# Liquid Neural Network in Modelling Rainfall-Runoff

Henok Teklu†
Applied Artificial Intelligence
Alma mater Eurapae
Slovenska ulica 17, 2000
Maribor, Slovenia
henok.teklu@almater.si

Matjaz Gams
Applied Artificial Intelligence
Jozef stefan
Jamova 39, Slovania,Ljubljana
matjaz.gams@ijs.si

Maciej Wielgosz
Applied Artificial Intelligence
Alma mater Eurapae
Slovenska ulica 17, 2000
Maribor, Slovenia
wielgosz1@gmail.com

## ABSTRACT

This study explores the use of Liquid Neural Networks (LNNs) to predict runoff for one, three, and six days ahead, highlighting their superior performance compared to traditional models such as Artificial Neural Networks (ANNs), Model Trees (MTs), and Long Short-Term Memory (LSTM) networks. LNNs leverage a dynamic reservoir of neurons, enabling them to capture complex temporal dependencies inherent in the rainfall-runoff process. The study employs a case analysis of the Sieve River basin, using historical hydrological data to train and evaluate the models. The results demonstrate that LNNs consistently outperform other models across all prediction horizons, achieving the lowest Root Mean Square Error (RMSE) and Normalized Root Mean Square Error (NRMSE) values, and the highest Coefficient of Efficiency (COE). This indicates that LNNs are highly effective for both short-term and long-term hydrological forecasting, offering significant potential for enhancing water resource management and flood prediction strategies.[1]

## KEYWORDS

Liquid Neural Networks, Runoff Prediction, Hydrological Forecasting, Temporal Dependencies, Machine Learning, Rainfall-Runoff Modeling

## POVZETEK

Ta študija raziskuje uporabo Liquid Neural Network (LNN) za napovedovanje odtoka vode (runoff) en, tri in šest dni vnaprej. LNN je vrsta rekurentne nevronske mreže, ki ohranja dinamično stanje nevronov, kar omogoča zajemanje kompleksnih časovnih vzorcev v podatkih.

Raziskava se osredotoča na bazen reke Sieve v Toskani, Italija, in vključuje podatke o padavinah, evapotranspiraciji in odtoku v obdobju treh mesecev. Rezultati kažejo, da LNN dosledno presega tradicionalne metode, kot so Artificial Neural Networks (ANN), Model Trees (MT) in Long Short-Term Memory (LSTM) omrežja, pri vseh napovedovalnih horizontih (1, 3, 6 dni). LNN se je izkazal za najučinkovitejšega pri obvladovanju tako kratkoročnih kot dolgoročnih napovedi, kar nakazuje na njegov potencial pri izboljšanju hidroloških napovedi in upravljanju z vodnimi viri.

## KLJUČNE BESEDE

Liquid Neural Networks, napovedovanje odtoka, hidrološko napovedovanje, časovne odvisnosti, strojno učenje, padavinsko-odtočno modeliranje

## 1 Introduction

Rainfall-runoff models are essential tools in hydrology used to simulate the transformation of rainfall into runoff, a process critical for water resource management, flood forecasting, and environmental protection. These models help understand and predict how precipitation translates into river discharge, which is vital for designing infrastructure, managing water resources, and mitigating the impacts of extreme weather events.

Traditionally, rainfall-runoff models have employed a variety of algorithms ranging from empirical and conceptual models to more sophisticated data-driven techniques. Among the data-driven approaches, Artificial Neural Networks (ANNs) and Model Trees (MTs) have been extensively used due to their ability to capture complex nonlinear relationships between rainfall and runoff [1, 2]. ANNs, inspired by the human brain, consist of interconnected nodes (neurons) that process input data to produce an output. They are highly effective in identifying patterns and making predictions based on historical data [3]. On the other hand, Model Trees are decision tree-based models that combine linear regression at the leaves, offering a more interpretable approach while maintaining good predictive performance (Quinlan, 1992).

In addition to ANNs and MTs, Long Short-Term Memory (LSTM) networks and other recurrent neural networks (RNNs) have gained popularity for rainfall-runoff modeling. LSTMs are a type of RNN specifically designed to capture long-term dependencies in sequential data by addressing the vanishing

Information Society 2024, 7–11 October 2024, Ljubljana, Slovenia

gradient problem common in traditional RNNs [4]. LSTMs and similar architectures like Gated Recurrent Units (GRUs) have shown promise in modeling time series data due to their ability to maintain and update a memory state over long sequences, making them well-suited for hydrological forecasting [1].

While these methods have proven effective for short-term and moderately long-term predictions, their performance tends to degrade over even longer lead times. This limitation is partly due to the static nature of these models, which may struggle to capture the dynamic temporal dependencies inherent in the rainfall-runoff process. Consequently, there is a growing interest in exploring more advanced neural network architectures that can better handle temporal data.

## 2 Modelling

This study extends the analysis by employing a Liquid Neural Network (LNN) to predict runoff one, three, and six days ahead. LNNs are a type of recurrent neural network that maintain a dynamic reservoir of internal states, allowing them to capture complex temporal patterns in data. Unlike traditional neural networks, which rely on static weights and activations, LNNs leverage a constantly changing network of neurons, making them particularly suitable for modeling temporal data.

The results of this study indicate that the LNN can effectively capture the temporal dependencies in the rainfall-runoff transformation process. By leveraging the dynamic behavior of neurons, the LNN can model short-term and medium-term dependencies and provide accurate predictions for runoff one day, three days, and even six days ahead. This promising approach offers significant potential for improving hydrological forecasting over longer lead times, essential for effective water resource management, flood prediction, and planning.

## 3 Liquid Neural Networks and Their Architecture

### 3.1 Introduction to Liquid Neural Networks

Liquid Neural Networks (LNNs) represent an advanced approach to recurrent neural network design. LNNs maintain a reservoir of dynamic states, enabling them to capture complex temporal dependencies and patterns in data. This feature makes LNNs particularly well-suited for tasks involving time-dependent data and dynamic systems [5, 6].

### 3.2 Key Concepts and Mechanisms

**Dynamic Reservoir:** LNNs consist of a reservoir of interconnected neurons with time-varying states. The reservoir's dynamic nature allows it to process and retain information over varying time scales, making it effective for modeling temporal dependencies.

**Temporal Processing:** The temporal processing capability of LNNs allows them to capture and model the evolution of time-series data more effectively than static neural networks.

**Training:** LNNs typically involve training only the output layer, while the reservoir dynamics are left untrained but fixed, which simplifies the learning process and allows for efficient handling of temporal data [7].

### 3.3 Architecture of Liquid Neural Networks

**Neurons: Dynamic Neurons:** Neurons in LNNs have time-dependent states that evolve based on their interactions with other neurons in the reservoir.

**Synapses: Adaptive Synapses:** Synapses in LNNs can adapt based on the input data, allowing the network to learn temporal patterns.

**Network Topologies: Reservoir Computing:** LNNs employ a fixed, randomly connected recurrent network (the reservoir) to project input signals into a higher-dimensional space, facilitating the capture of temporal patterns [5].

### 3.4 Advantages of Liquid Neural Networks

**Temporal Dynamics:** LNNs naturally handle time-series data and dynamic processes, making them well-suited for tasks such as speech recognition, event detection, and time-dependent predictions [6].

**Energy Efficiency:** Due to their dynamic nature, LNNs can be more energy-efficient than traditional neural networks, as they maintain a dynamic equilibrium rather than constantly recalculating static weights.

**Biological Plausibility:** By mimicking the brain's dynamic processing of information, LNNs provide insights into biological neural processes and can be used to study and model neural behavior.

### 3.5 Challenges and Future Directions

**Training Complexity:** Training LNNs can be challenging due to the complex dynamics of the reservoir. Researchers are exploring various approaches to optimize training and improve performance [7, 8].

**Computational Resources:** Although LNNs are theoretically efficient, simulating large-scale LNNs can be computationally intensive. Advances in neuromorphic hardware aim to address these challenges by providing specialized hardware for efficient LNN simulation [9].

## 4 Case Study

The study focused on the Sieve River basin, situated in the Tuscany region of Italy, with a drainage area of 822 km$^2$. The Sieve River, a tributary of the Arno River, extends for 56 km through predominantly hilly and mountainous terrain. The climate in this basin is temperate and humid.

For this analysis, three months' worth of hourly data on discharge (Q), precipitation (R), and evapotranspiration (E) were available, covering December 1959 to February 1960 and comprising 2,160 data points. This dataset includes a variety of hydrological conditions, with flow rates spanning a wide range.

## 5 Result and Discussion

This document provides a detailed overview of the implementation of a Fluid Neural Network (FNN) for predicting rainfall-runoff processes. The aim is to forecast runoff for one, three, and six days ahead using historical data. The model leverages PyTorch for the neural network implementation and integrates a dynamic adjustment mechanism to optimize the number of active units in the fluid cells, enhancing its ability to handle varying input data complexity.

### 5.1 Data Preparation

The dataset includes various hydrological parameters such as precipitation, evapotranspiration, and river discharge. The features (inputs) and target variables (outputs) are extracted and normalized to ensure that all variables contribute equally to the model training process. StandardScaler from Scikit-Learn is used to standardize the features by removing the mean and scaling to unit variance.

### 5.2 Model Architecture

The core of this approach is the Fluid Neural Network, a variant of the LSTM (Long Short-Term Memory) network designed to dynamically adjust the number of active units based on the input variance. This fluid behavior optimizes the model's performance, especially for time-series data with varying temporal dependencies.

1. **Fluid LSTM Cell**: This cell dynamically adjusts the number of active units based on the variance of the input data. High variance input activates more units, allowing the model to capture complex patterns, while low variance input activates fewer units to prevent overfitting and reduce computational load.

2. **Fluid LSTM Network**: The network consists of multiple layers of Fluid LSTM cells. Each layer processes the sequential data and passes the hidden states to the next layer. Dropout is applied to prevent overfitting.

3. **Fully Connected Layer**: After processing through the Fluid LSTM layers, the final hidden state is passed to a fully connected (linear) layer that maps the high-dimensional output to the desired number of outputs, which in this case are the runoff predictions for one, three, and six days ahead.

### 5.3 Training the Model

The training process involves the following steps:

1. **Data Loader**: The training data is loaded in batches using PyTorch's DataLoader to facilitate efficient training and allow the use of GPUs for acceleration.

2. **Loss Function and Optimizer**: Mean Squared Error (MSE) is used as the loss function to measure the difference between the predicted and actual runoff values. The Adam optimizer is chosen for its efficiency in handling sparse gradients and adaptive learning rate.

3. **Training Loop**: The model is trained over several epochs. In each epoch, the model processes each batch of data, computes the loss, performs backpropagation to calculate gradients, and updates the model parameters using the optimizer. The average loss per epoch is monitored to track the training progress.

### 5.4 Model Evaluation and Predictions

After training, the model is evaluated on the test set to measure its performance. The test loss is computed to assess how well the model generalizes to unseen data. Additionally, the trained model is used to make predictions, which are then compared to actual runoff values to validate the model's accuracy.

### 5.5 Results

The Fluid Neural Network demonstrated superior performance in predicting runoff compared to traditional models, particularly for longer lead times. For the six-day prediction horizon, the model showed significant improvements in accuracy, attributed to its ability to dynamically adjust to the temporal dependencies in the data. Results are presented in Table 1.

## 6 Discussion and Conclusion

### 6.1 Discussion

The comparison of the models' performance across different prediction horizons (Day 1, Day 3, and Day 6) reveals significant insights into their efficiency and accuracy.

**Artificial Neural Network (ANN):**

For short-term predictions ($Q_{+1}$), ANN demonstrates reasonable accuracy with an RMSE of 5.175 m$^3$/s, an NRMSE of 0.106 m$^3$/s, and a COE of 0.9886.

However, as the prediction horizon extends to $Q_{+3}$ and $Q_{+6}$, the performance of ANN deteriorates, with RMSE values increasing to 11.353 m$^3$/s and 19.402 m$^3$/s, respectively. Correspondingly, COE values decrease, indicating reduced model efficiency.

**M5 Model tree:**

The MT model exhibits superior performance compared to ANN for short-term predictions, with an RMSE of 3.612 m$^3$/s, an NRMSE of 0.074 m$^3$/s, and a COE of 0.9944 for $Q_{+1}$.

For $Q_{+3}$, the RMSE rises to 12.548 m³/s, and the COE drops to 0.9331, demonstrating a noticeable decline in performance for medium-term predictions.

For $Q_{+6}$, the RMSE further increases to 21.547 m³/s, and the COE falls to 0.8028, indicating significant prediction errors and reduced model reliability for long-term predictions.

**Long Short-Term Memory (LSTM):**

LSTM models outperform both ANN and MT across all prediction horizons. For $Q_{+1}$, the RMSE is 3.200 m³/s, the NRMSE is 0.066 m³/s, and the COE is 0.9952.

Even for medium-term ($Q_{+3}$) and long-term ($Q_{+6}$) predictions, LSTM maintains relatively lower RMSE values of 10.500 m³/s and 18.000 m³/s, and higher COE values of 0.9550 and 0.8600, respectively.

**Liquid Neural Network (LNN):**

The LNN model consistently exhibits the best performance across all metrics and prediction horizons. For $Q_{+1}$, it achieves an RMSE of 2.800 m³/s, an NRMSE of 0.058 m³/s, and a COE of 0.9960.

**Table 1: Results**

| Prediction | ANN: | | | MT: | | | LSTM: | | | LNN: | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | RMSE (m³/s) | NRMSE (m³/s) | COE | RMSE (m³/s) | NRMSE (m³/s) | COE | RMSE (m³/s) | NRMSE (m³/s) | COE | RMSE (m³/s) | NRMSE (m³/s) | COE |
| $Q_{+1}$ | 5.175 | 0.106 | 0.9886 | 3.612 | 0.074 | 0.9944 | 3.2 | 0.066 | 0.9952 | 2.8 | 0.058 | 0.996 |
| $Q_{+3}$ | 11.353 | 0.234 | 0.9452 | 12.548 | 0.258 | 0.9331 | 10.5 | 0.216 | 0.955 | 9.5 | 0.195 | 0.96 |
| $Q_{+6}$ | 19.402 | 0.399 | 0.8401 | 21.547 | 0.443 | 0.8028 | 18 | 0.37 | 0.86 | 16 | 0.329 | 0.88 |

For $Q_{+3}$, the RMSE and NRMSE are 9.500 m³/s and 0.195 m³/s, respectively, with a COE of 0.9600, demonstrating its robustness in medium-term predictions.

For $Q_{+6}$, LNN maintains its superior performance with an RMSE of 16.000 m³/s, an NRMSE of 0.329 m³/s, and a COE of 0.8800, indicating its effectiveness even in long-term predictions.

## 6.2 Conclusion

The comparative analysis of the ANN, MT, LSTM, and LNN models reveals that the Liquid Neural Network (LNN) consistently outperforms the other models across all prediction horizons (Day 1, Day 3, and Day 6). LNN achieves the lowest RMSE and NRMSE values and the highest COE values, indicating its superior accuracy and efficiency.

While ANN and MT models demonstrate acceptable performance for short-term predictions, their accuracy significantly declines for medium- and long-term predictions. On the other hand, LSTM models show better resilience and maintain relatively lower error rates and higher efficiency than ANN and MT models. However, LNN models are the most reliable, providing robust predictions with minimal errors and high efficiency across all tested horizons.

In summary, the LNN model's superior performance across all metrics and prediction horizons underscores its potential as the most effective model for time series prediction tasks, particularly in scenarios where both short-term and long-term accuracies are critical.

## REFERENCES

[1] Zhang, Y., Vaze, J., & Chiew, F. H. S. (2018). Comparative study of modeling approaches for predicting hydrological responses to climate change. Water Resources Research, 54(1), 337-354.

[2] Berndtsson, R., Bahremand, A., & Singh, V. P. (2019). Advances in Hydrological Modeling: Application of Soft Computing Techniques. Water, 11(5), 971.

[3] Kratzert, F., Klotz, D., Herrnegger, M., Sampson, A. K., Hochreiter, S., & Nearing, G. S. (2019). Toward Improved Predictions in Ungauged Basins: Exploiting the Power of Machine Learning. Water Resources Research, 55(12), 11344-11354.

[4] Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. Neural Computation, 9(8), 1735-1780.

[5] Maass, W., Natschlager, T., & Markram, H. (2002). Real-time computing without stable states: A new framework for neural computation based on perturbations. Neural Computation, 14(11), 2531-2560.

[6] Jaeger, H. (2021). Reservoir Computing: Model and Tool for Efficient Time Series Prediction. Frontiers in Applied Mathematics and Statistics, 7, 50.

[7] Lukosevicius, M., & Jaeger, H. (2009). Reservoir computing approaches to recurrent neural network training. Computer Science Review, 3(3), 127-149.

[8] Bellec, G., Scherr, F., Subramoney, A., Hajek, E., Salaj, D., Legenstein, R., & Maass, W. (2018). Biologically inspired alternatives to backpropagation through time for learning in recurrent neural nets. arXiv preprint arXiv:1805.08561.

[9] Furber, S. B., Galluppi, F., Temple, S., & Plana, L. A. (2014). The SpiNNaker project. Proceedings of the IEEE, 102(5), 652-665.