Lecture 7:
RNNs and Time Series Analysis

Yanbin Liu

Auckland University of Technology

April 18, 2024

Table of Contents

Time Series Regression

- Time series regression is a statistical method for predicting a **future response** based on the **response history** (known as *autoregressive* dynamics) and the transfer of dynamics from relevant predictors.
- Time series regression can help us understand and predict the behavior of dynamic systems from experimental or observational data.
- Time series regression is employed for modeling and forecasting *economic*, *traffic*, and *weather* systems.

https://au.mathworks.com/discovery/time-series-regression.html

Time Series Analysis

Linear Regression Model

$$\mathbf{y}_t = \mathbf{x}_t\beta + e_t$$

$\mathbf{x}_t$: includes *current* and *past* observations by time $\mathbf{t}$
$\mathbf{y}_t$: an estimate of a *linear* relationship of the response
β: linear parameter estimates
e_t: innovation terms, difference between *observed* and *predicted*

Non-Linear Regression Model

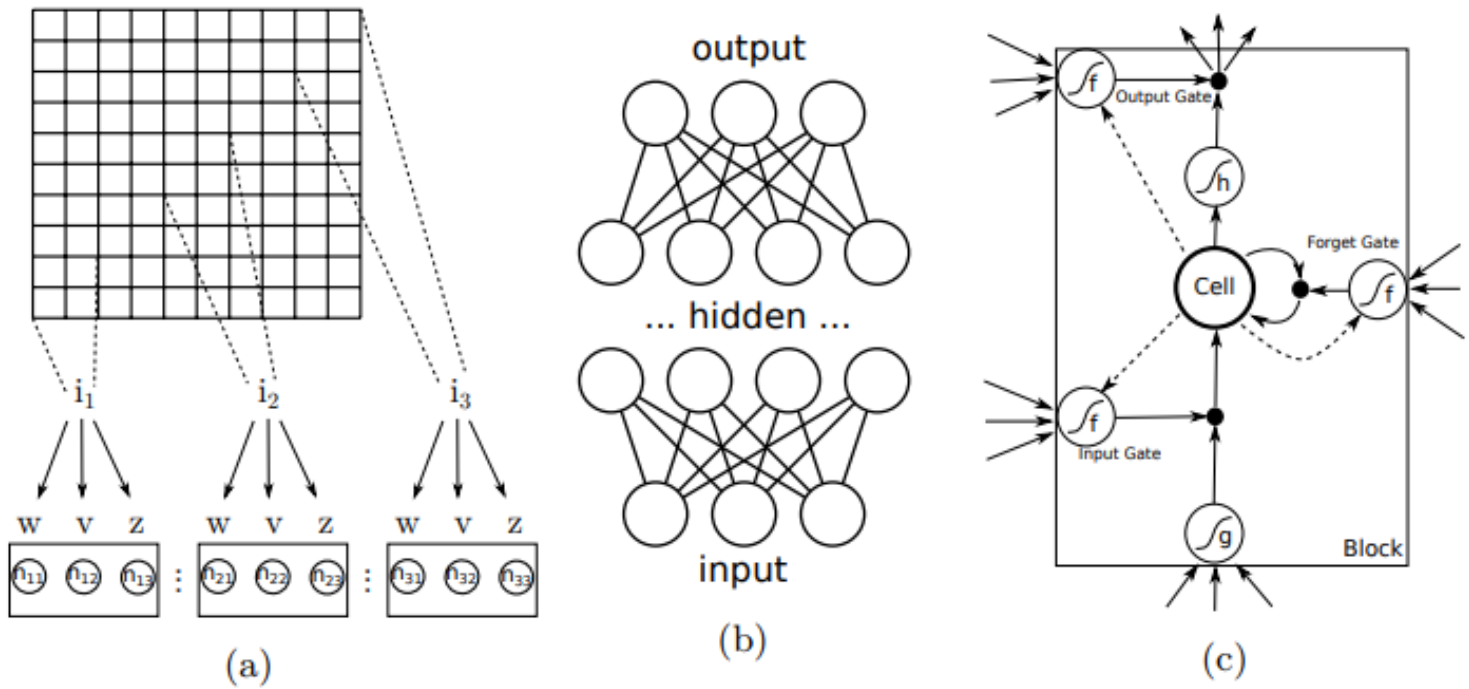
$$\mathbf{y}_t = f(\mathbf{x}_t, e_t)$$

f can be:

- CNN
- RNN such as LSTM and GRU
- Transformer

https://au.mathworks.com/discovery/time-series-regression.html

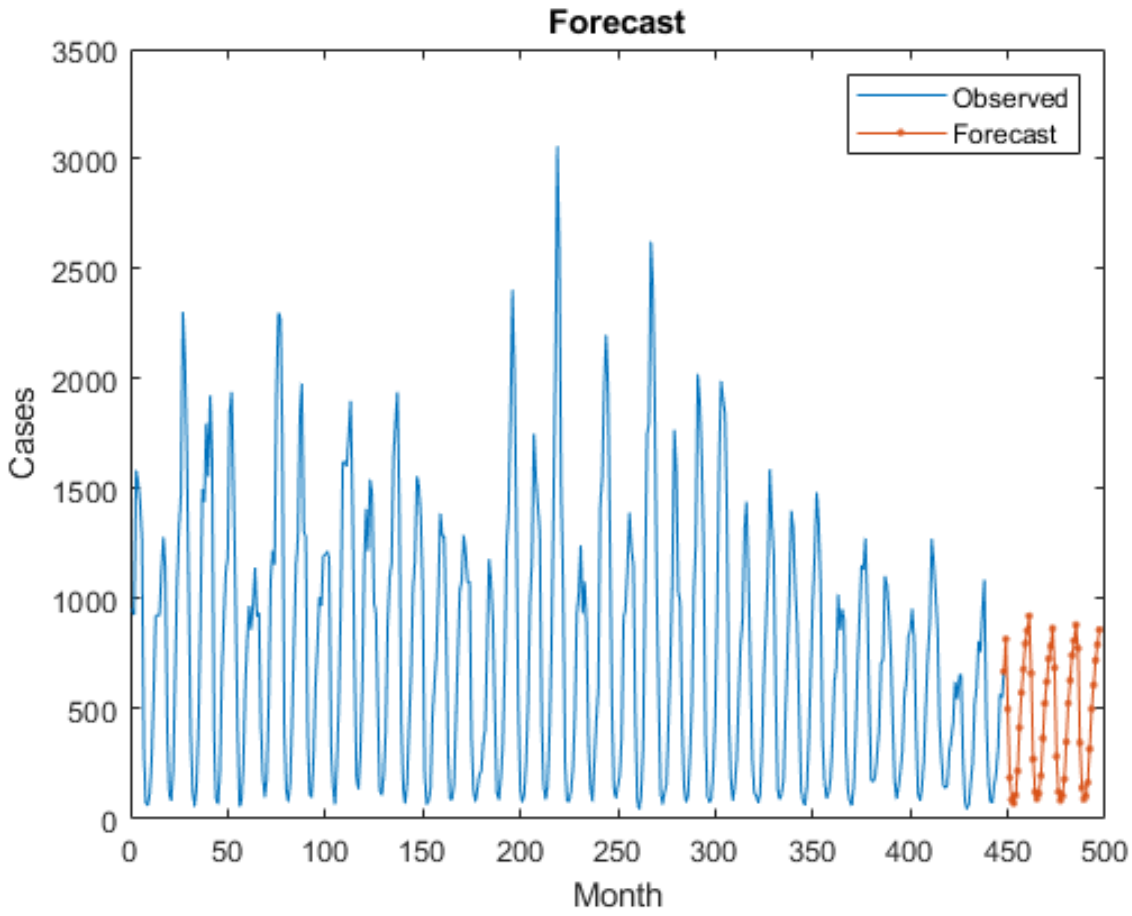
Deep Learning for Time Series Analysis



(a) The convolutional layer of a CNN with three groups (filters). (b) An artificial neural network (c) An LSTM block.

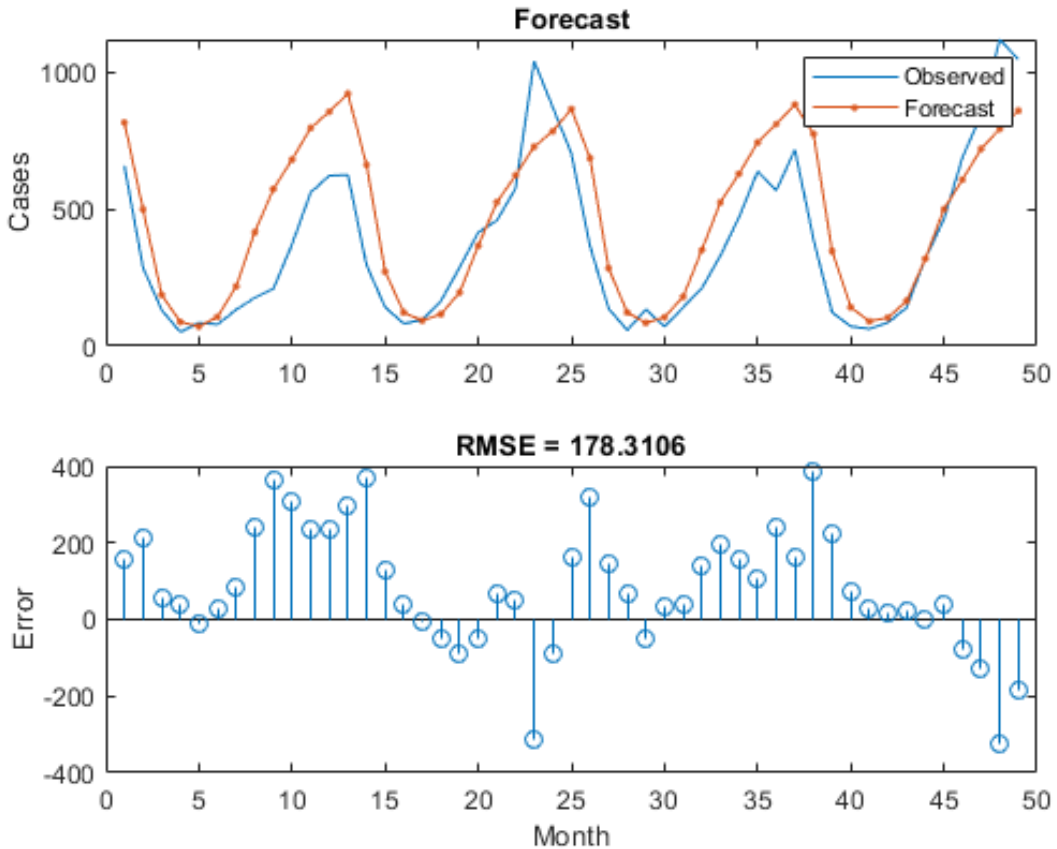
J. Gamboa. (2017) Deep Learning for Time-Series Analysis.

MATLAB LSTM for Time Series Forecast



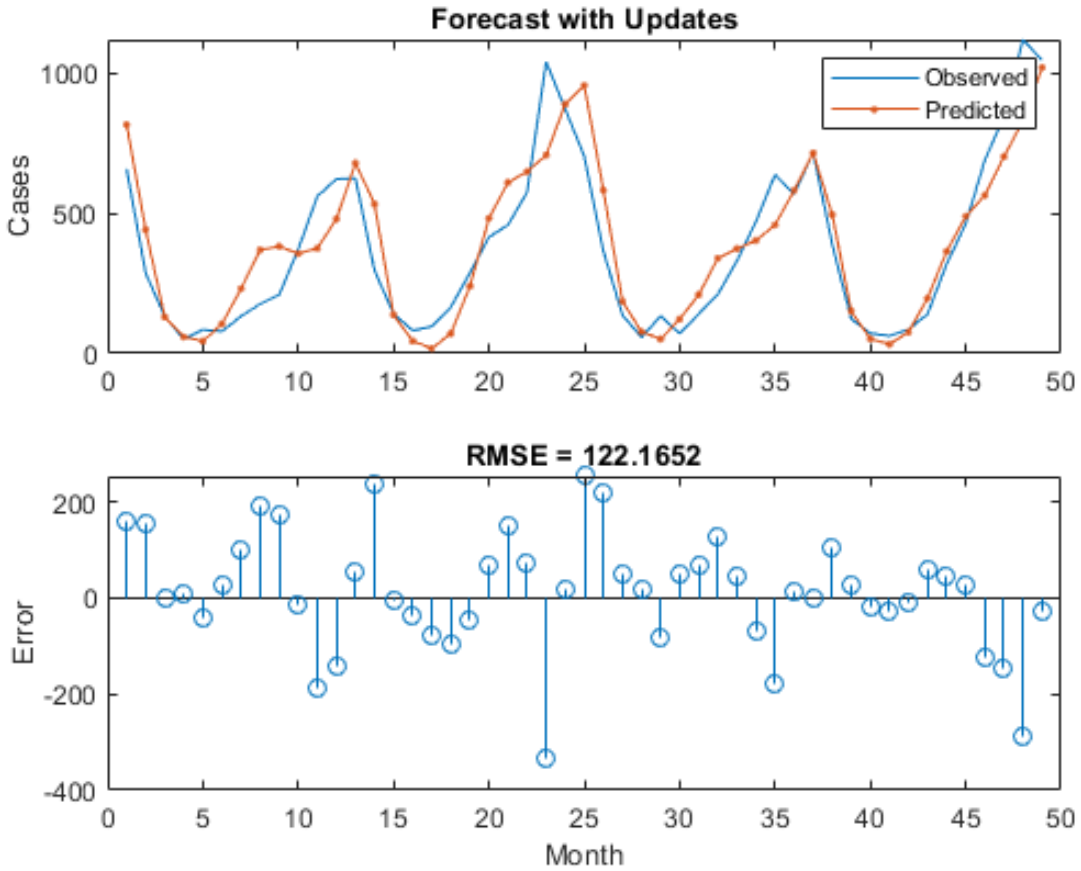
https://au.mathworks.com/help/nnet/examples/time-series-forecasting-using-deep-learning.html

MATLAB LSTM for Time Series Forecast



https://au.mathworks.com/help/nnet/examples/time-series-forecasting-using-deep-learning.html

MATLAB LSTM for Time Series Forecast with Updates

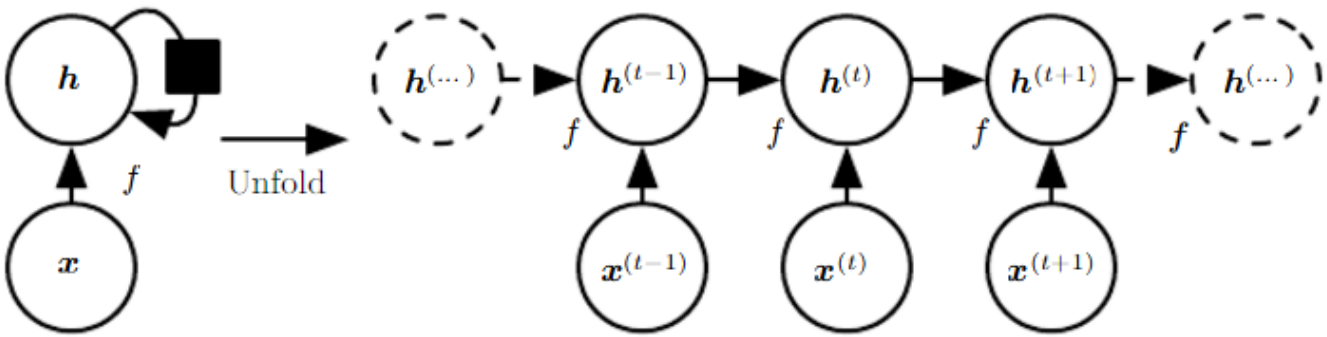


https://au.mathworks.com/help/nnet/examples/time-series-forecasting-using-deep-learning.html

Questions?

Recurrent Neural Networks

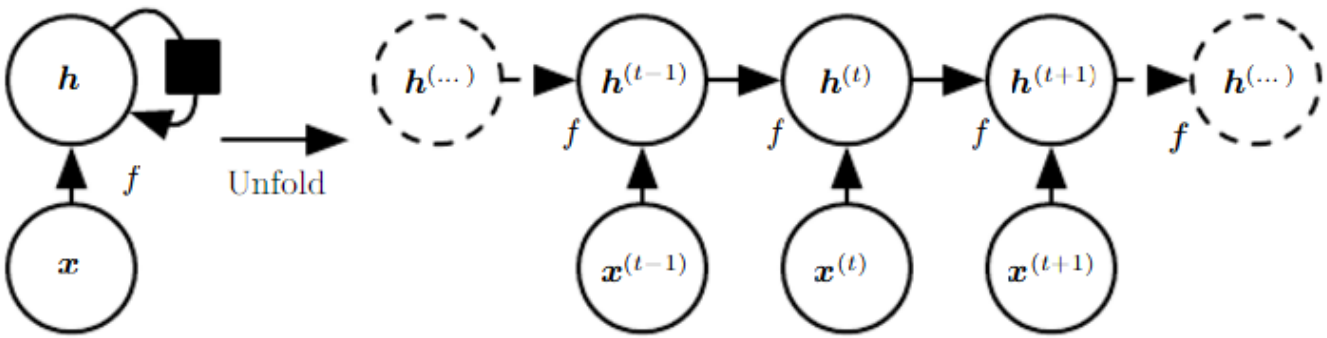
RNNs are a family of neural networks for processing *sequential data*, which is a dynamical system. It uses the **same** *transition* function with the **same** *parameters* at every time step.



I. Goodfellow (2016) Deep Learning. MIT Press.

Unfolded RNNs

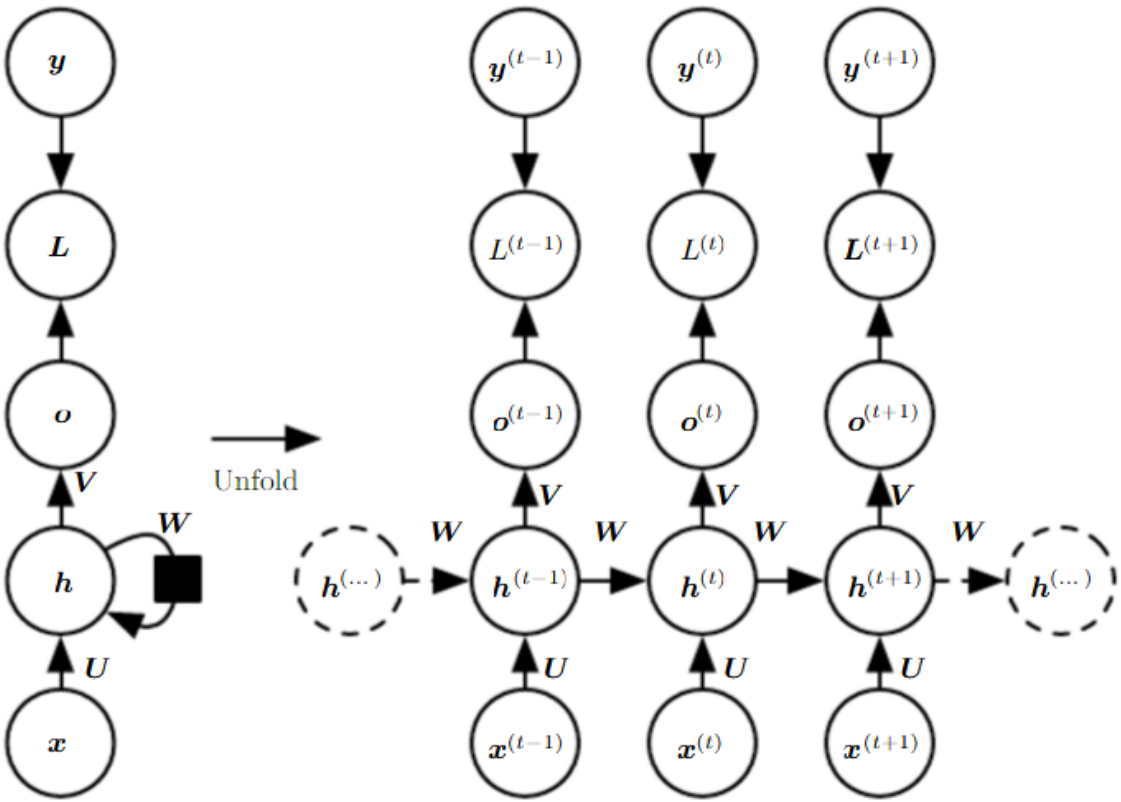
- RNNs produce an output at each time step and have **recurrent connections** between hidden units.
- RNNs read an entire sequence for processing and finally produce a single output.



I. Goodfellow (2016) Deep Learning. MIT Press.

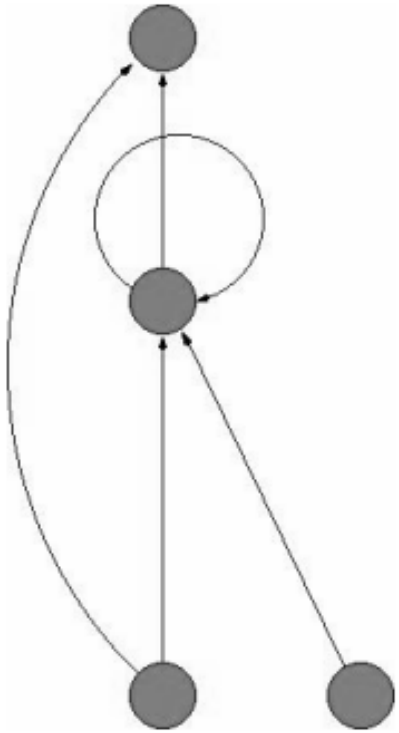
Unfolded RNNs

The notations are: Input x, state h, output o, loss function L, training target y, weights U, V, and W.



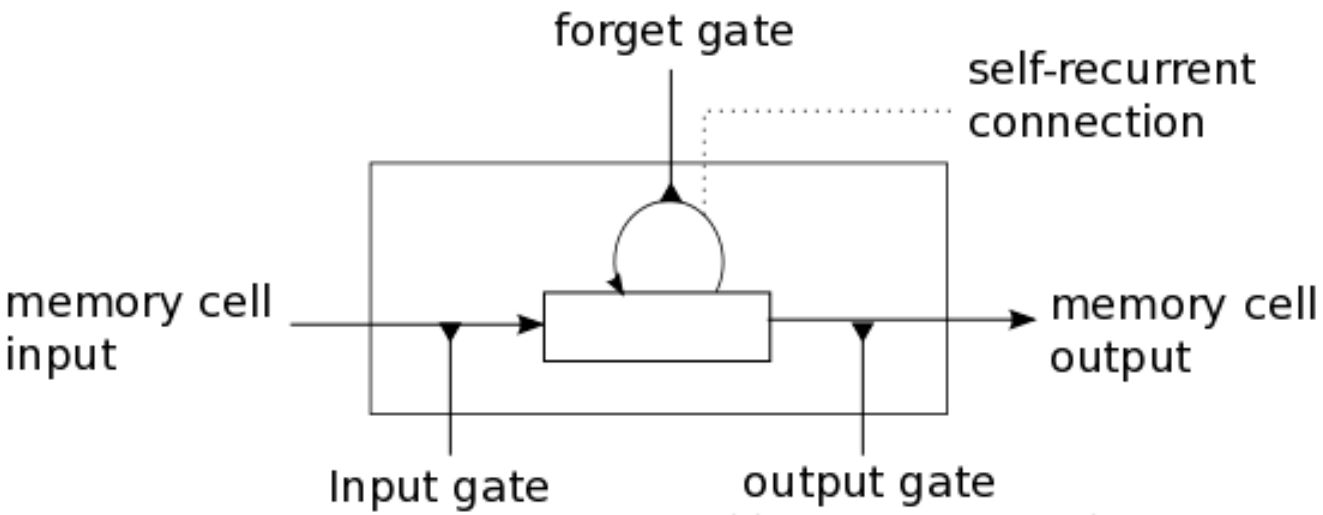
I. Goodfellow (2016) Deep Learning. MIT Press.

LSTM: Long Short-Term Memory

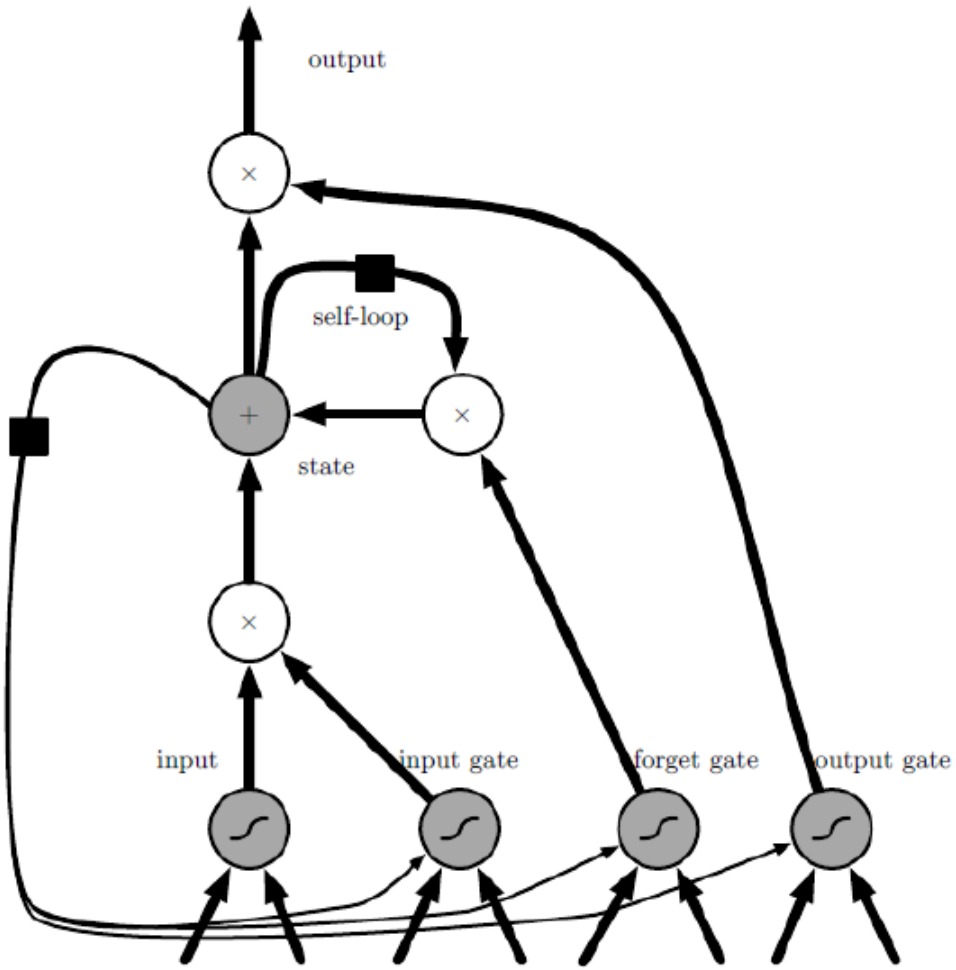


S. Hochreiter, et al. (1997) Long short-term memory, Neural computation, 9(8):1735-1780.

LSTM: Long Short-Term Memory

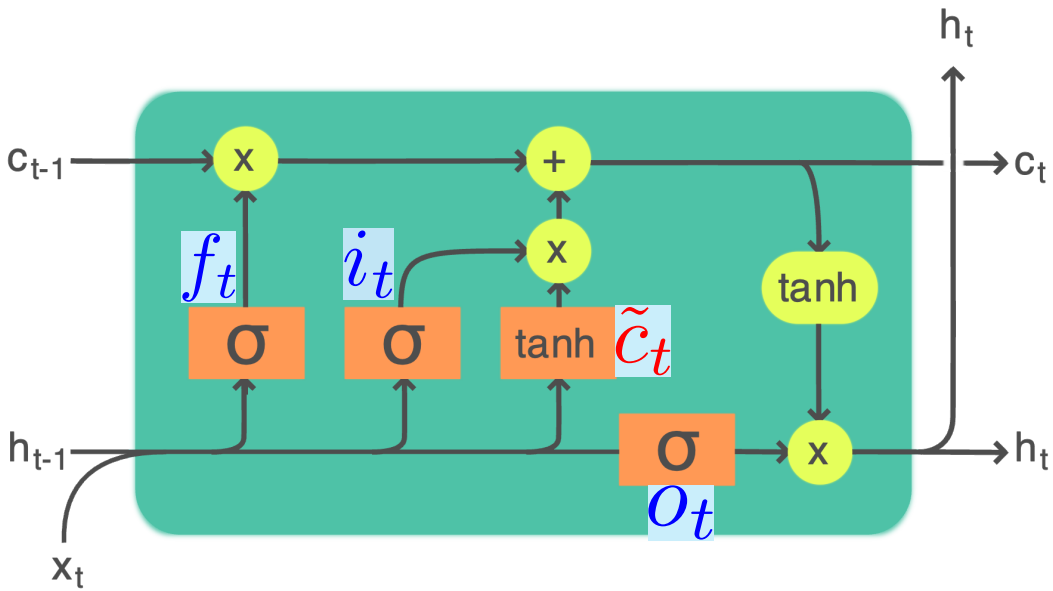


LSTM: Long Short-Term Memory



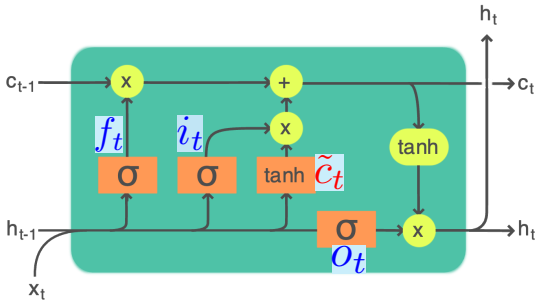
Page 411. I. Goodfellow (2016) Deep Learning. MIT Press.

LSTM: Long Short-Term Memory



https://en.wikipedia.org/wiki/Long_short-term_memory.

LSTM: Long Short-Term Memory



$$f_t = \sigma_g(W_f x_t + U_f h_{t-1} + b_f) \quad \text{Forget gate}$$
$$i_t = \sigma_g(W_i x_t + U_i h_{t-1} + b_i) \quad \text{Input gate}$$
$$o_t = \sigma_g(W_o x_t + U_o h_{t-1} + b_o) \quad \text{Output gate}$$
$$\tilde{c}_t = \sigma_c(W_c x_t + U_c h_{t-1} + b_c) \quad \text{Cell state}$$
$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \quad \text{Cell Update}$$
$$h_t = o_t \odot \sigma_h(c_t) \quad \text{Output}$$

https://en.wikipedia.org/wiki/Long_short-term_memory.

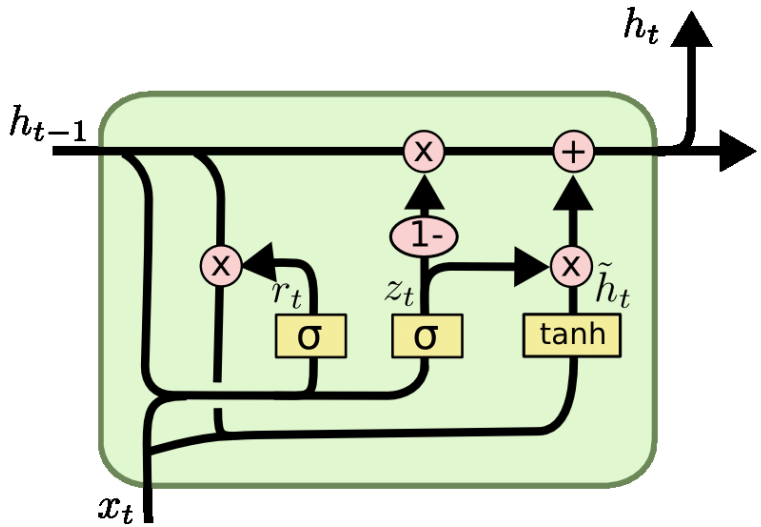
LSTM: Long Short-Term Memory

- Cell state c_t represents **long-term** memory
  Hidden state h_t represents **short-term** memory
- Three **gates**: input gate, forget gate, and output gate.
- LSTM gates compute an activation, often using the logistic function.
- LSTM is well-suited to *classify*, *process* and *predict* time series given time lags of unknown size and duration between important events.
- LSTM was developed to deal with the *exploding* and *vanishing* gradient problems.

Web: https://en.wikipedia.org/wiki/Long_short-term_memory.

GRU: Gated Recurrent Unit

Only two gates: reset and update
z_t and $1 - z_t$ updates.



r_t : reset gate

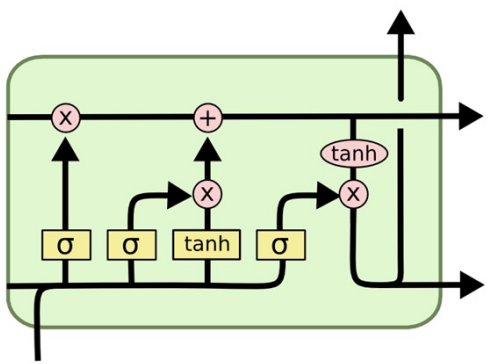
z_t : update gate

Web: https://en.wikipedia.org/wiki/Gated_recurrent_unit

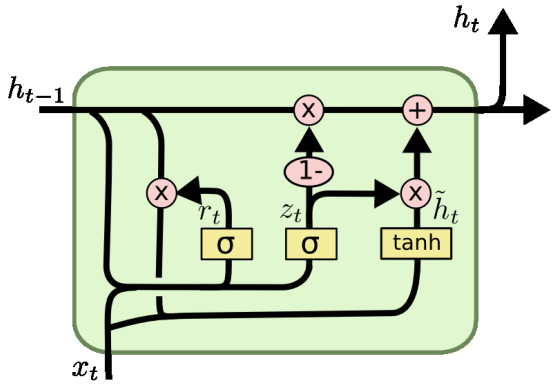
GRU: Gated Recurrent Unit

Compare LSTM and GRU

- 3 gates vs. 2 gates
- More parameters vs. Fewer parameters
- Higher computation vs. lower computation
- Complex problems (intricate) vs. Smaller datasets (efficient)



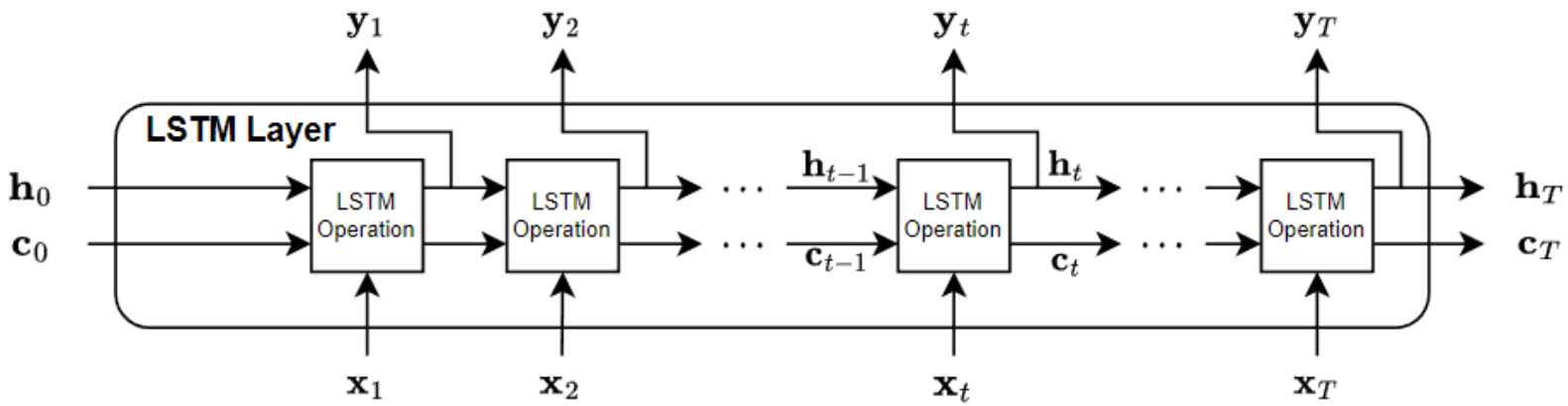
LSTM



GRU

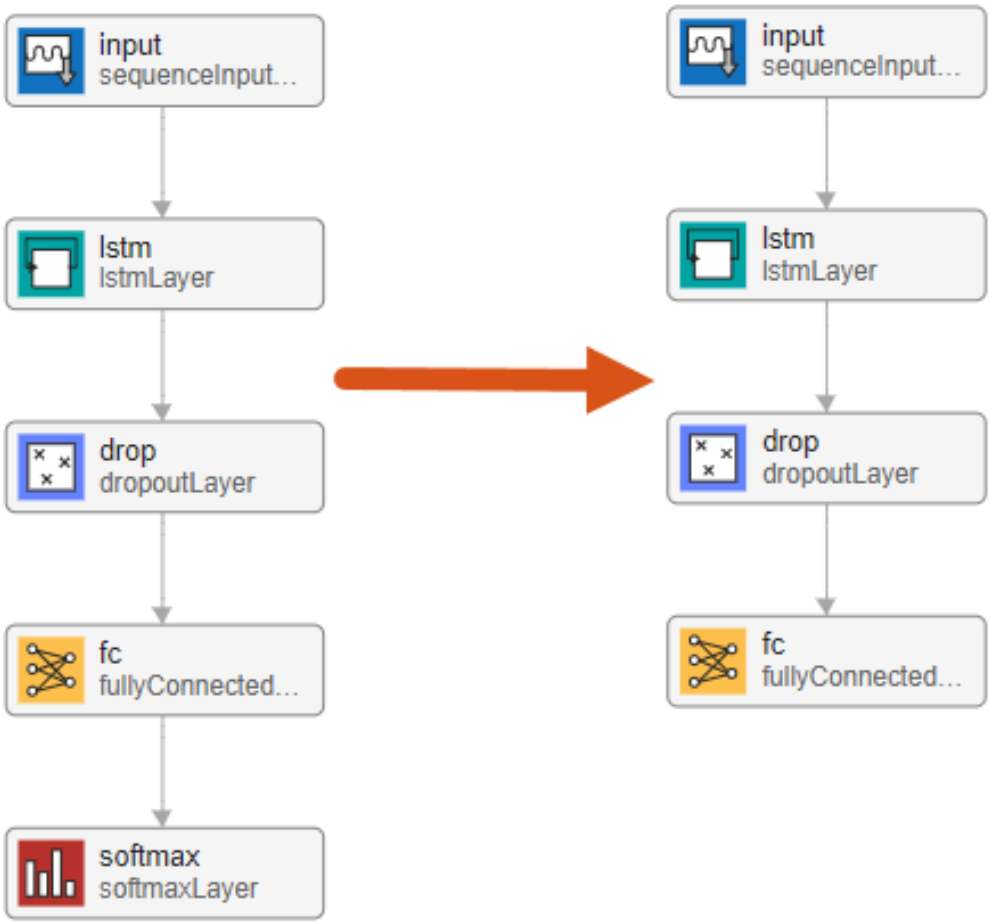
Questions?

MATLAB LSTM Architecture



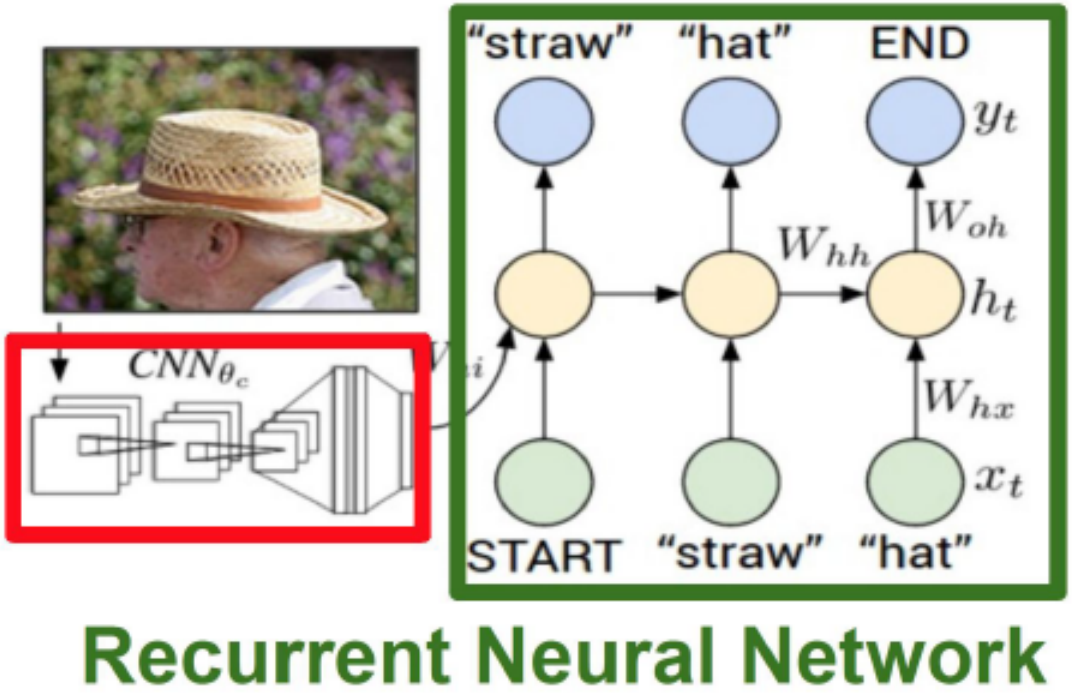
https://au.mathworks.com/help/deeplearning/ug/long-short-term-memory-networks.html

MATLAB LSTM in Time Series Analysis



https://au.mathworks.com/help/deeplearning/ug/time-series-forecasting-using-deep-network-designer.html

ConvLSTM: Convolutional LSTM



https://people.cs.pitt.edu/ kovashka/cs2770_sp17/vision_14_rnns.pdf

Transformer Networks

Rise of Transformer Model

- First designed for Natural Language Processing (NLP).
- Then extended to diverse fields: *computer vision, audio, video, etc.*
- Now the most powerful network architecture.
- Suitable for efficient and effective **time-series** prediction.

Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com

Noam Shazeer* Google Brain noam@google.com

Niki Parmar* Google Research nikip@google.com

Jakob Uszkoreit* Google Research usz@google.com

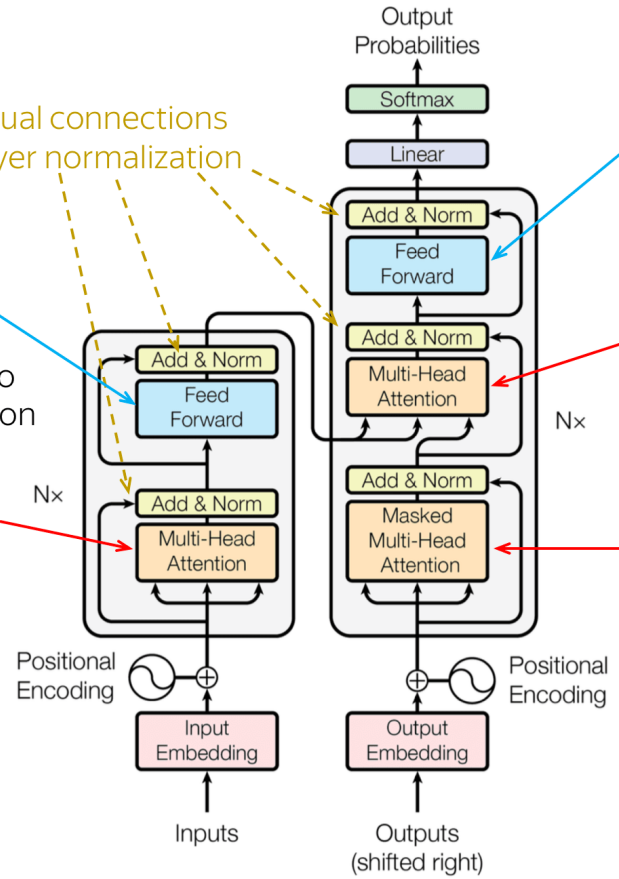
Llion Jones* Google Research llion@google.com

Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu

Łukasz Kaiser* Google Brain lukaszkaiser@google.com

Illia Polosukhin* ‡ illia.polosukhin@gmail.com

Transformer Architecture



Residual connections and layer normalization

Feed-forward network: after taking information from other tokens, take a moment to think and process this information

Feed-forward network: after taking information from other tokens, take a moment to think and process this information

Encoder self-attention: tokens look at each other

queries, keys, values are computed from encoder states

Decoder-encoder attention: target token looks at the source

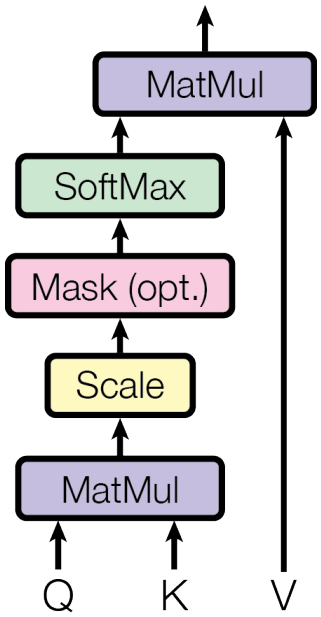
queries – from decoder states; keys and values from encoder states

Decoder self-attention (masked): tokens look at the previous tokens

queries, keys, values are computed from decoder states

Multi-head Attention

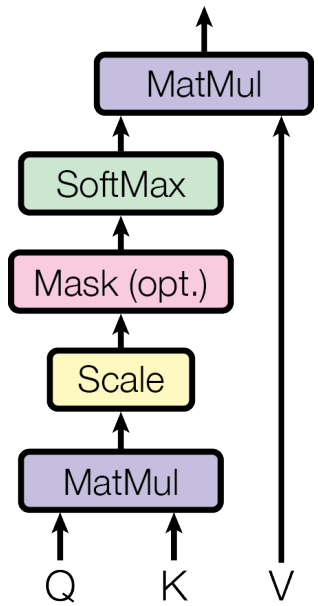
Scaled Dot-Product Attention



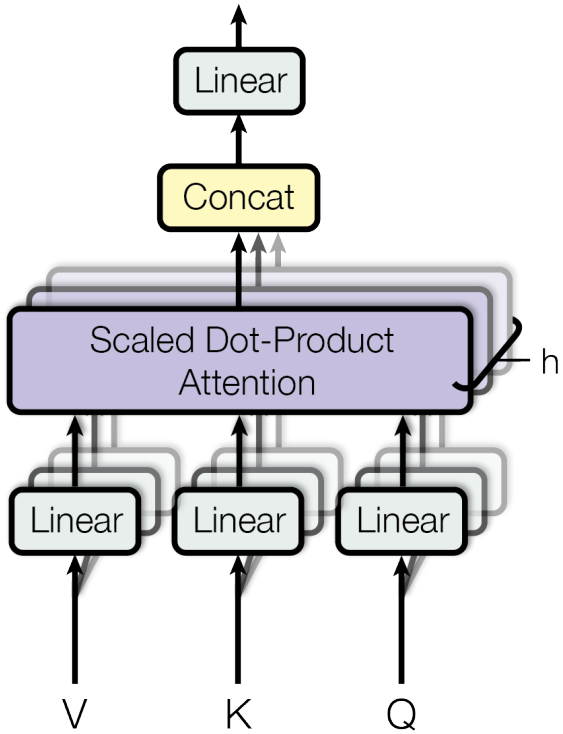
$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

Multi-head Attention

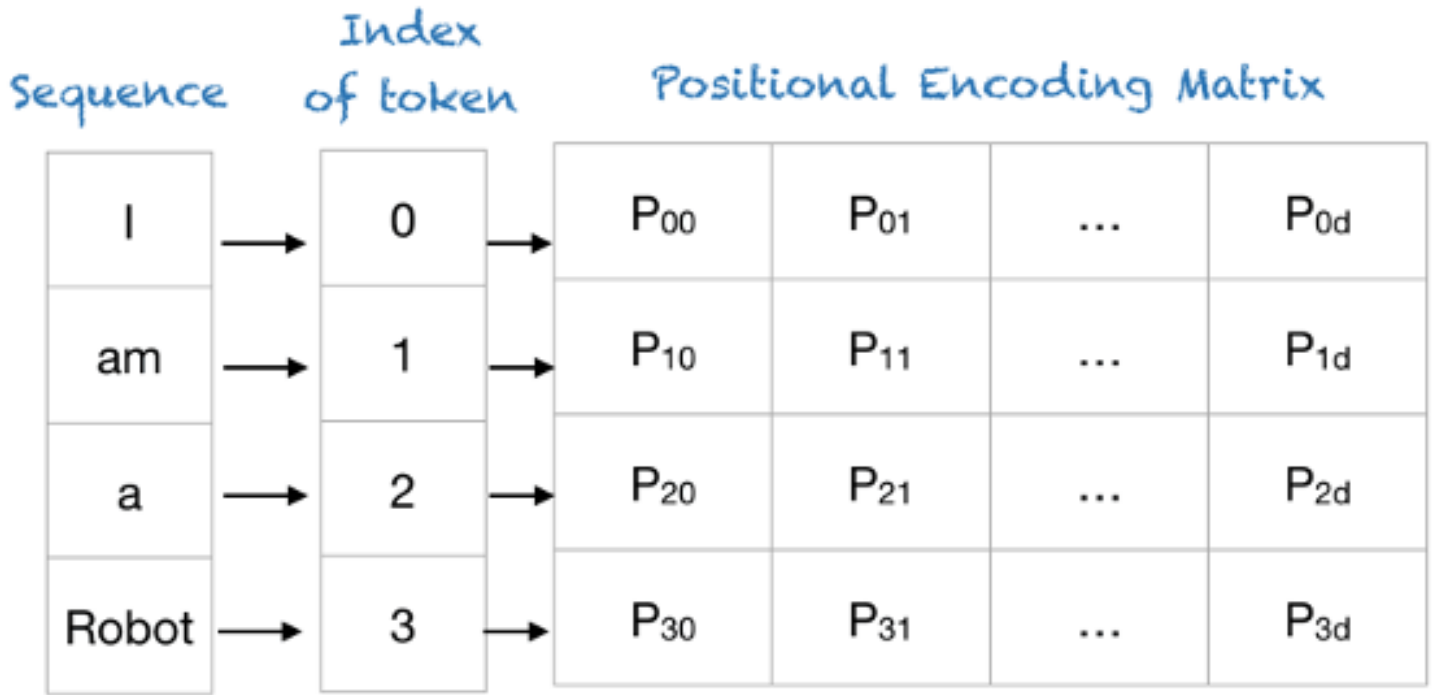
Scaled Dot-Product Attention



Multi-Head Attention

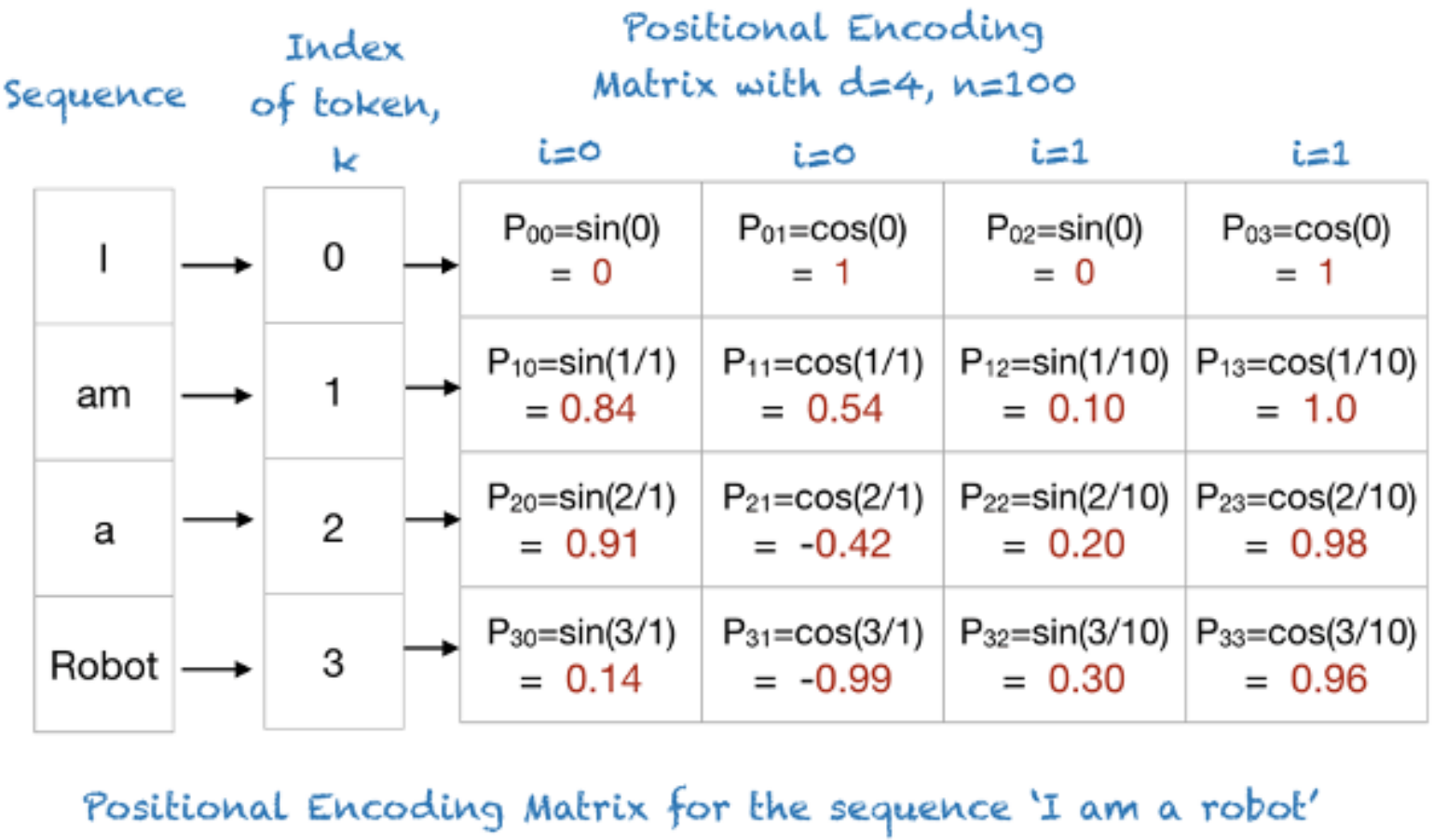


Position Encoding



Positional Encoding Matrix for the sequence 'I am a robot'

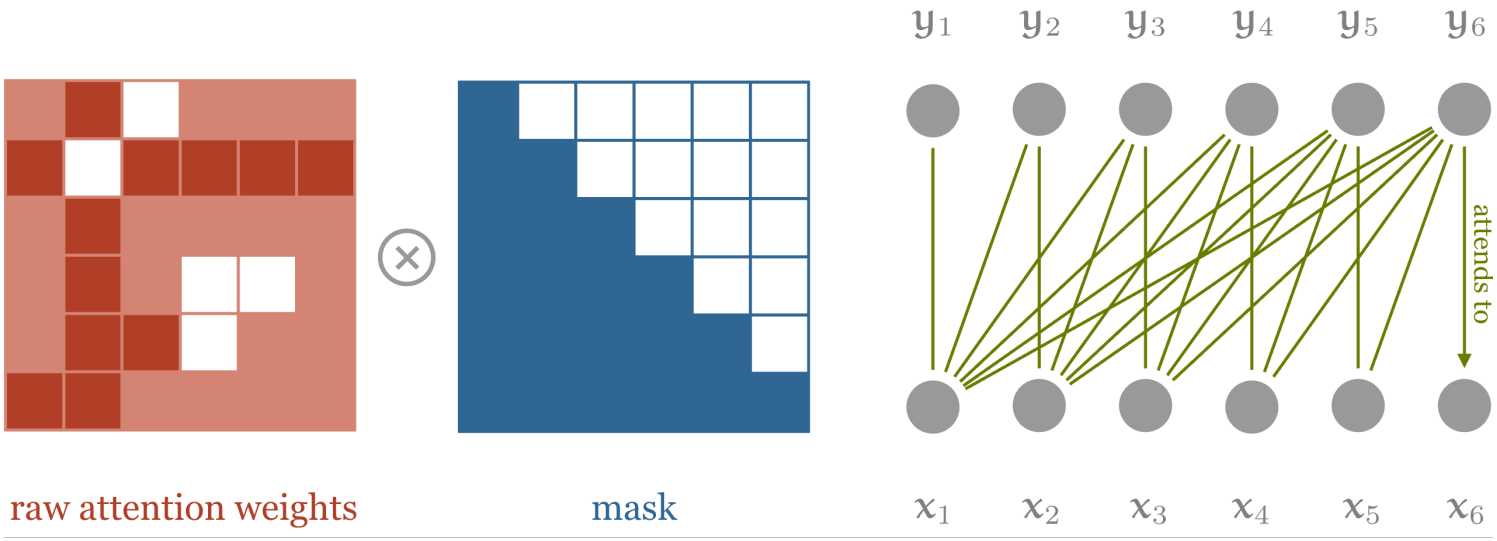
Position Encoding



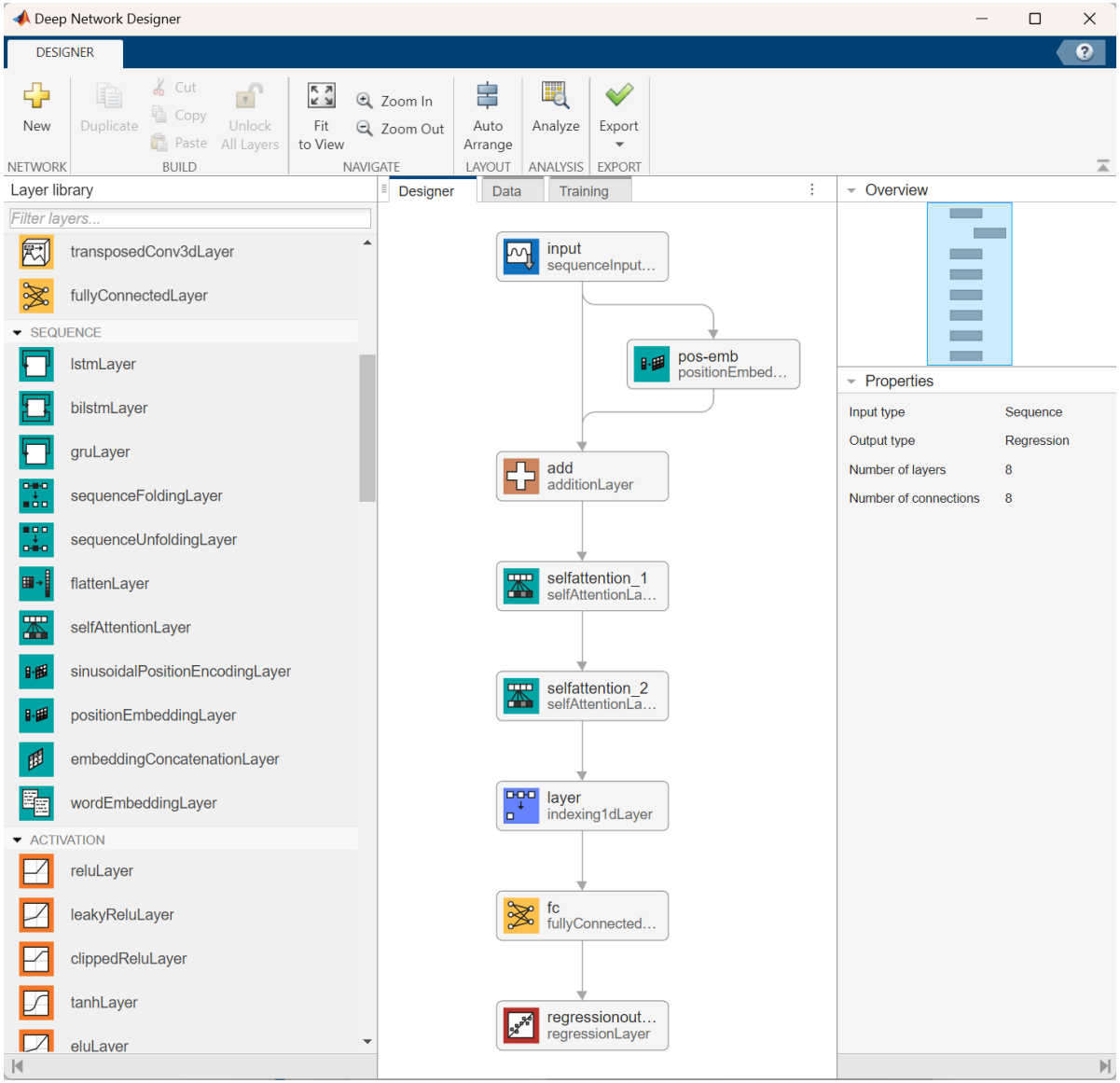
Positional Encoding Matrix for the sequence 'I am a robot'

Mask Attention Mechanism

- Predicting y_t can only use $x_1, \ldots, x_{t-1}$.
- The model can only *use past rather than future information.*
- For example, you cannot use tomorrow's weather/stock to predict today or yesterday.



Transformer Networks in MATLAB



https://au.mathworks.com/matlabcentral/fileexchange/161016-transformer-networks-for-time-series-prediction

Questions?