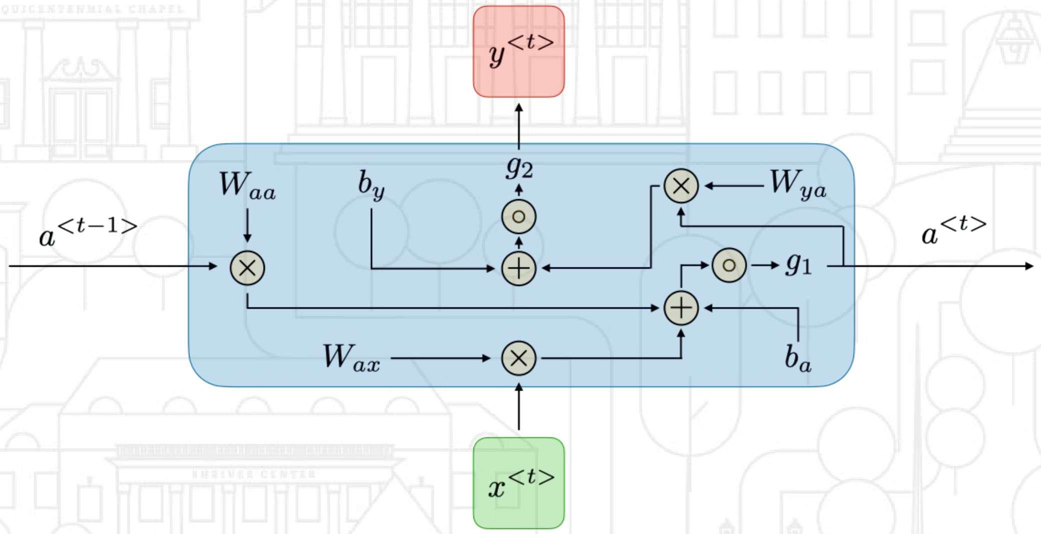
Module 6

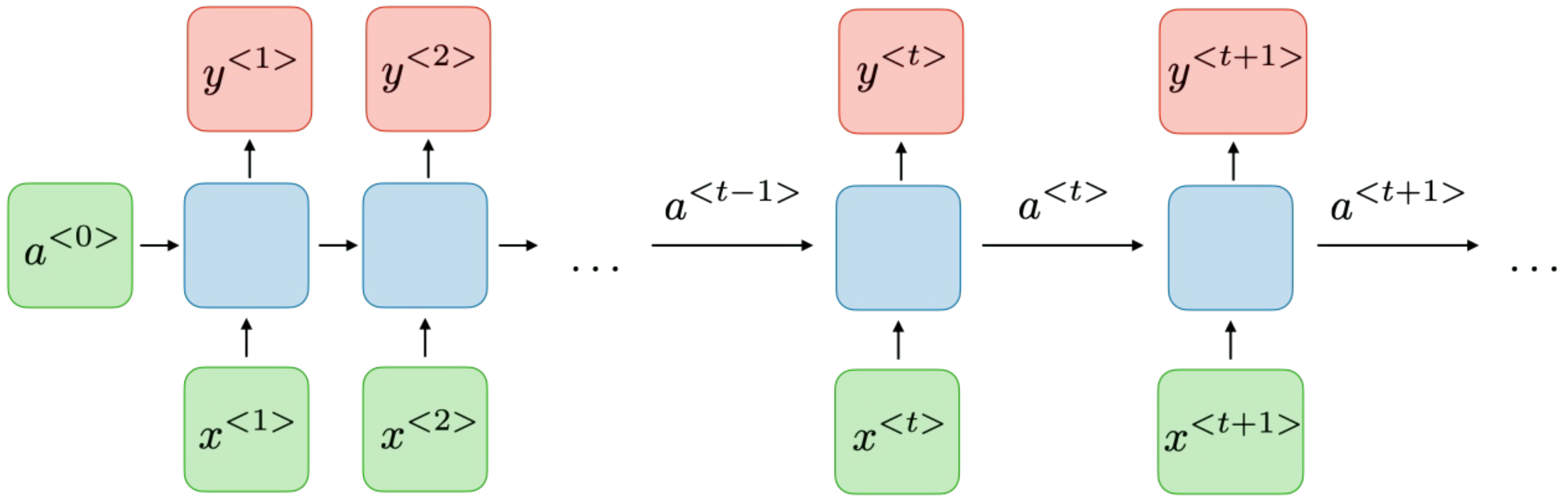
Recurrent Neural Networks

DL: RNNs



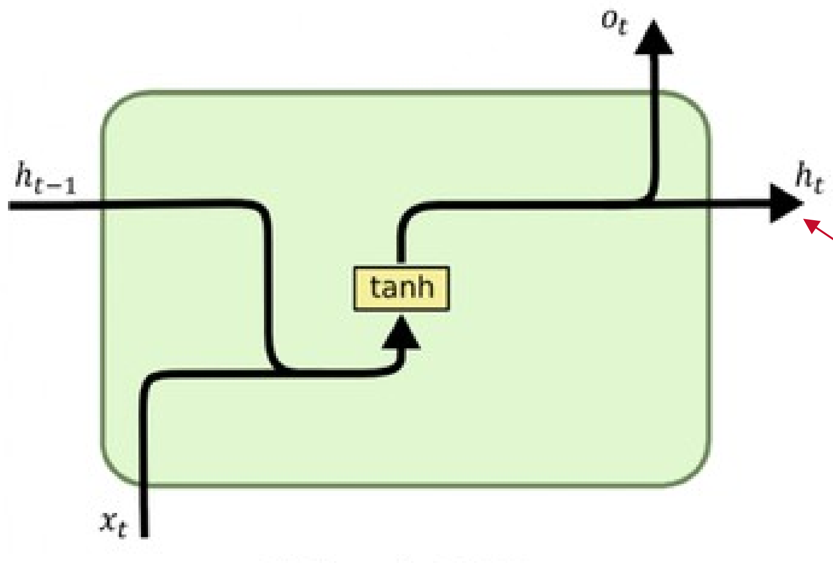
Recurrent Neural Networks

- Composed on cells with recurrent loops



Recurrent Neural Networks

- Simple RNN cells pass a hidden state over the loop
- The hidden state keeps information about previous values

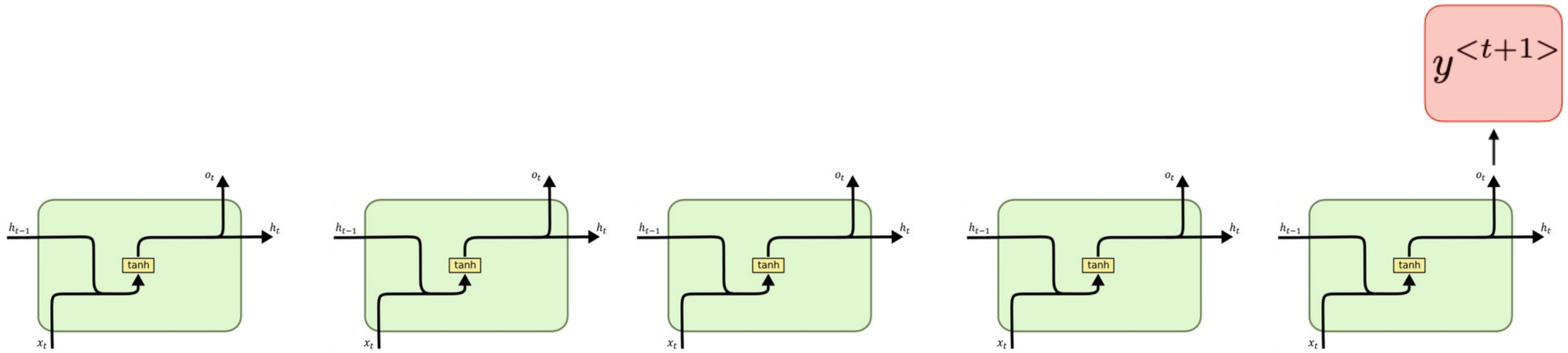


(a) Simple RNN

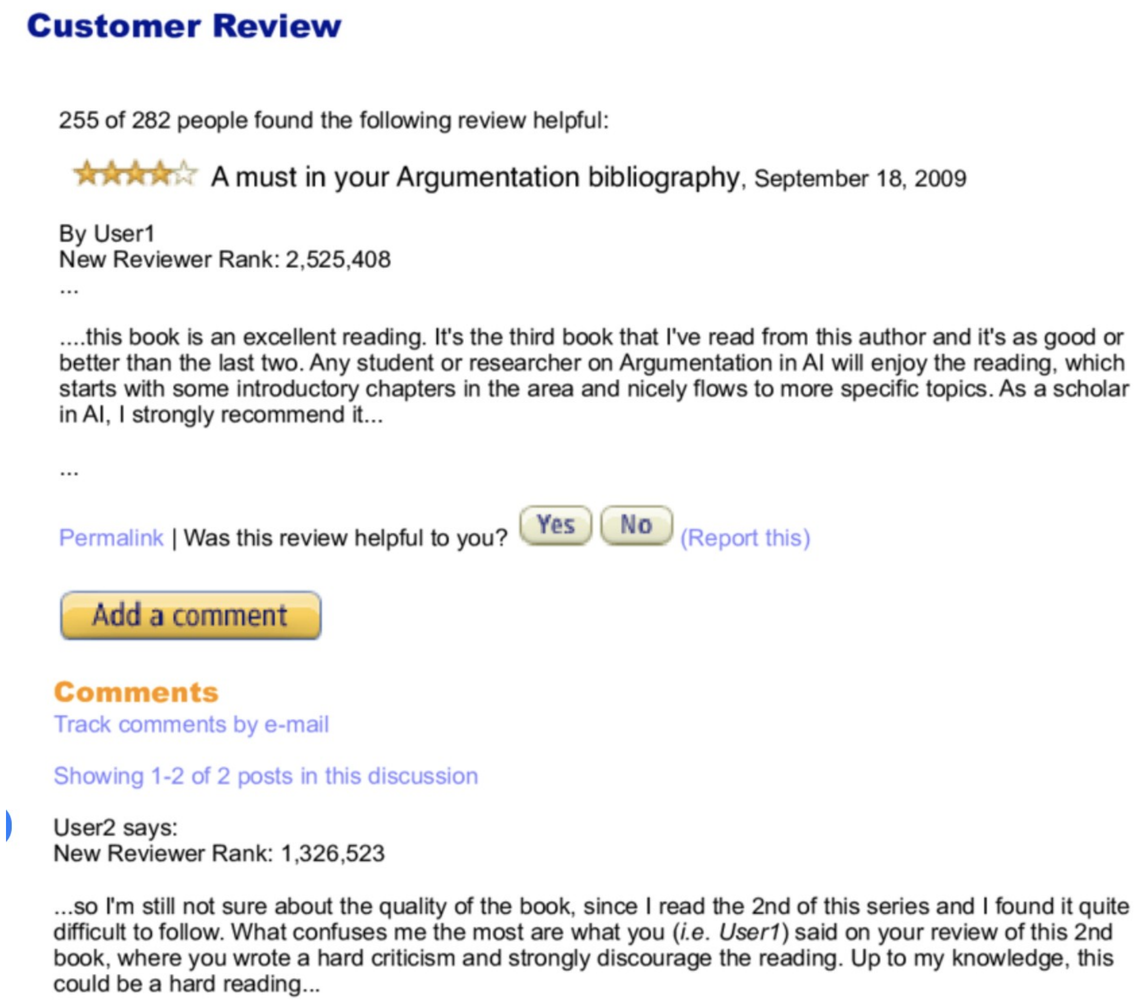
HIDDEN STATE

Back Propagation over Time

- Error is propagated backward over time to update the weights
- Vanishing gradient or exploding gradients are issues
- Early values of the sequence have less importance
- Short-term memory is lost fast



Back Propagation over Time



Customer Review

255 of 282 people found the following review helpful:

★★★★☆ A must in your Argumentation bibliography, September 18, 2009

By User1
New Reviewer Rank: 2,525,408
...

....this book is an excellent reading. It's the third book that I've read from this author and it's as good or better than the last two. Any student or researcher on Argumentation in AI will enjoy the reading, which starts with some introductory chapters in the area and nicely flows to more specific topics. As a scholar in AI, I strongly recommend it...

...

Comments
Track comments by e-mail

Showing 1-2 of 2 posts in this discussion

User2 says:
New Reviewer Rank: 1,326,523

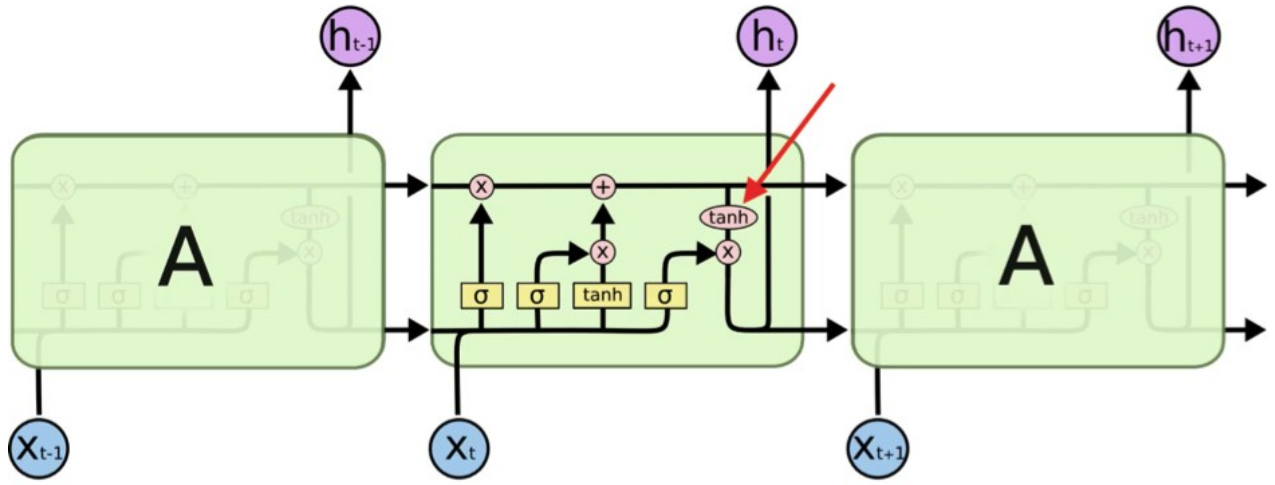
...so I'm still not sure about the quality of the book, since I read the 2nd of this series and I found it quite difficult to follow. What confuses me the most are what you (*i.e. User1*) said on your review of this 2nd book, where you wrote a hard criticism and strongly discourage the reading. Up to my knowledge, this could be a hard reading...

Text as sequences

- Sequences are important in text learning
- Words in a sentence **depend on the words before and after** them.
- Meaning often changes based on **order** and **context**
  - A dog bit Dr. Martinez
  - Dr. Martinez bit the hot dog

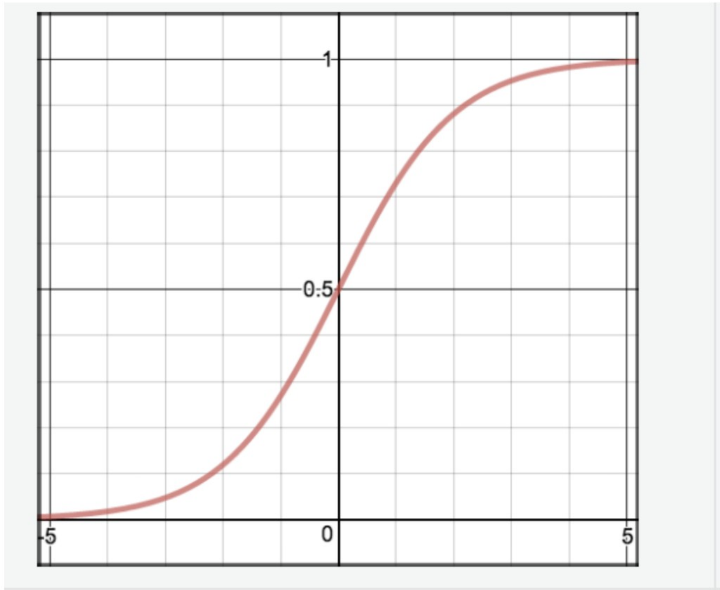
Long Short-Term Memory (LSTM)

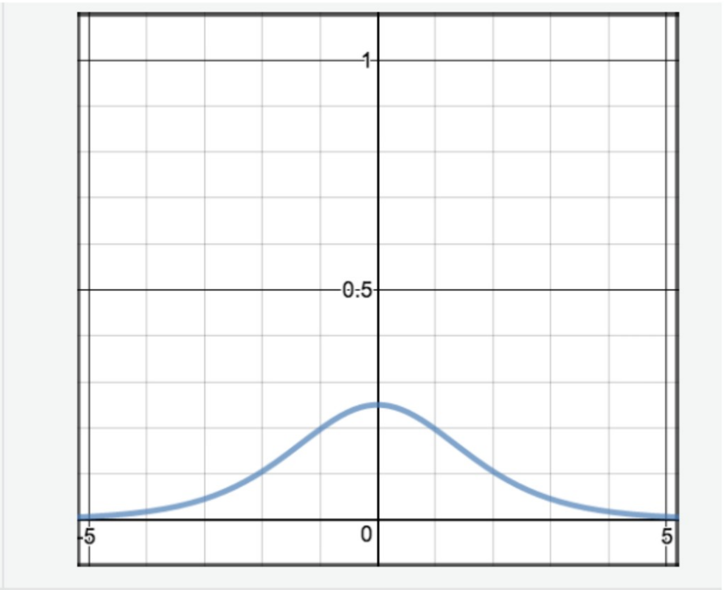
- It is a type of RNN (works with sequences)
- Deals with the vanishing gradient problem
- Allows information to persist (long-term)



Review Activations: Sigmoid / Logistic

- Given an input signal z, it returns $f(z) = \frac{1}{1+e^{-z}}$
- Derivative $f'(z) = f(z)(1 - f(z))$



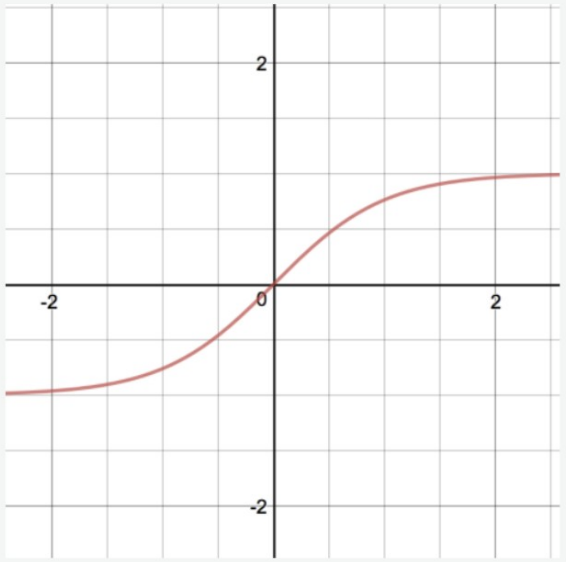


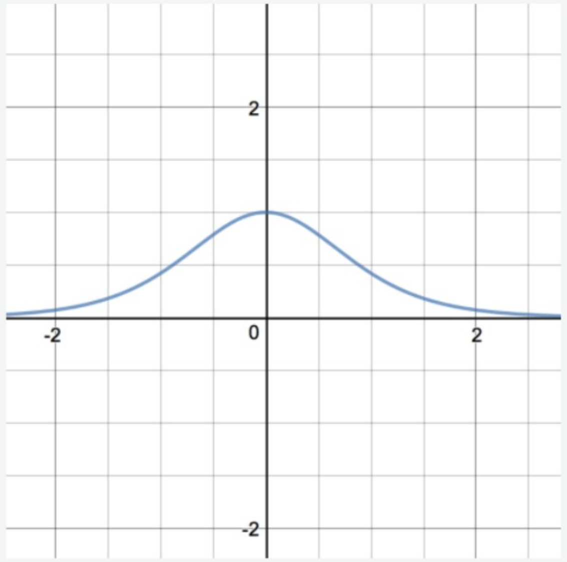
```
def sigmoid(z):
  return 1.0 / (1 + np.exp(-z))
```

```
def sigmoid_prime(z):
  return sigmoid(z) * (1-sigmoid(z))
```

Review Activations: ”tanh”

- Given an input signal z, it returns $f(z) = \frac{e^{z}-e^{-z}}{e^{z}+e^{-z}}$
- Derivative $f'(z) = 1 - f(z)^2$



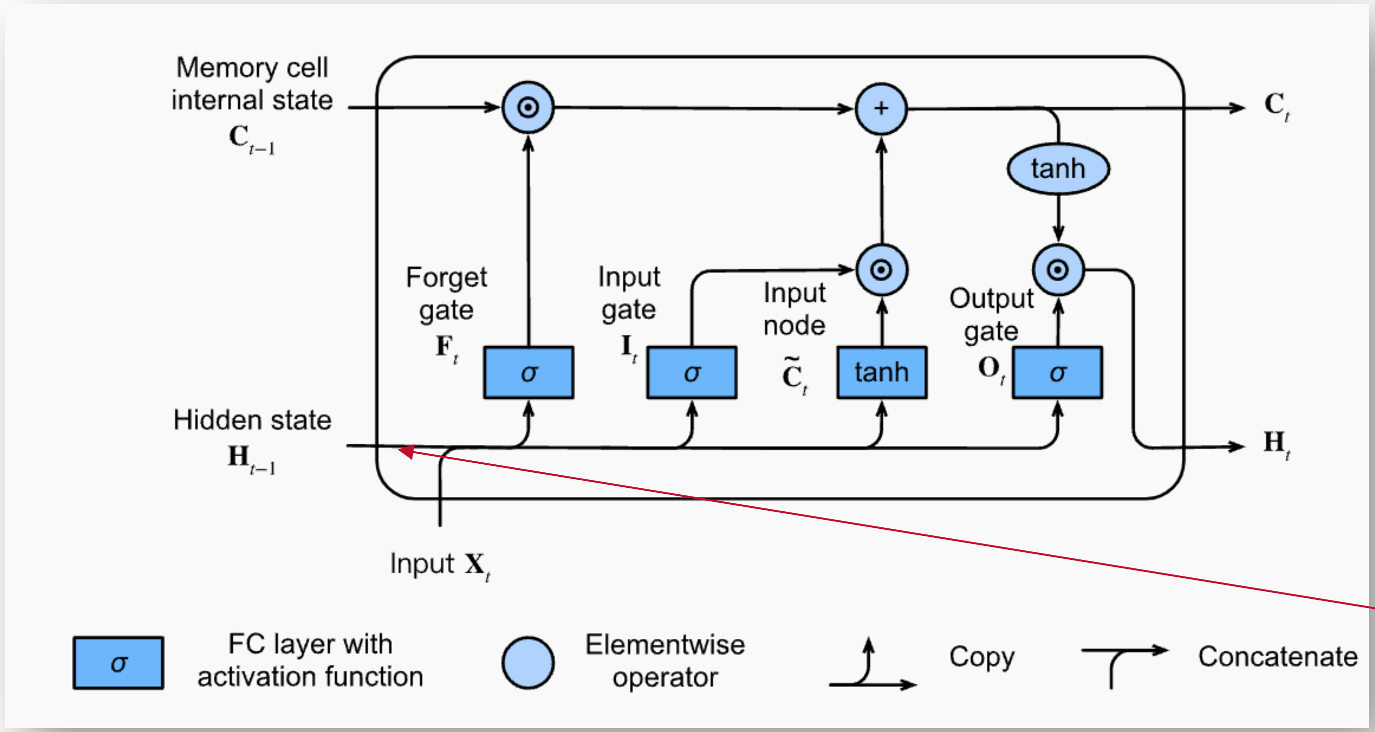


```
def tanh(z):
        return (np.exp(z) - np.exp(-z)) / (np.exp(z) + np.exp(-z))
```

```
def tanh_prime(z):
        return 1 - np.power(tanh(z), 2)
```

Long Short-Term Memory (LSTM)

- Neurons are composed on multiple operations

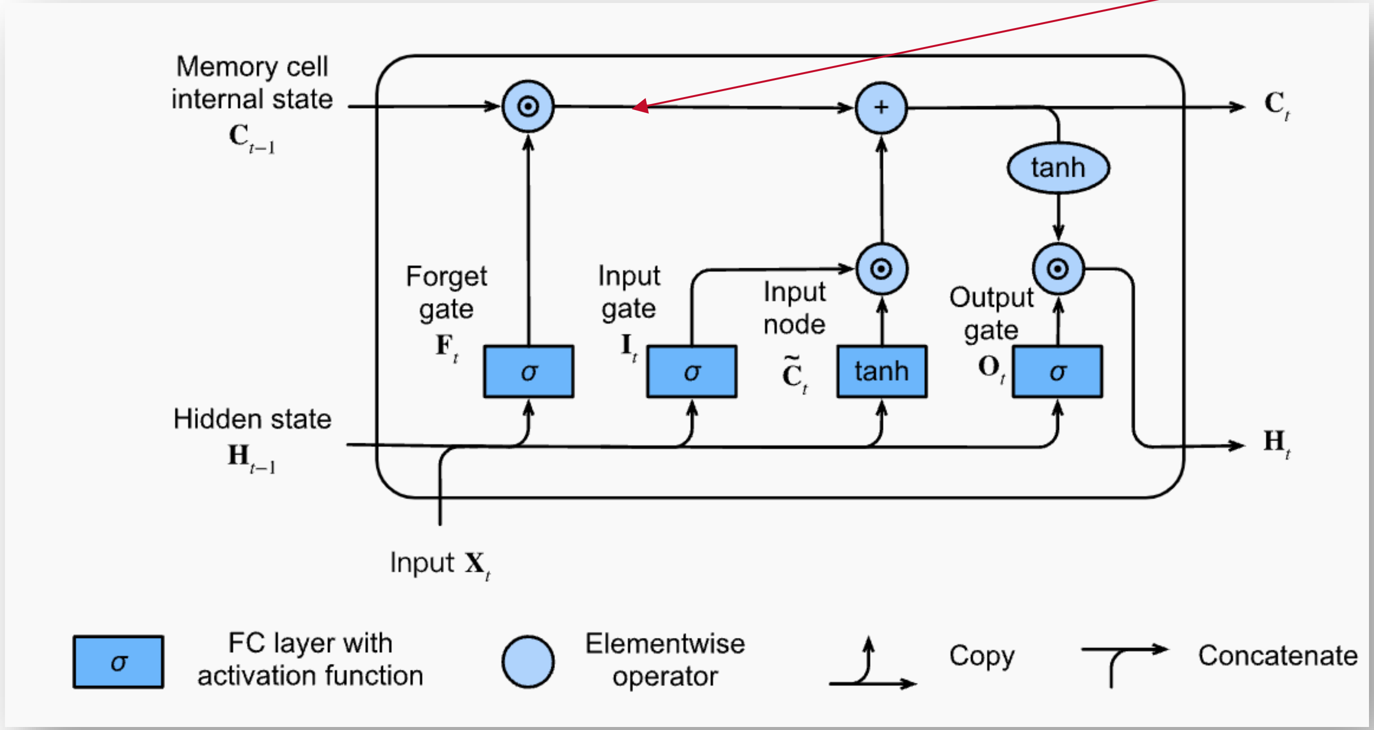

Memory cell internal state $\mathbf{C}_{t-1}$
$\mathbf{C}_t$
tanh
Forget gate $\mathbf{F}_t$
Input gate $\mathbf{I}_t$
Input node $\tilde{\mathbf{C}}_t$
Output gate $\mathbf{O}_t$
σ
tanh
Hidden state $\mathbf{H}_{t-1}$
$\mathbf{H}_t$
Input $\mathbf{X}_t$
FC layer with activation function
Elementwise operator
Copy
Concatenate

SHORT-TERM MEMORY (HIDDEN STATE)

Long Short-Term Memory (LSTM)

- Neurons are composed on multiple operations

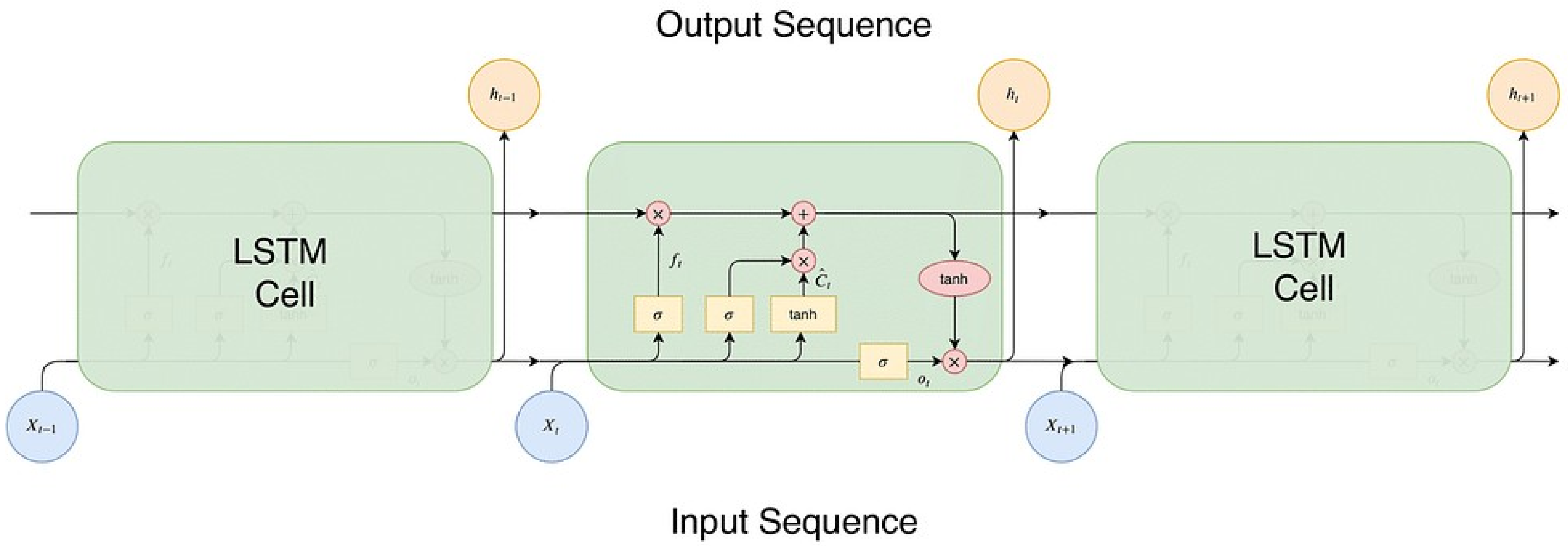
LONG-TERM MEMORY (INTERNAL STATE)



Memory cell internal state $\mathbf{C}_{t-1}$
Forget gate $\mathbf{F}_t$
Input gate $\mathbf{I}_t$
Input node $\tilde{\mathbf{C}}_t$
Output gate $\mathbf{O}_t$
tanh
σ
+
$\mathbf{C}_t$
Hidden state $\mathbf{H}_{t-1}$
$\mathbf{H}_t$
Input $\mathbf{X}_t$
FC layer with activation function
Elementwise operator
Copy
Concatenate

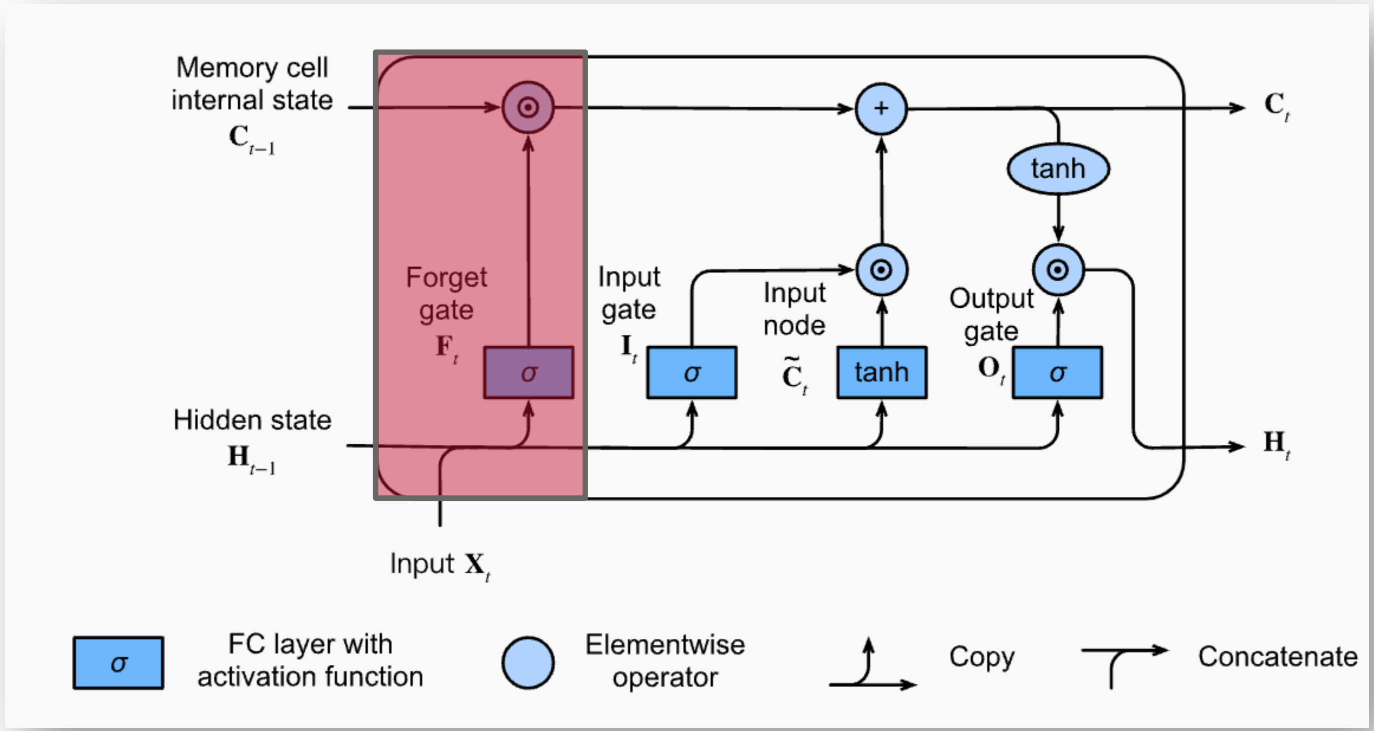
Long Short-Term Memory (LSTM)

- Neurons are composed on multiple operations



LSTM Forget Gate

- determines how much of the previous data is kept or forgotten



Memory cell internal state $\mathbf{C}_{t-1}$
Forget gate $\mathbf{F}_t$
Input gate $\mathbf{I}_t$
Input node $\tilde{\mathbf{C}}_t$
Output gate $\mathbf{O}_t$
Hidden state $\mathbf{H}_{t-1}$
Input $\mathbf{X}_t$
$\mathbf{C}_t$
$\mathbf{H}_t$
σ
tanh
FC layer with activation function
Elementwise operator
Copy
Concatenate

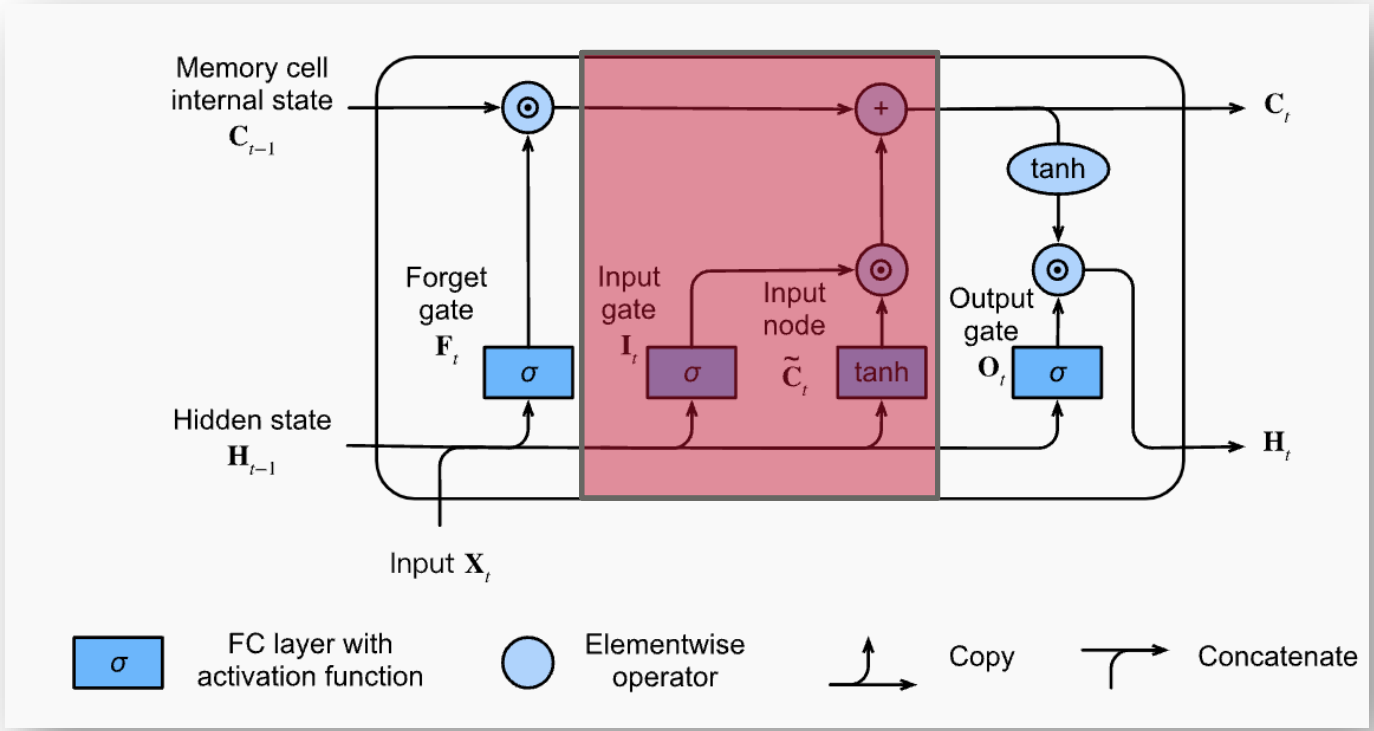
$$\mathbf{F}_t = \sigma(\mathbf{X}_t \mathbf{W}_{\mathrm{xf}} + \mathbf{H}_{t-1} \mathbf{W}_{\mathrm{hf}} + \mathbf{b}_{\mathrm{f}})$$

The output will be a value between 0 and 1

$\boldsymbol{F_t}$ is multiplied times the internal state $\boldsymbol{C_{t-1}}$

LSTM Input Gate and Input Node

- Creates potential long-term memories


Memory cell internal state $\mathbf{C}_{t-1}$
Forget gate $\mathbf{F}_t$
Input gate $\mathbf{I}_t$
Input node $\tilde{\mathbf{C}}_t$
Output gate $\mathbf{O}_t$
σ
tanh
+
$\mathbf{C}_t$
$\mathbf{H}_t$
Hidden state $\mathbf{H}_{t-1}$
Input $\mathbf{X}_t$
FC layer with activation function
Elementwise operator
Copy
Concatenate

$$\mathbf{I}_t = \sigma(\mathbf{X}_t\mathbf{W}_{\text{xi}} + \mathbf{H}_{t-1}\mathbf{W}_{\text{hi}} + \mathbf{b}_{\text{i}})$$

The output will be a value between 0 and 1

Will represent %memorized from input node

$$\tilde{\mathbf{C}}_t = \tanh(\mathbf{X}_t\mathbf{W}_{\text{xc}} + \mathbf{H}_{t-1}\mathbf{W}_{\text{hc}} + \mathbf{b}_{\text{c}}).$$

Represents the part of short-term to keep

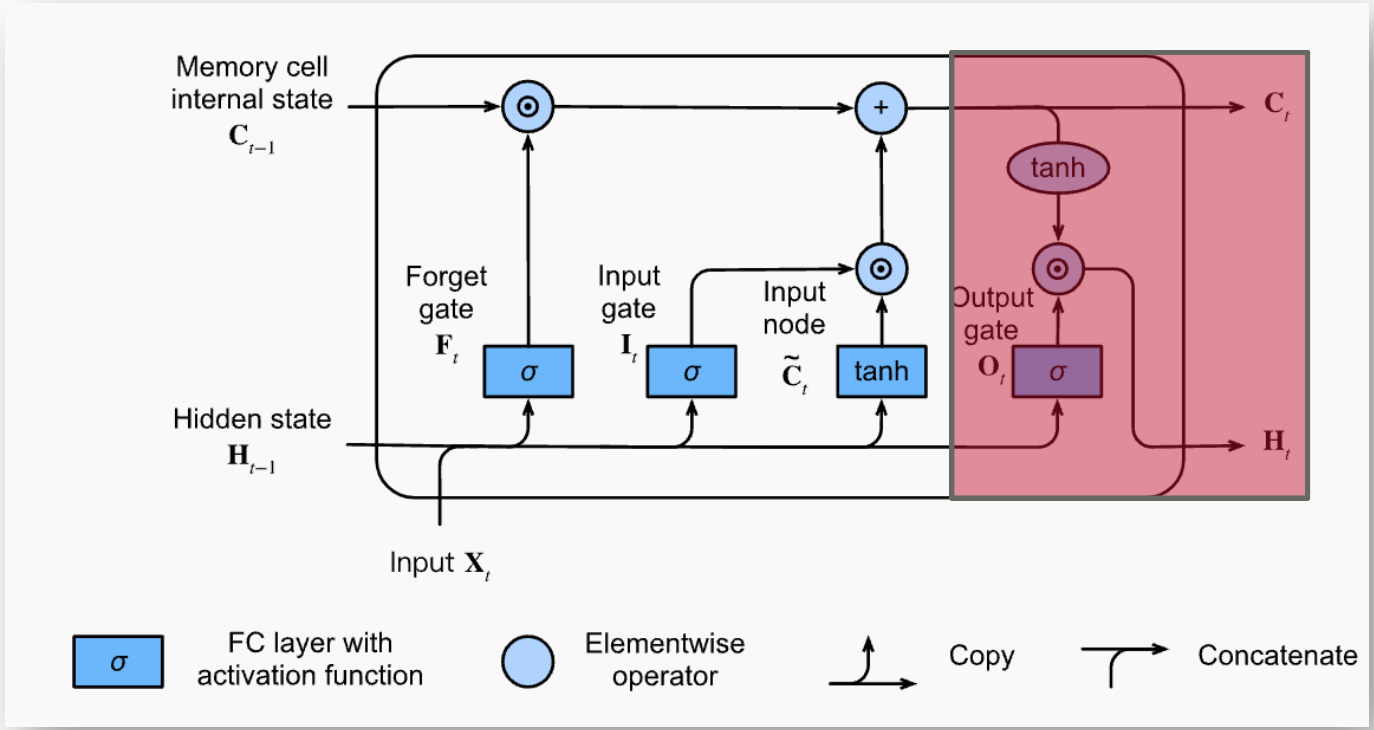
$$\mathbf{C}_t = \mathbf{F}_t \odot \mathbf{C}_{t-1} + \mathbf{I}_t \odot \tilde{\mathbf{C}}_t$$

This is how internal state is modified at this point

Forget + Remembered

LSTM Output Gate

- Creates potential long-term memories



Memory cell internal state $\mathbf{C}_{t-1}$
Forget gate $\mathbf{F}_t$
Input gate $\mathbf{I}_t$
Input node $\tilde{\mathbf{C}}_t$
Output gate $\mathbf{O}_t$
Hidden state $\mathbf{H}_{t-1}$
Input $\mathbf{X}_t$
$\mathbf{C}_t$
$\mathbf{H}_t$
σ
tanh
FC layer with activation function
Elementwise operator
Copy
Concatenate

$$\mathbf{O}_t = \sigma(\mathbf{X}_t \mathbf{W}_{\text{xo}} + \mathbf{H}_{t-1} \mathbf{W}_{\text{ho}} + \mathbf{b}_\text{o})$$

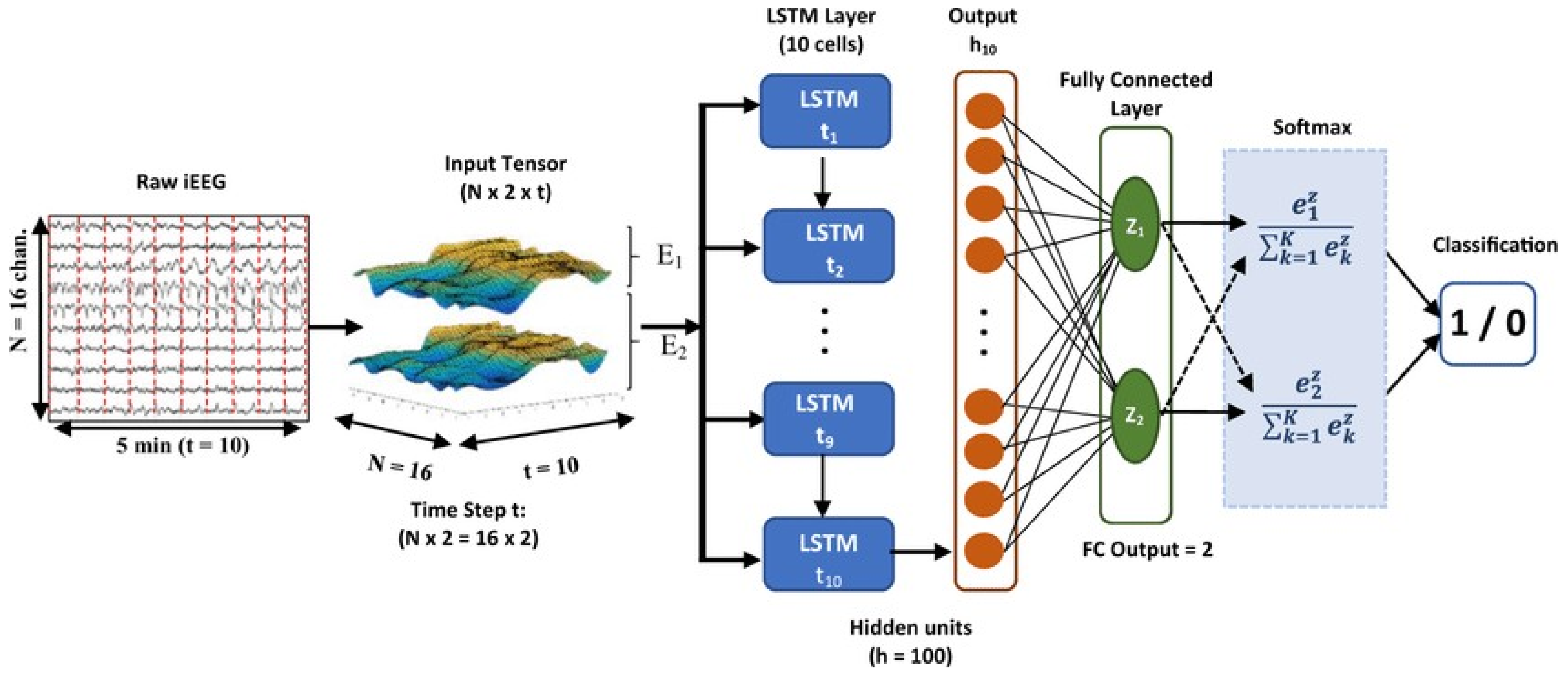
Will represent % output maintained

$$\mathbf{H}_t = \mathbf{O}_t \odot \tanh(\mathbf{C}_t).$$

Hidden state is updated with both internal state and output gate

LSTM Layers

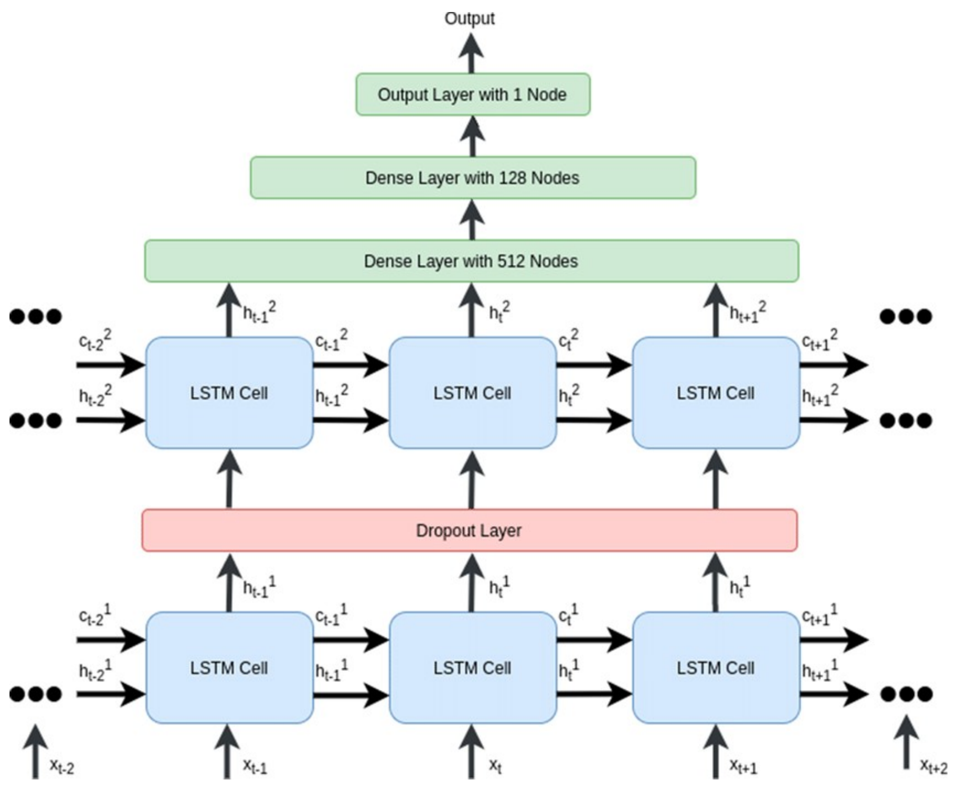
- We create multiple LSTM neurons to form a layer



Source: https://www.researchgate.net/figure/LSTM-neural-network-architecture-Input-layer-consists-of-10-sequential-time-steps-10_fig3_336900652

Stacking LSTM Layers

- We create multiple LSTM neurons to form a layer

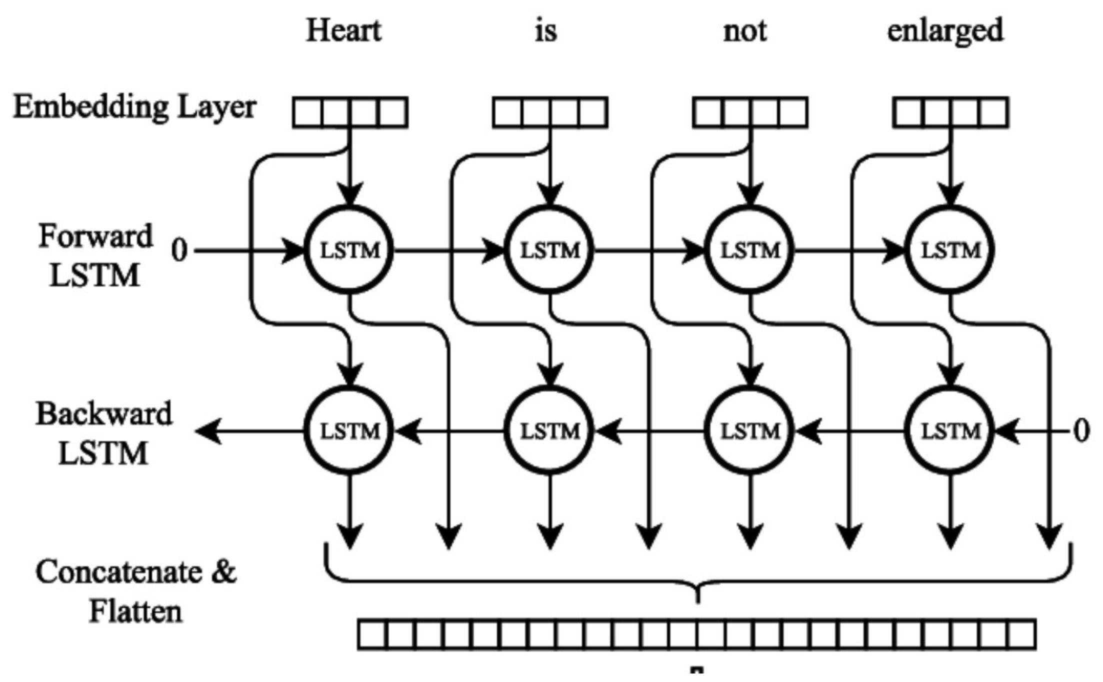


Source: https://www.researchgate.net/figure/Two-layered-stacked-LSTM-network_fig3_350105975

Python

Bidirectional LSTM Layers

- A Bidirectional LSTM (BiLSTM) runs two LSTM layers: forward and backward.
- It captures information from both past and future contexts.
- Enhances sequence understanding compared to a standard (unidirectional) LSTM.

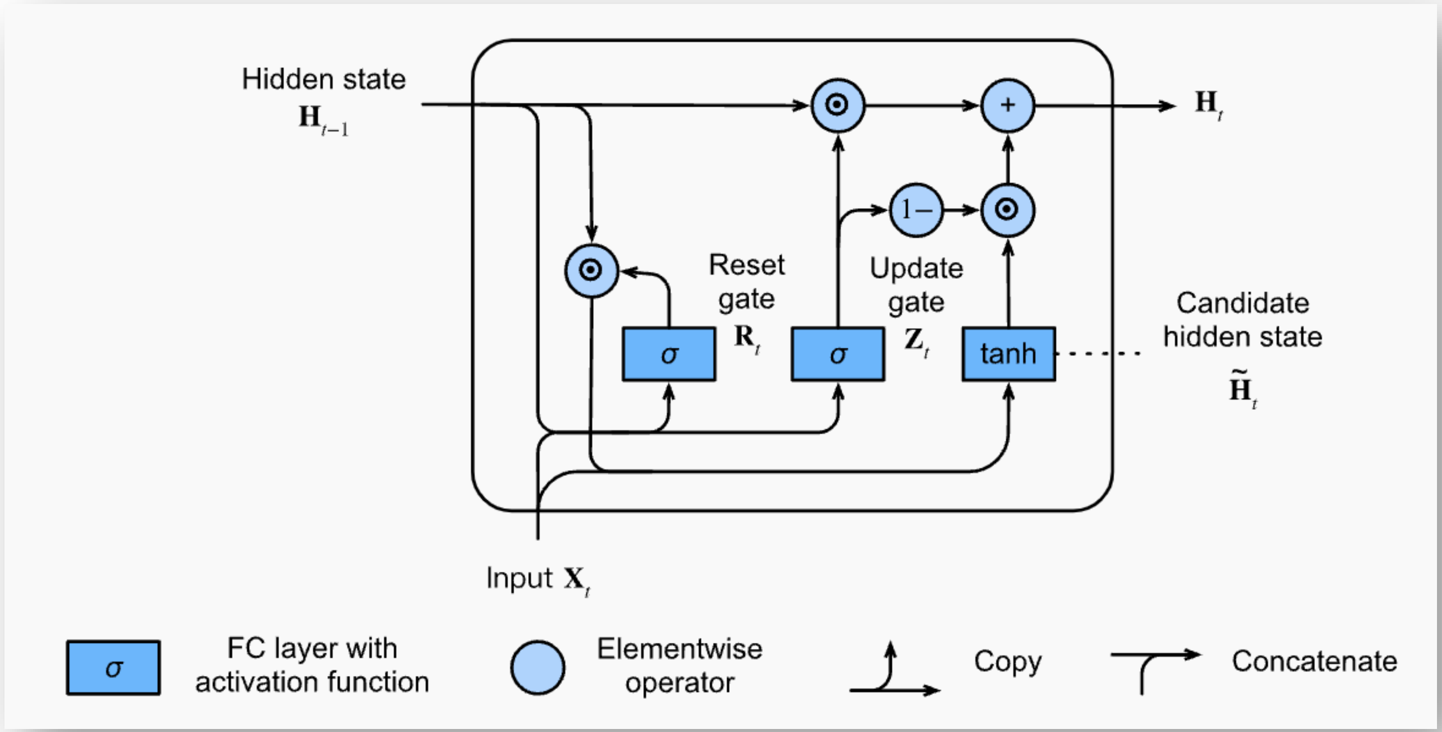


Source: Modelling Radiological Language with Bidirectional Long Short-Term Memory Networks, Cornegruta et al

Python

Gated Recurrent Unit (GRU)

- Streamlined version of the LSTM
- Achieves comparable performance


Hidden state $\mathbf{H}_{t-1}$
$\mathbf{H}_t$
Reset gate $\mathbf{R}_t$
Update gate $\mathbf{Z}_t$
Candidate hidden state $\tilde{\mathbf{H}}_t$
σ
σ
tanh
1−
+
Input $\mathbf{X}_t$
FC layer with activation function
Elementwise operator
Copy
Concatenate

Python