, Autoencoders, and Transformers, to evaluate their effectiveness in predicting network traffic within the NOTED framework.

## 2 Methodology

During my internship, I systematically explored different deep learning models with unique strengths in handling time series data to enhance NOTED's ability to forecast network traffic. The methodology involved several key steps: Selection and implementation of different machine learning algorithms, dataset preprocessing, model testing and training, and evaluation of computational efficiency.

### 2.1 Machine Learning Algorithms

This section provides a detailed overview of the machine learning algorithms I studied during my internship, all based on Encoder-Decoder architectures. These algorithms are schematically illustrated in Figure 1, and all models were implemented using Keras in TensorFlow.

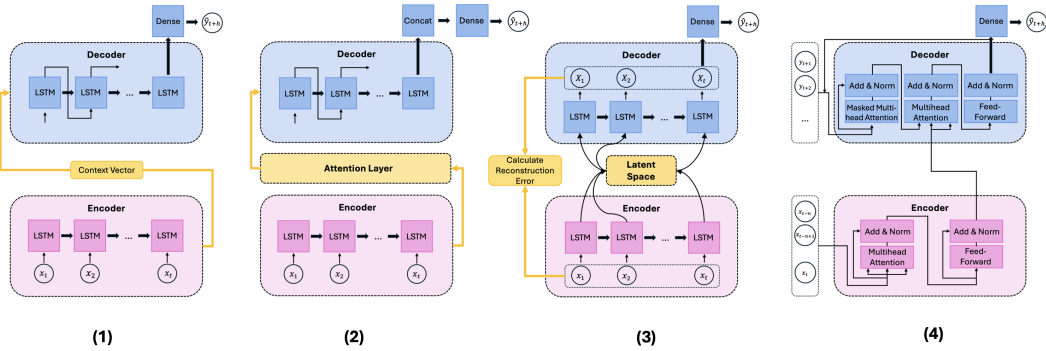


Figure 1: Schematic representation of the different machine learning algorithms used for time series forecasting: (1) Seq2Seq Encoder-Decoder model (2) Seq2Seq Encoder-Decoder model with an additional attention layer (3) Autoencoder (4) Encoder-Decoder Transformer

**Seq2Seq Encoder-Decoder Model** The Seq2Seq model comprises an encoder that compresses the input sequence into a fixed-length context vector and a decoder that generates the output sequence from this vector. However, this approach can be limiting for long sequences, as compressing all essential information into a single vector may lead to a loss of important details and reduced output accuracy [3].

To address this, I incorporated an attention mechanism (AM), allowing the decoder to dynamically focus on the most relevant parts of the input sequence during output generation. This attention mechanism significantly improves prediction accuracy, especially when future outputs depend on specific, potentially non-contiguous, past-time steps [4] [5].

The decoder and encoder are based on LSTMs.

**Autoencoders** An Autoencoder is a type of neural network designed to capture key patterns within sequences by compressing them into a latent space, a lower dimensional space, that encodes essential dependencies. It minimizes the difference between the original input and its reconstruction, effectively learning the critical features of the data. When applied to historical time series data, Autoencoders can reconstruct sequences and capture important temporal information automatically [6].

Similar to the Seq2Seq model, an Autoencoder consists of an encoder and a decoder. However, instead of using a context vector, the Autoencoder compresses the input into a lower-dimensional latent space from which the decoder reconstructs the original data [7]. In a modified version, I adapted the decoder to predict future values rather than simply reconstructing the input sequence. This approach involves training the model with historical data as inputs and corresponding future sequences as targets, enabling the Autoencoder to generate accurate future predictions [8].

The decoder and encoder are also based on LSTMs.

**Transformers** Transformers, introduced by Vaswani et al. in his paper "Attention is all you need" in 2017 [9], are deep learning architectures that use self-attention mechanisms to process sequences, unlike traditional RNNs (Recurrent Neural Networks) and LSTMs. The self-attention mechanism allows Transformers to capture complex dependencies, including temporal ones, by focusing on the most relevant parts of the input sequence, which is particularly useful for tasks like predicting future values. Additionally, Transformers are well-suited for time series data due to their use of positional encoding, which helps the model understand the order of data points [7].

Like Seq2Seq models and Autoencoders, Transformers also consist of an encoder and a decoder. The encoder processes and encodes the entire input sequence into a detailed representation while the decoder generates the output sequence. Transformers are distinguished by their multi-layered architecture, where each encoder layer contains both a self-attention mechanism and a feed-forward neural network. The decoder layers also include an additional sublayer to attend to the encoder's output, enabling the model to handle complex dependencies more effectively than traditional sequence models [10].

A notable advantage of Transformers is their flexibility, as they can be configured in different setups: Encoder-Decoder, Encoder-Only, or Decoder-Only [11]. In this study, I explored both the Encoder-Only and Encoder-Decoder configurations.

## 2.2 Dataset Preparation

The models were trained and tested on a carefully selected dataset representing network traffic parameters over a 16-hour period for a specific link within the LHCONE (Large Hadron Collider Open Network Environment), shown in Figure 2. This dataset shows real network traffic produced by the File Transfer Service (FTS), a key service employed by LHC experiments to distribute data across various sites within the Worldwide LHC Computing Grid (WLCG). It was chosen due to its relevance to the challenges faced by the NOTED project.

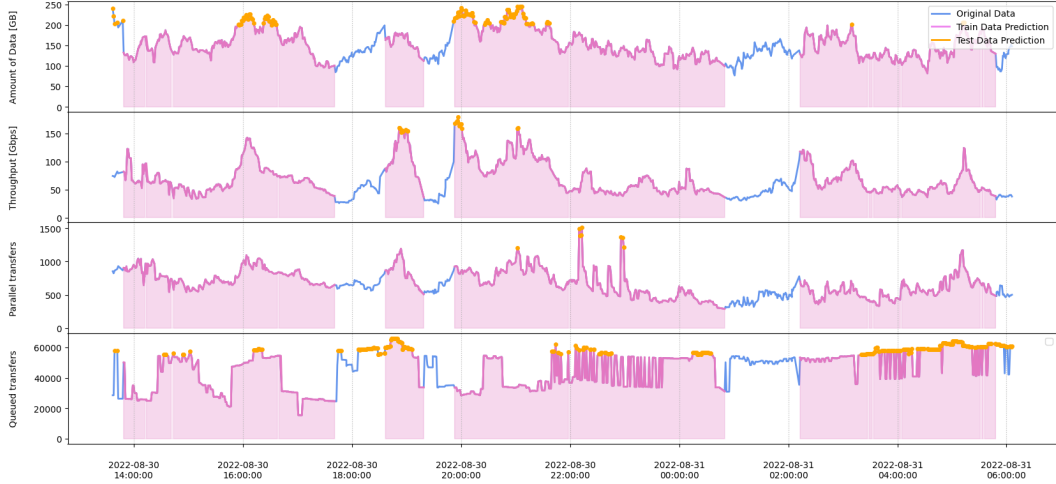


Figure 2: Test dataset consisting of four features describing the network traffic for one particular link in the LHCONE over a period of 16 hours on August 30<sup>th</sup> 2022. From top to bottom the individual graphs represent the following parameters: the amount of data in GB, the throughput in Gbps, number of parallel transfers and number of queued transfers.

The dataset comprised several critical features, each providing unique insights into the network’s performance:

- **Amount of Data Transferred (in GB):** This feature reflects the total volume of data moved across the network link during each time interval, offering a direct measure of network usage.
- **Throughput (in Gbps):** Throughput indicates the rate at which data is successfully transmitted over the network, serving as a key indicator of the link’s capacity and performance under varying load conditions.
- **Number of Parallel Transfers:** This feature captures the number of simultaneous data transfers occurring at any given time, providing insights into how network resources are allocated and utilized.
- **Queued Transfers:** The number of queued transfers reflects the backlog of data waiting to be transmitted, highlighting potential congestion points and bottlenecks in the network.

The dataset was divided into a training set (70%) and a testing set (30%) to enable the evaluation of the models’ predictive performance. It is important to note, that as throughput is the key parameter for NOTEDs functionality, this project centered on predicting future throughput.

## 2.3 Model Training and Testing

Each model was trained using the training set and then tested on the testing set to evaluate its ability to predict network traffic. The training process involved optimizing the models’ hyperparameters, namely epochs, batch size, and look-back period, to achieve the lowest possible Root Mean Squared Error (RMSE). It is to note that the used hyperparameters are not necessarily the optimal choice for the algorithms in general. They are just the combination of parameters that portrayed the best results during training in this specific scenario. The hyperparameters for each model are summarized in table 1.

The performance of the models was assessed based on RMSE, which measures the average magnitude of errors between predicted and actual values, and the Spearman Rank correlation coefficient ( $r_s$ ), which evaluates the strength and direction of the association between the predicted and actual network traffic values.

Table 1: Hyperparameters for different machine learning models

Algorithm	Look Back	Epochs	Batch Size
Seq2Seq Encoder-Decoder model	30	200	1
Seq2Seq Encoder-Decoder model with AM	30	200	1
Autoencoder	10	200	1
Transformer (Encoder-Only)	20	200	8
Transformer (Encoder-Decoder)	10	200	16

Table 2: Comparison of the different machine learning models performance on predicting future network throughput. RMSE is the mean value over 10 training iterations of the Root Mean Squared Error,  $\sigma$  is the corresponding standard deviation, and  $r_s$  is the Spearman Rank correlation coefficient. The CPU time indicates the required time for training each model (Apple M2 Max 12-Core CPU: 8 High-Performance Cores & 4 High-Efficiency Cores)

Algorithm	RMSE	$\sigma(\text{RMSE})$	$r_s$	CPU Time
Seq2Seq Encoder-Decoder model	5.783	0.141	0.9779	1min 54s
Seq2Seq Encoder-Decoder model with AM	5.740	0.106	0.9780	1min 59s
Autoencoder	11.609	0.034	0.9044	1min 57s
Transformer (Encoder-Only)	5.904	0.242	0.9782	7min 37s
Transformer (Encoder-Decoder)	11.775	0.321	0.9012	11min 19s

## 2.4 Evaluation of Computational Efficiency

While predictive accuracy is crucial, the computational efficiency of each model is equally important, particularly in real-time applications like NOTED. Here, rapid decision-making is essential, as there is a stringent time constraint of just a few minutes to predict congestion for the upcoming 20-30 minute window. To this end, I conducted an evaluation of the time required to train each model. This assessment involved measuring the CPU time consumed during the training phase using an Apple M2 Max Chip with a 12-core CPU.

Evaluating computational efficiency provided valuable insights into the trade-offs between accuracy and speed. Models that achieved high predictive accuracy but required excessive computation time might be impractical for real-time network management, where quick model updates and low latency are necessary. Conversely, models that are trained quickly but sacrifice accuracy might not provide the precision needed for effective traffic forecasting.

By balancing these considerations, I could identify the models that offered the best combination of accuracy and efficiency, making them suitable for deployment within the NOTED architecture. These findings not only contributed to the NOTED project’s immediate goals but also provided a framework for evaluating and selecting machine learning models for similar applications in other network management contexts.

## 3 Results & Discussion

Figure 3 shows the forecasting results for each model graphically. The original LHCONE throughput data is portrayed in blue, while the training and testing predictions made by the respective algorithms are illustrated in pink and orange. As mentioned above, the results focus on the throughput prediction values.

The results are detailed in Table 2, offering a thorough comparison of the performance of various machine learning models in predicting network throughput over a 4.8-hour period. The model’s performance was evaluated by computing the mean value of the RMSE over ten iterations and the corresponding standard deviation  $\sigma(\text{RMSE})$ . Additionally as mentioned above, the Spearman Rank correlation coefficient  $r_s$ , a non-parametric statistic that evaluates how well a monotonic relationship describes the connection between two variables without assuming any specific distribution, is computed to measure the strength of the association between the actual and estimated traffic [2]. The computational efficiency was evaluated by measuring the CPU time during training.

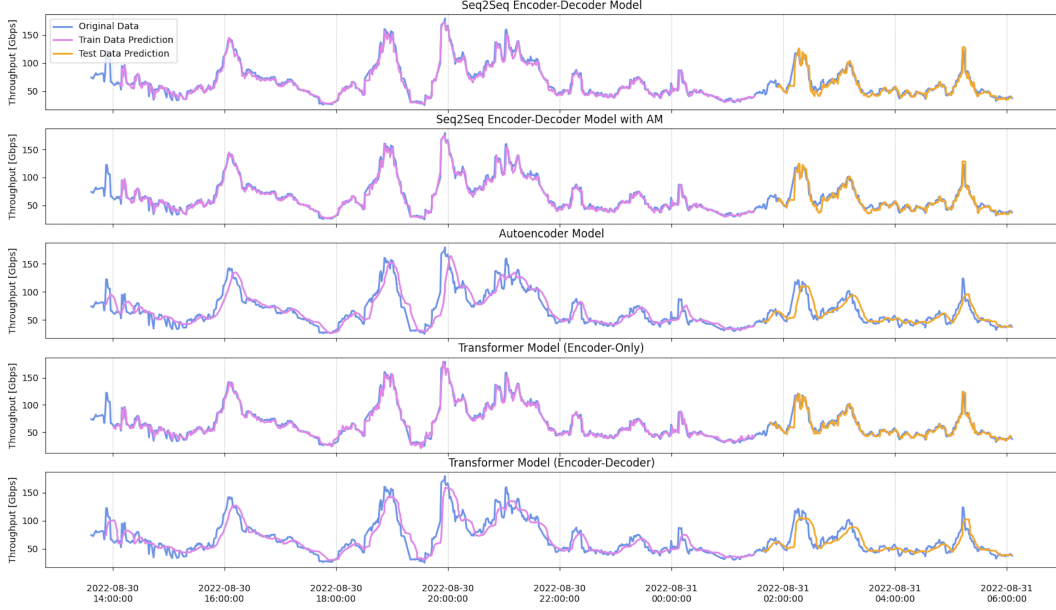


Figure 3: Comparison of different machine learning models trained and tested on the test dataset parameter throughput. The original data is portrayed in blue, while the predicted values are shown in pink, respectively orange for the training & testing phase of each model.

The results show that the Seq2Seq model with an attention mechanism performed the best, achieving the lowest RMSE and a high correlation with actual network traffic patterns. This model effectively captured the relevant temporal dependencies, making it a strong candidate for real-time network traffic forecasting within the NOTED architecture.

The Encoder-Only Transformer model, also performed well, with a high Spearman Rank correlation coefficient indicating its robustness in maintaining the rank order of predicted values. However, the slightly higher RMSE compared to Seq2Seq models suggests that further fine-tuning could improve their accuracy. The Encoder-Decoder model did not perform as well, with the highest RMSE value and the lowest  $r_s$  value.

While effective in other contexts, the Autoencoder underperformed in this application, with higher RMSE and lower correlation coefficients. This suggests that more complex temporal dependencies in network traffic data may require advanced mechanisms like attention, which Autoencoders lack.

Regarding computational efficiency, Seq2Seq models were the quickest to train and test, making them ideal for real-time applications. While justified by their accuracy, the significantly longer training times of Transformers may pose challenges in scenarios requiring rapid model updates and real-time traffic forecasting.

Overall, my work contributed to enhancing NOTED’s ability to forecast real-time network traffic more accurately and efficiently. By integrating these advanced machine learning techniques, NOTED can better manage data transfers across CERN’s global network, supporting the LHC’s critical data-sharing needs. Future work could explore hybrid models or apply these techniques across different network links to further generalize the results.



## References

- [1] Carmen Misa Moreira, Edoardo Martelli, and Tony Cass. NOTED: An intelligent network controller to improve the throughput of large data transfers in File Transfer Services by handling dynamic circuits. *EPJ Web of Conferences*, 295:07034, 2024. Publisher: EDP Sciences.
- [2] Joanna Waczyńska, Edoardo Martelli, Sofia Vallecorsa, Edward Karavakis, and Tony Cass. Convolutional LSTM models to estimate network traffic. *EPJ Web of Conferences*, 251:02050, 2021. Publisher: EDP Sciences.
- [3] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to Sequence Learning with Neural Networks, December 2014. arXiv:1409.3215 [cs].
- [4] Yao Qin, Dongjin Song, Haifeng Chen, Wei Cheng, Guofei Jiang, and Garrison Cottrell. A Dual-Stage Attention-Based Recurrent Neural Network for Time Series Prediction, April 2017.
- [5] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural Machine Translation by Jointly Learning to Align and Translate, May 2016. arXiv:1409.0473 [cs, stat].
- [6] Pengzhi Li, Yan Pei, and Jianqiang Li. A comprehensive survey on design and application of autoencoder in deep learning. *Applied Soft Computing*, 138:110176, May 2023.
- [7] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.
- [8] Borui Cai, Shuiqiao Yang, Longxiang Gao, and Yong Xiang. Hybrid variational autoencoder for time series forecasting. *Knowledge-Based Systems*, 281:111079, December 2023.
- [9] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention Is All You Need, August 2023. arXiv:1706.03762 [cs].
- [10] Sabeen Ahmed, Ian E. Nielsen, Aakash Tripathi, Shamooun Siddiqui, Ravi P. Ramachandran, and Ghulam Rasool. Transformers in Time-Series Analysis: A Tutorial. *Circuits, Systems, and Signal Processing*, 42(12):7433–7466, December 2023.
- [11] Tianyang Lin, Yuxin Wang, Xiangyang Liu, and Xipeng Qiu. A survey of transformers. *AI Open*, 3:111–132, January 2022.