KI Service Zentrum by Hasso-Plattner-Institut

Gefördert durch:
Bundesministerium für Forschung, Technologie und Raumfahrt

Time Series Forecasting

3.2 Recurrent Neural Networks

Mario Tormo Romero

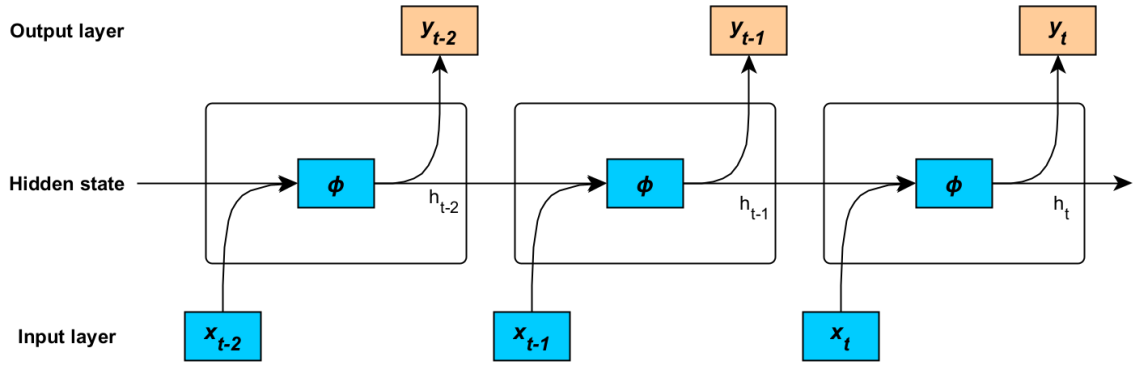
Design IT.
Create Knowledge.

What we'll cover in this video

- Understand the role of RNNs in modeling sequential and time-dependent data.
- Explore common RNN architectures:
  - RNN → Fully Connected
  - RNN → RNN (stacked)
- Learn about advanced RNN variants:
  - Long Short-Term Memory (LSTM)
  - Gated Recurrent Unit (GRU)
- Compare RNN, LSTM, and GRU for forecasting tasks.
- Get practical guidelines for selecting the right architecture for your forecasting problem.

Recurrent Neural Networks (RNNs)

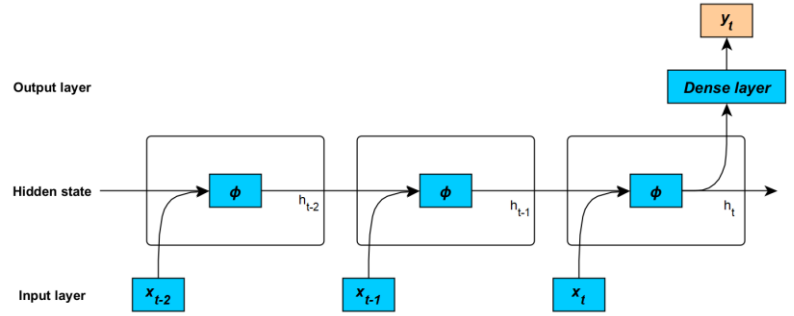
- Designed to handle **sequential data** (e.g., time series, speech, text).
- Capture temporal dependencies — today depends on yesterday.
- Struggle with long-term dependencies due to the **vanishing gradient problem**.

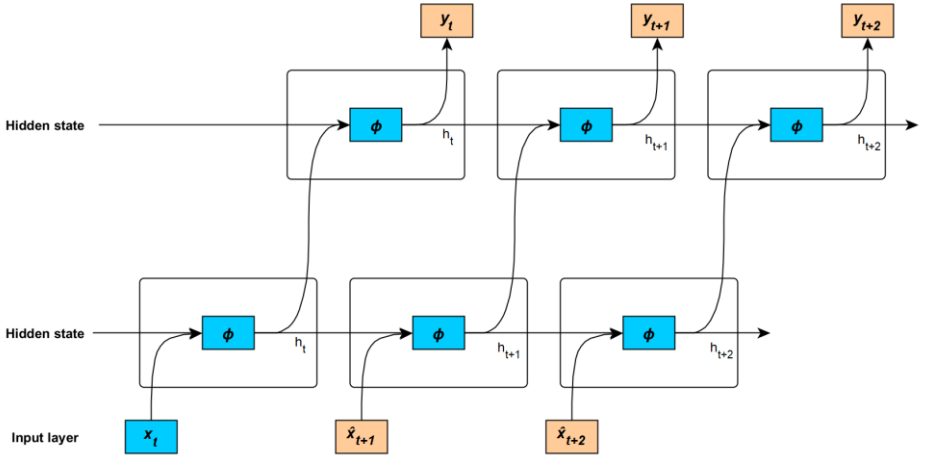


- How they work:
  - Process inputs step-by-step.
  - Pass information forward via a **hidden state**.

Connecting RNN layers

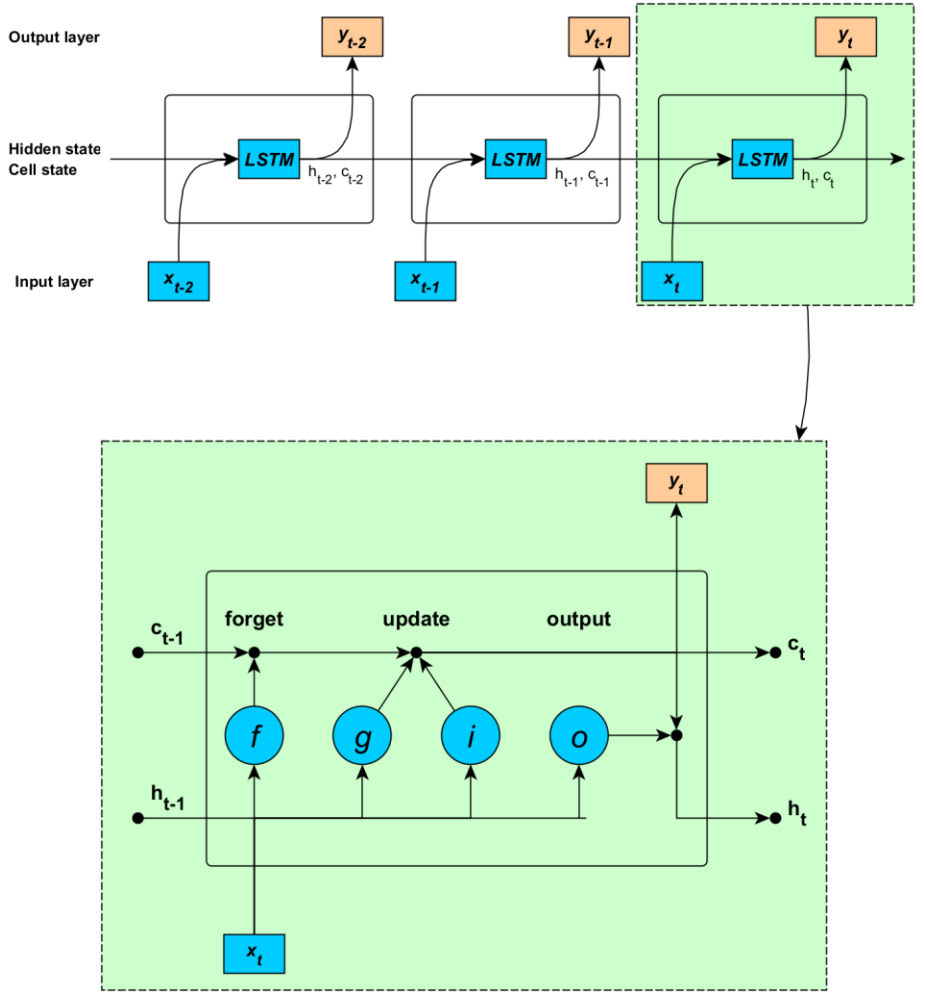
- **RNN → Fully Connected (Dense)**
  - RNN processes sequence → final hidden state → fully connected layer → prediction.
  - Useful for single-step forecasting (predicting one future value).
- **RNN → RNN (Stacked RNNs)**
  - Output sequence from one RNN layer → input to another RNN layer.
  - Allows the network to learn more complex patterns and can improve accuracy in multi-step forecasting or data with multiple seasonalities.





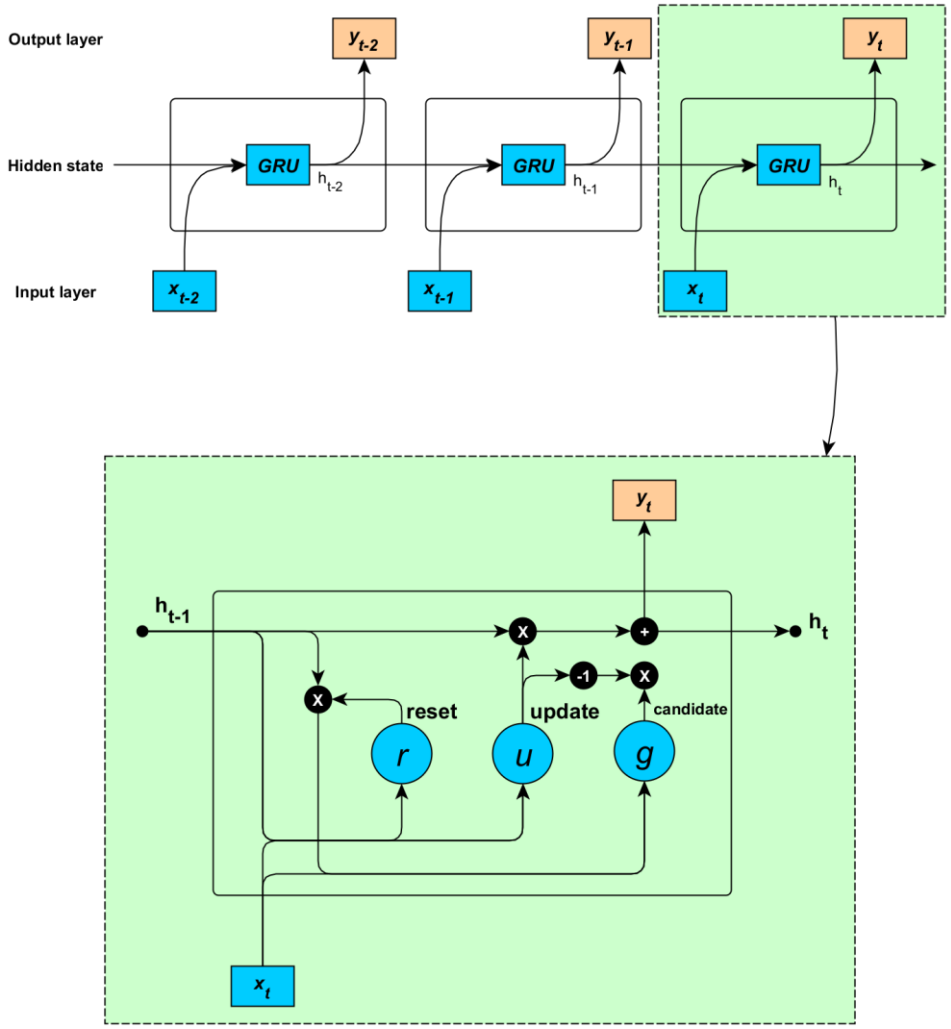
Long Short-Term Memory (LSTM)

- LSTM uses a **cell state** to preserve information over longer sequences.
- How it works:
  - Three gates control information flow:
    - **Forget gate** – what to discard.
    - **Input gate** – what new info to store.
    - **Output gate** – what to reveal in the hidden state.
- LSTMs can capture long-range dependencies, such as seasonal cycles or delayed effects.
- More parameters → higher computational cost.



Simplifying Memory with GRUs

- A **Gated Recurrent Unit (GRU)** is a streamlined version of the LSTM designed to capture long-term dependencies with fewer components.
- Instead of three gates, GRUs use only two gates:
  - **Reset gate**: controls how much past information to forget.
  - **Update gate**: decides how much new information to add and what to keep from the past.
- More computationally efficient and faster to train than LSTMs, and often delivers comparable performance on many time series tasks.
- Great for applications with limited resources or where speed matters.



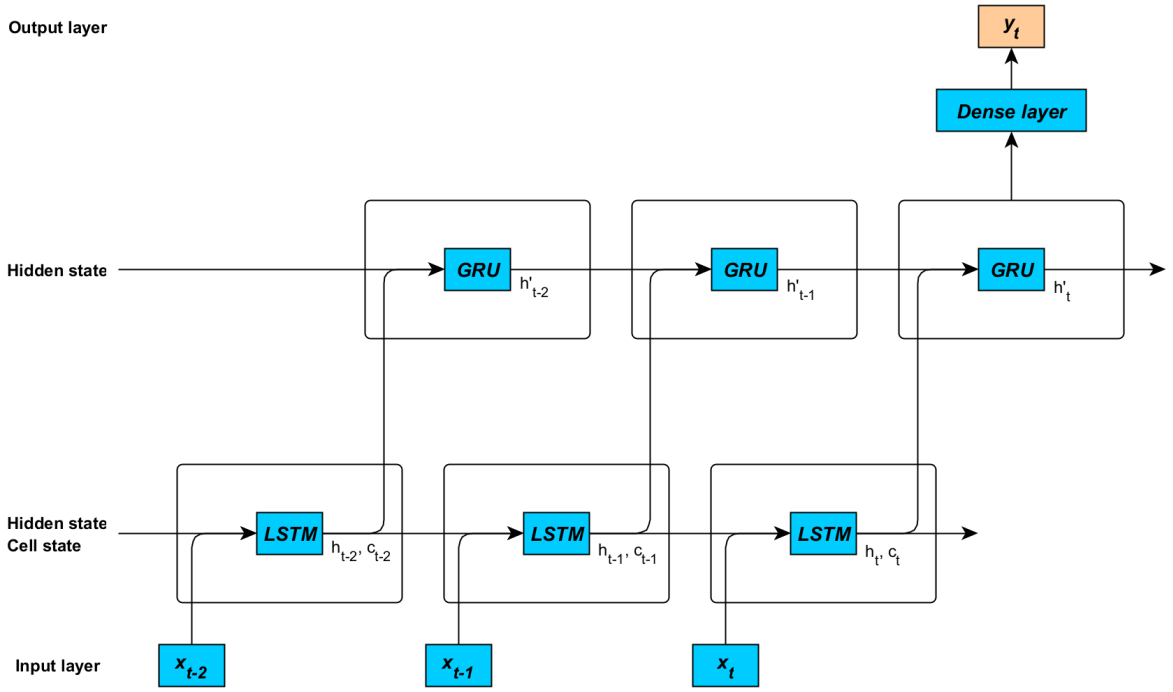
LSTM vs GRU in Practice

- Both LSTM and GRU consistently outperform simple baselines (vanilla RNNs, ARIMA) in many forecasting tasks with nonlinear patterns or multiple features.
- GRU: Faster training, fewer parameters, often matches LSTM performance on short/medium sequences or smaller datasets.
- LSTM: Better at capturing very long-term dependencies, irregular seasonal patterns, or delayed effects.
- No universal winner — choice depends on sequence complexity, dataset size, and compute budget.

Feature	LSTM	GRU
Gates	3 (Forget, Input, Output)	2 (Reset, Update)
States	Cell state (long-term) + Hidden state (short-term)	Single hidden state
Parameters	More, so heavier & slower to train	Fewer, so lighter & faster to train
Strengths	Captures very long dependencies; complex patterns	Trains faster; works well on smaller datasets
When it Wins	Long sequences, irregular seasonality, delayed effects	Limited compute, faster experiments, moderate sequences
Typical Performance	Slight edge on very complex tasks	Often matches LSTM in real-world datasets

Hybrid Architectures

- Example design:
  - Layer 1: LSTM → extracts long-term temporal patterns from the raw sequence.
  - Layer 2: GRU → refines and adapts to recent changes and shorter-term dependencies.
  - Layer 3: Dense (fully connected) → maps the extracted features to the forecasting target.
- When it helps:
  - Datasets with both slow-changing seasonal patterns and rapid fluctuations.
  - Medium-to-large datasets where training cost is acceptable.
  - Situations where neither an LSTM-only nor GRU-only approach captures all the dynamics.
- Trade-offs:
  - More parameters → longer training and higher risk of overfitting on small datasets.
  - Performance gains are dataset-dependent, not guaranteed.



What we've learnt

- **RNN**s are well suited for time series data but struggle with long-term dependencies due to vanishing/exploding gradients.

- **LSTM**s overcome this by using gated memory cells (input, forget, output gates) to capture long-range patterns, often outperforming traditional models like ARIMA.

- **GRU**s offer a streamlined alternative with fewer gates (reset, update), providing faster training and competitive accuracy, especially on shorter or moderate-length sequences.

- Empirically, both LSTM and GRU outperform simpler baselines, with LSTM sometimes excelling on very complex or long-term dependency tasks.

- Hybrid architectures that stack LSTM and GRU layers combine their strengths, improving forecasting accuracy on datasets with mixed long- and short-term dynamics.