Deep learning (INFOMDLR) - Assignment 1

Mahshid Jafar Tajrishi
Graduate School of Natural Sciences
Utrecht University
Netherlands
m.jafartajrishi@students.uu.nl

Bar Melinarskiy
Graduate School of Natural Sciences
Utrecht University
Netherlands
b.melinarskiy@students.uu.nl

Cis van Aken
Graduate School of Natural Sciences
Utrecht University
Netherlands
c.j.f.vanaken@students.uu.nl

Simon van Klompenburg
Graduate School of Natural Sciences
Utrecht University
Netherlands
a.s.vanklompenburg@students.uu.nl

***Abstract*—to be added**
***Index Terms*—RNN, GRU, LSTM, Time series forecasting**

1. Introduction

Predicting future values in temporal measurement datasets is essential across domains like signal processing, forecasting, and automated control systems. Our study focuses on developing a neural network model to predict subsequent data points in a laser-based temporal measurement dataset. The challenge involves identifying the optimal architecture, whether RNN, LSTM, or GRU, and fine-tuning hyperparameters such as window size, layer count, and dropout rates. Using a real-world laser dataset, we will evaluate these deep learning approaches through MSE and MAE performance metrics while assessing how accurately each model reproduces the signal characteristics in our test data.

2. Used Models

Time series forecasting involves capturing temporal dependencies and patterns across sequential data points, making Recurrent Neural Networks (RNNs) inherently suited for this task. RNNs maintain a hidden state that evolves over time, enabling the model to learn from previous time steps [1]. In this study, we explored three RNN architectures: Vanilla RNN, Long Short-Term Memory (LSTM), and Gated Recurrent Unit (GRU) to predict one-step-ahead laser measurements. These models were chosen due to their proven ability to capture temporal dependencies in sequential data [2, 3, 4]. Below, we briefly describe each model and the rationale for its selection.

A. Recurrent Neural Networks (RNNs)

RNNs are a class of neural networks designed to process sequential data by maintaining a hidden state that evolves over time. However, vanilla RNNs suffer from vanishing gradient problems, which limit their ability to capture long-term dependencies [5, 6]. Despite this limitation, we included RNNs as a baseline model to compare their performance against more advanced architectures.

B. Long Short-Term Memory (LSTM)

LSTMs address the vanishing gradient problem by introducing memory cells and gating mechanisms that regulate the flow of information [1]. This makes LSTMs particularly effective for capturing long-term dependencies in time-series data. Given the sequential nature of the laser measurement dataset, LSTMs were a natural choice for this task.

C. Gated Recurrent Unit (GRU)

GRUs are a streamlined alternative to LSTMs that use gating mechanisms to control the flow of information without relying on a separate memory cell. They merge the functionalities of the input and forget gates into a single update gate, and use a reset gate to modulate the influence of previous hidden states [4]. This simplification reduces the model's computational overhead while preserving its ability to capture temporal dependencies. GRUs were included to assess whether this balance of efficiency and performance could match the predictive accuracy of LSTMs, especially given our relatively small dataset that might not require an overly complex model.

All three models share a similar architecture: a recurrent layer (RNN, LSTM, or GRU) followed by a fully connected layer that outputs a single prediction for each input sequence. To ensure consistency across models, we used the following key parameters:

- ***input_size***: The number of features at each time step, which is set to 1 for this univariate time-series dataset.
- ***hidden_size***: The size of the hidden state, which determines the model's capacity to learn and represent patterns in the data.
- ***num_stacked_layers***: The number of recurrent layers stacked on top of each other. Increasing the depth allows the model to capture more complex temporal dependencies.

3. Approach

A. Training Methodology

The laser dataset, consisting of 1000 measurements, was divided into a training set with 750 samples and a validation set

with 250 samples. All models were implemented in PyTorch and trained using the Adam optimizer [7], which is well-suited for handling sparse gradients and adaptive learning rates. To prevent overfitting, we employed the ReduceLROnPlateau learning rate scheduler, which reduces the learning rate when the validation loss plateaus. Additionally, early stopping was used to terminate training if the validation loss did not improve for 10 consecutive epochs (patience = 10), with training capped at a maximum of 100 epochs.

We used Mean Absolute Error (MAE) as the loss function instead of Mean Squared Error (MSE). MAE is less sensitive to outliers compared to MSE, making it more robust for real-world datasets with potential noise or anomalies [8].

B. Hyperparameter Optimization

Hyperparameter tuning was performed using Optuna [9], a state-of-the-art framework for automated hyperparameter optimization. The following hyperparameters were tuned for each model:

- **Hidden Size**: Number of features in the hidden state, ranging from 128 to 256 in steps of 32.
- **Number of Layers**: Number of stacked RNN / LSTM / GRU layers, ranging from 1 to 3.
- **Learning Rate**: Initial learning rate for the Adam optimizer, sampled logarithmically between 10^{-5} and 10^{-2}.
- **Batch Size:** Number of samples per training batch, with options of 32, 64, and 128.
- **Window Size**: Number of past time steps used as input, ranging from 7 to 20 in steps of 1.

The best hyperparameters varied across the three models, reflecting their architectural differences and computational requirements. For the RNN, the optimal configuration included a hidden size of 224, two layers, a learning rate of 0.0012, a batch size of 128, and a window size of 15. This setup balanced the simplicity of the RNN architecture with sufficient capacity to capture temporal dependencies. In contrast, the LSTM achieved its best performance with a larger hidden size of 256, a single layer, a learning rate of 0.0025, a batch size of 64, and a smaller window size of 10. The larger hidden size and smaller window size highlight the LSTM's ability to effectively model long-term dependencies while requiring fewer input steps. Finally, the GRU performed best with a hidden size of 192, three layers, a learning rate of 0.0030, a batch size of 128, and a window size of 16. The GRU's smaller hidden size and deeper architecture leverage its efficient gating mechanisms to capture complex temporal patterns while maintaining computational efficiency. These differences underscore the trade-offs between model complexity, capacity, and the ability to generalize to the laser measurement dataset.

4. Results

The performance of RNN, LSTM and GRU models was systematically assessed for one-step-ahead prediction of laser measurements using MSE and MAE values across the training and validation sets, compared to LSTM and RNN.

As depicted in Figures 1 and 2, the GRU model exhibited faster convergence and lower final loss values, indicating efficient learning and generalization. The LSTM model showed moderate performance, while the RNN exhibited higher variance and slower convergence, reflecting its limited capacity to model long-term dependencies.

Figure I highlights the predictive accuracy of the models over 200 time steps. The GRU model's predictions closely aligned with the actual oscillatory patterns, while the LSTM displayed minor deviations. In contrast, the RNN predictions diverged significantly over extended horizons, indicating less robustness in recursive forecasting.

In summary, the GRU model proved to be the most efficient and accurate choice for this time-series forecasting task, balancing computational efficiency and predictive accuracy more effectively than the LSTM and RNN models.

TABLE I: Comparison of MSE and MAE across models (RNN, LSTM, GRU) on train, validation, and test sets. Rankings (in dark blue) indicate relative performance, with (1) being the best and (3) the worst.

Model	MSE (Train)	MSE (Validation)	MSE (Test)	MAE (Train)	MAE (Validation)	MAE (Test)
RNN	0.0037 (3)	$3.484 \cdot 10^{-5}$ (3)	X.X (3)	0.0170 (3)	0.0047 (3)	X.X (3)
LSTM	0.0017 (2)	$2.715 \cdot 10^{-5}$ (2)	X.X (2)	0.0108 (2)	0.0042 (2)	X.X (2)
GRU	0.0014 (1)	$1.755 \cdot 10^{-5}$ (1)	X.X (1)	0.0099 (1)	0.0033 (1)	X.X (1)

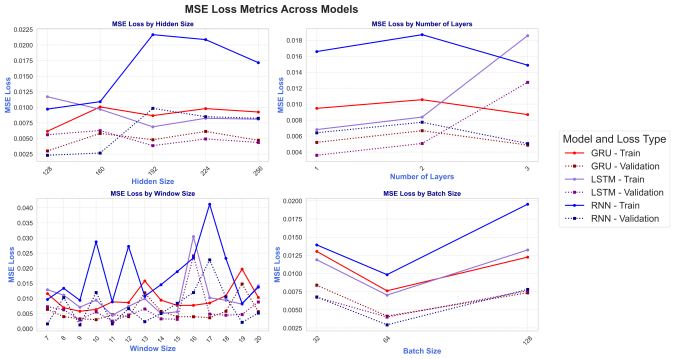

Fig. 1: MSE Loss Metrics Across Models. Comparison of Mean Squared Error (MSE) loss across GRU, LSTM, and RNN models for both training and validation data, illustrating how different hyperparameters (hidden size, number of layers, window size, and batch size) affect model performance.

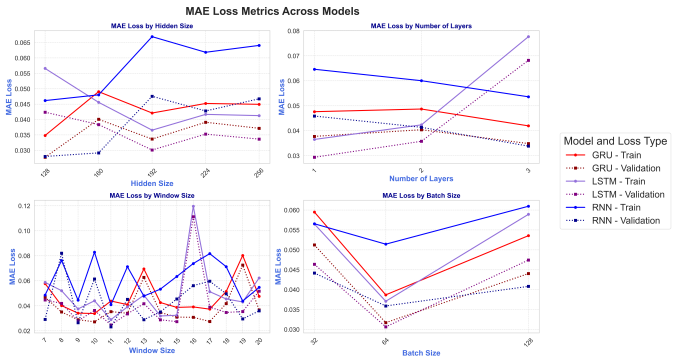

Fig. 2: MAE Loss Metrics Across Models. Comparison of Mean Absolute Error (MAE) loss across GRU, LSTM, and RNN models for both training and validation data, showing performance variations based on different hyperparameters: hidden size, number of layers, window size, and batch size.

To evaluate each model's ability to capture the laser data signal, we conducted a recursive forecasting experiment on the validation dataset. Starting with an initial window of real validation data, we predicted the next data point recursively and plotted the predicted signal against the actual data. This allowed us to assess how well the models captured the signal's frequency, magnitude, and amplitude. The results, shown in Fig. 3, highlight how prediction accuracy evolves over longer horizons, with varying deviations from the true oscillating pattern.

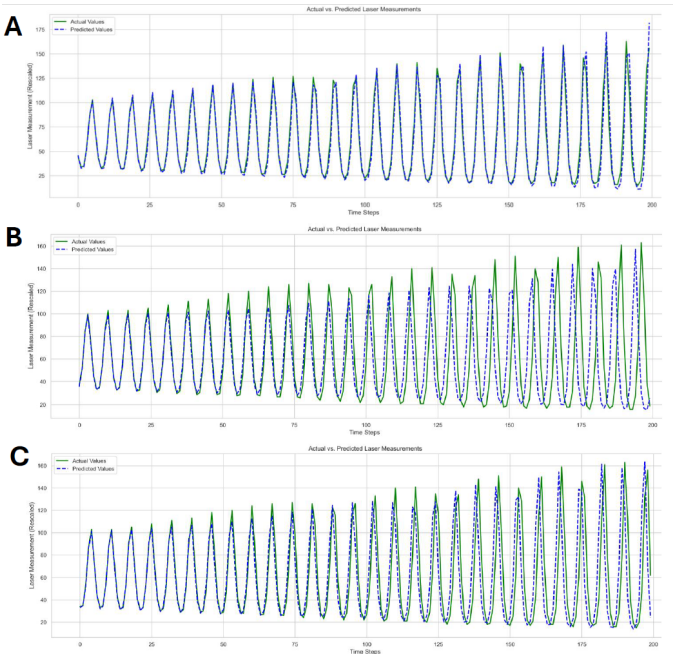

Fig. 3: Actual vs. Predicted Laser Measurements Across Three Model Types. Comparison of actual (green solid line) and predicted (blue dashed line) laser measurements over 200 time steps using recursive prediction for different recurrent neural network architectures: (A) RNN, (B) LSTM, and (C) GRU. Each model was trained on historical data and then used to predict 200 steps ahead by recursively feeding each prediction back into the model to generate subsequent forecasts.

All three models captured the signal's frequency reasonably well, though the LSTM slightly deviated as the forecast progressed. In terms of magnitude, the predicted wave widths mostly aligned with the actual signal. However, differences emerged in amplitude: the RNN, despite its lower validation MSE and MAE, aligned well with signal peaks but tended to overshoot over time, likely due to accumulated errors and difficulty with long-range dependencies. The LSTM, on the other hand, consistently undershot, producing lower peaks. This may be attributed to its higher complexity compared to the GRU, which could have led to underperformance given the relatively small dataset and the hyperparameter ranges optimized for simpler models. The GRU achieved the best overall fit, closely matching the actual signal in frequency, magnitude, and amplitude, demonstrating its ability to balance complexity and efficiency effectively.

5. Conclusion

By leveraging advanced recurrent architectures and rigorous hyperparameter tuning, we aimed to identify the most suitable model for one-step-ahead prediction of laser measurements. The use of MAE as the loss function, combined with techniques like early stopping and learning rate scheduling, ensured robust training and minimized overfitting.

REFERENCES

[1] Alex Sherstinsky. "Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network". In: *Physica D: Nonlinear Phenomena* 404 (2020), p. 132306.

[2] Sepp Hochreiter and Jürgen Schmidhuber. "Long short-term memory". In: *Neural computation* 9.8 (1997), pp. 1735–1780.

[3] Kyunghyun Cho et al. "Learning phrase representations using RNN encoder-decoder for statistical machine translation". In: *arXiv preprint arXiv:1406.1078* (2014).

[4] Rahul Dey and Fathi M Salem. "Gate-variants of gated recurrent unit (GRU) neural networks". In: *2017 IEEE 60th international midwest symposium on circuits and systems (MWSCAS)*. IEEE. 2017, pp. 1597–1600.

[5] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. "Learning long-term dependencies with gradient descent is difficult". In: *IEEE transactions on neural networks* 5.2 (1994), pp. 157–166.

[6] Liam Johnston et al. "Revisiting the problem of learning long-term dependencies in recurrent neural networks". In: *Neural Networks* 183 (2025), p. 106887. ISSN: 0893-6080. DOI: https://doi.org/10.1016/j.neunet.2024.106887. URL: https://www.sciencedirect.com/science/article/pii/S0893608024008165.

[7] Diederik P Kingma and Jimmy Ba. "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980* (2014).

[8] Cort J Willmott and Kenji Matsuura. "Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance". In: *Climate research* 30.1 (2005), pp. 79–82.

[9] Takuya Akiba et al. "Optuna: A Next-generation Hyperparameter Optimization Framework". In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM. 2019, pp. 2623–2631. DOI: 10.1145/3292500.3330701.