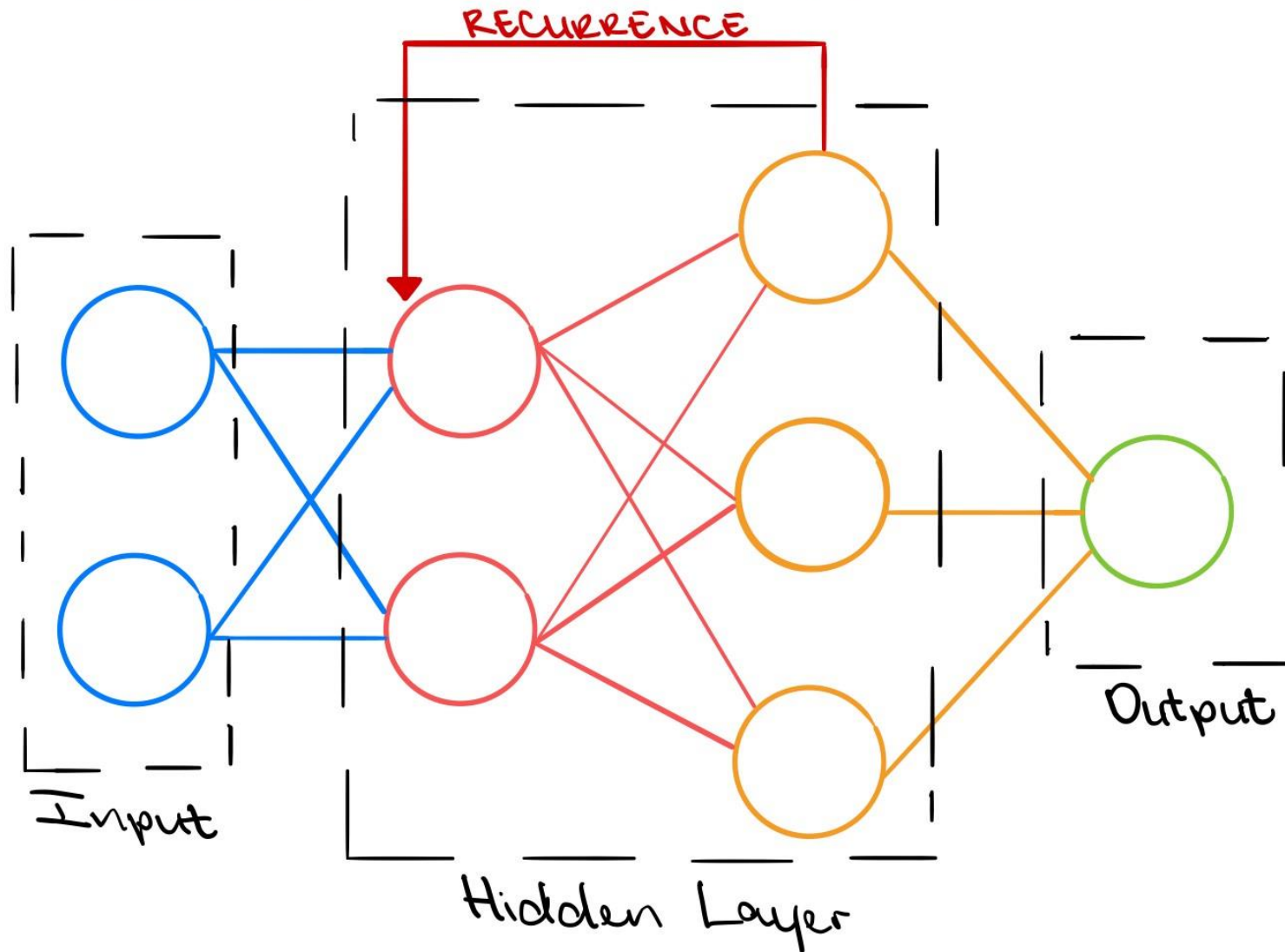
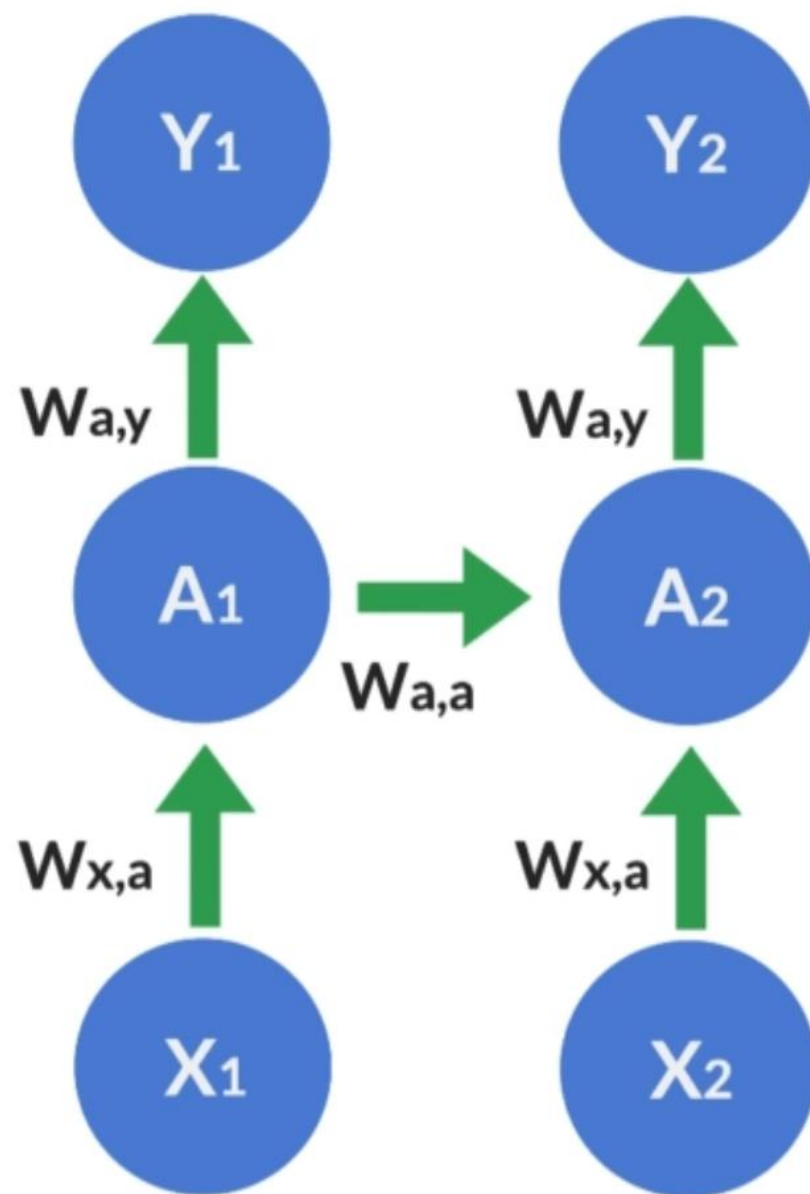
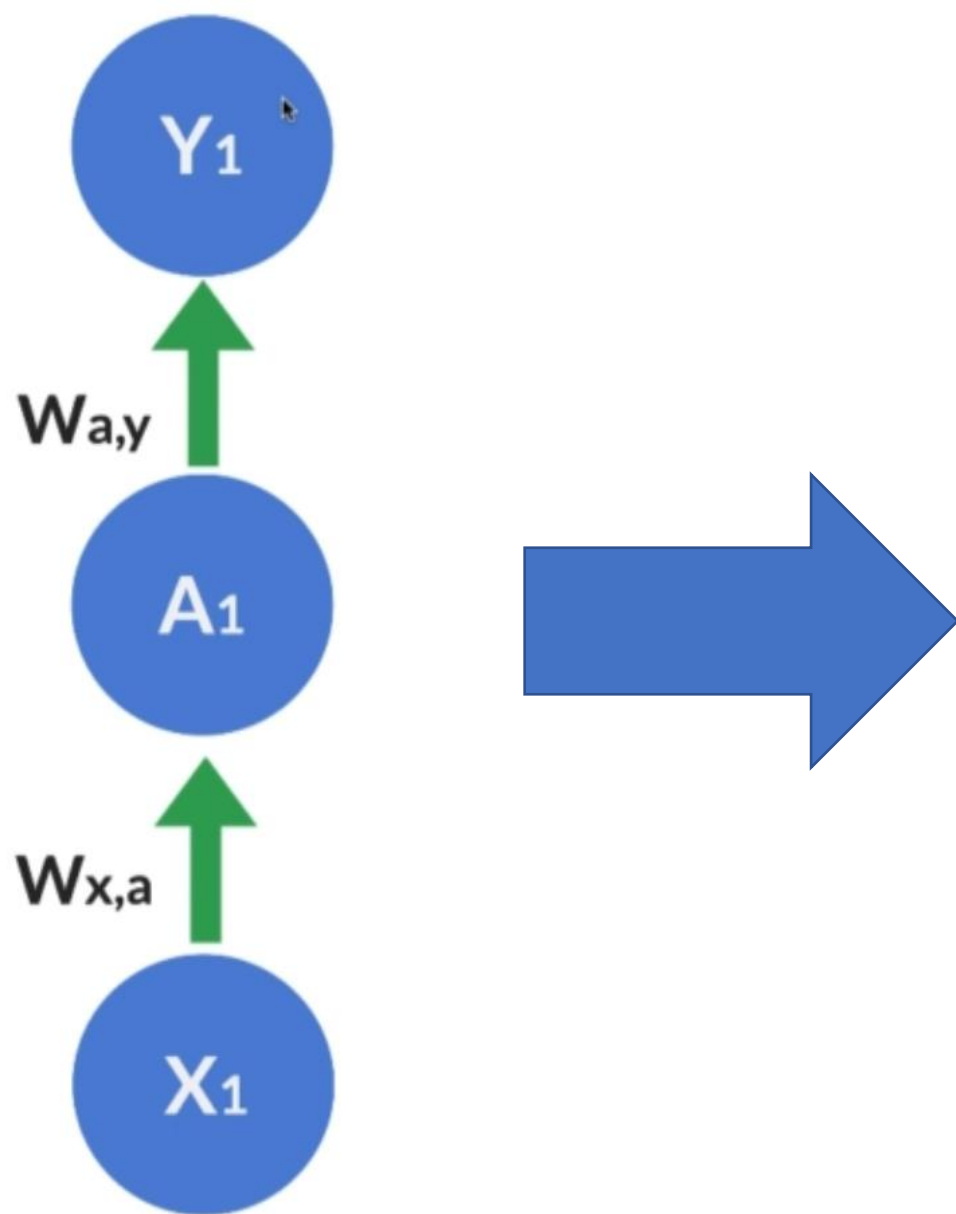
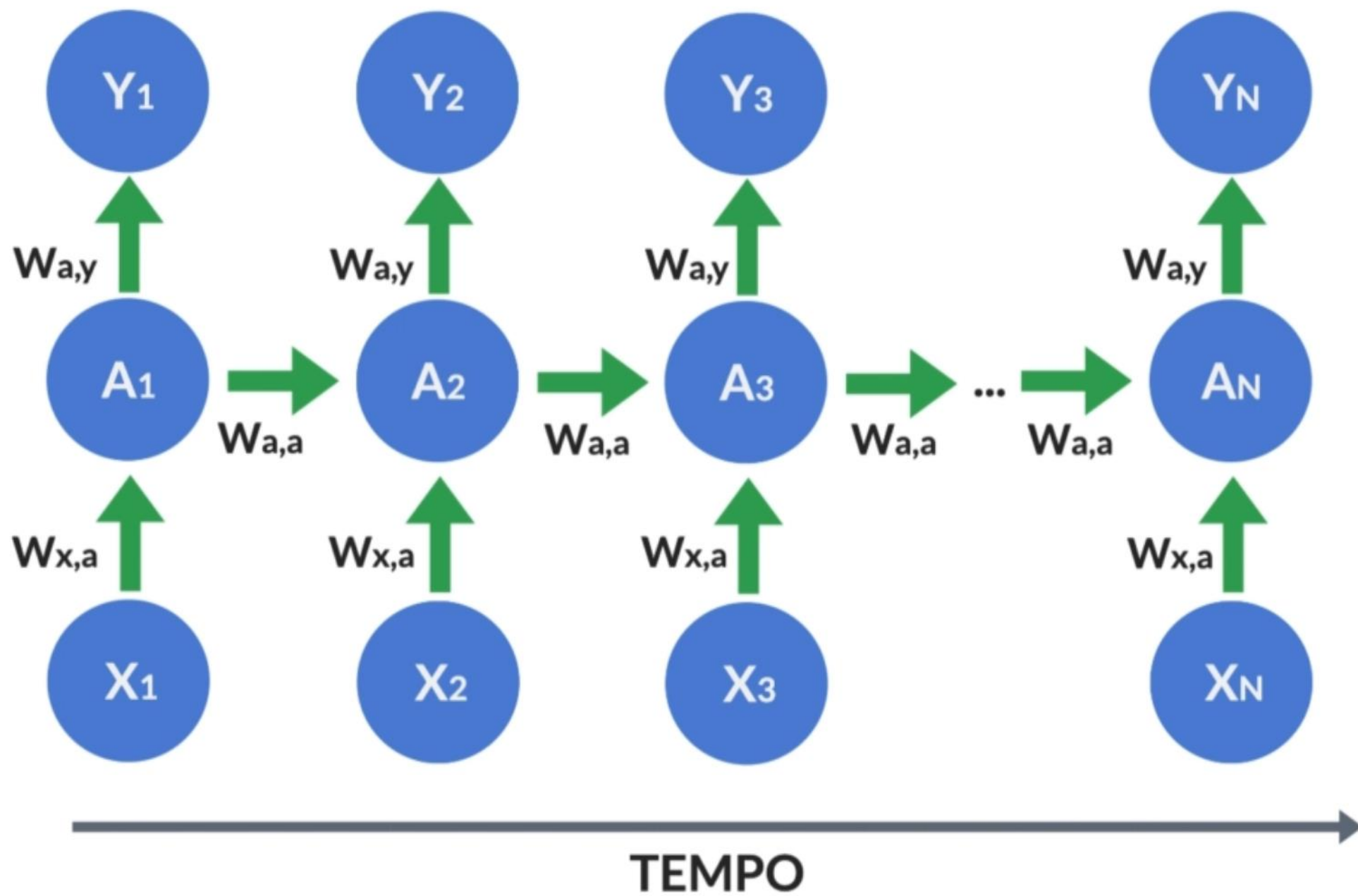
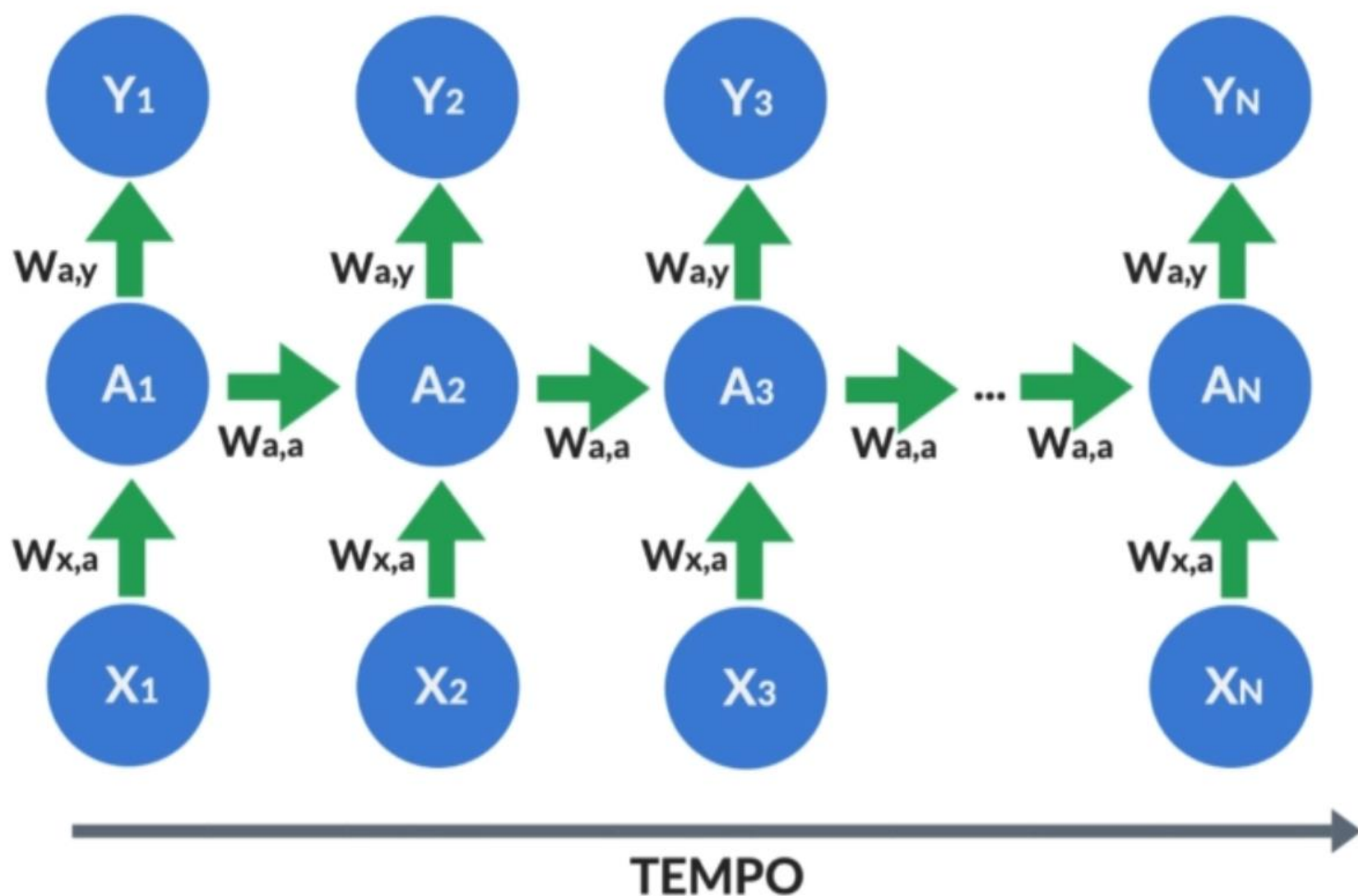


RECURRENT NEURAL NETWORKS









$$a_1 = \tanh(W_{x,a}x_1 + b_a)$$

$$y_1 = \text{softmax}(W_{a,y}a_1 + b_y)$$

$$a_2 = \tanh(W_{a,a}a_1 + W_{x,a}x_2 + b_a)$$

$$y_2 = \text{softmax}(W_{a,y}a_2 + b_y)$$

$$a_3 = \tanh(W_{a,a}a_2 + W_{x,a}x_3 + b_a)$$

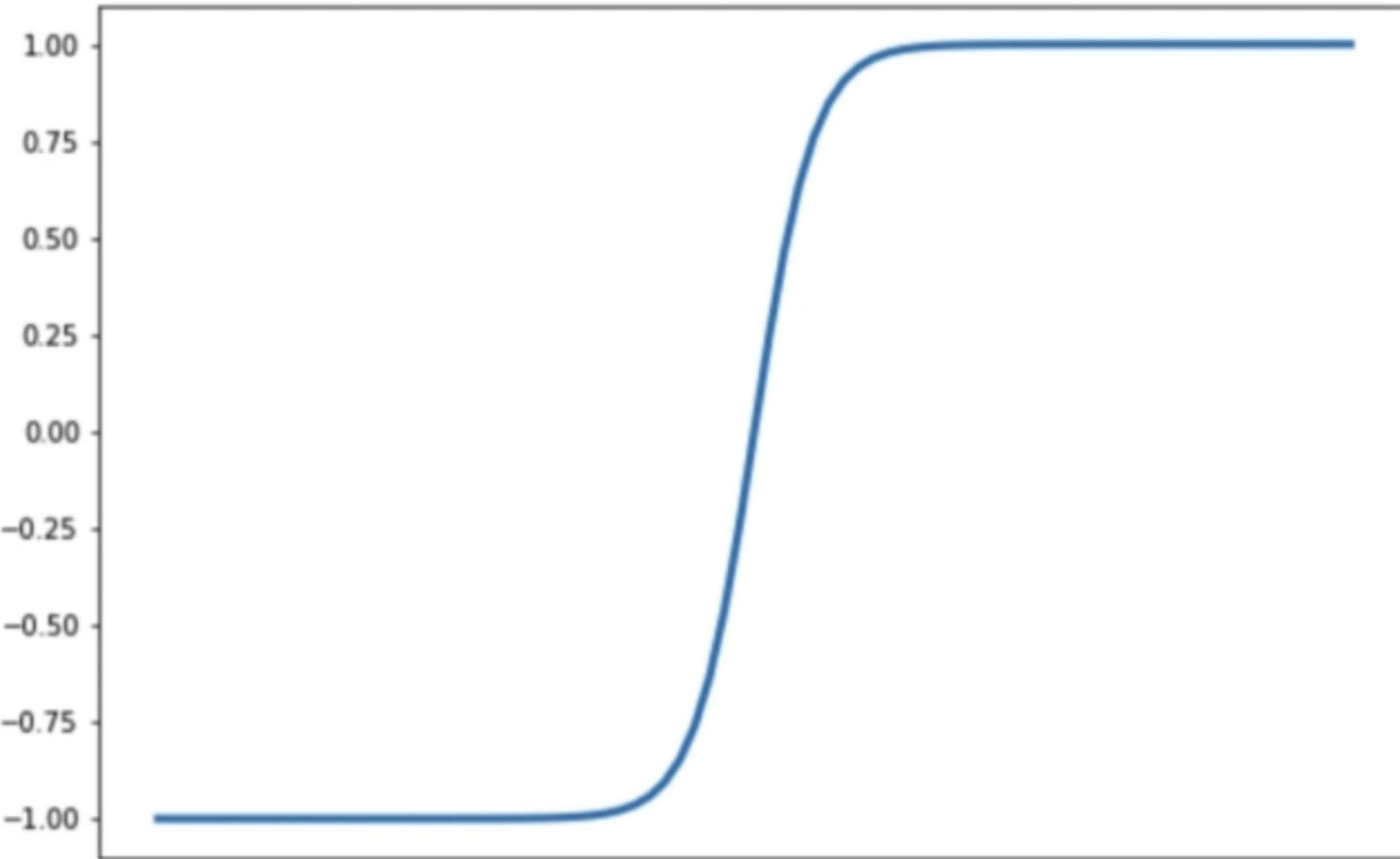
$$y_3 = \text{softmax}(W_{a,y}a_3 + b_y)$$

...

$$a_N = \tanh(W_{a,a}a_{N-1} + W_{x,a}x_N + b_a)$$

$$y_N = \text{softmax}(W_{a,y}a_N + b_y)$$

Tangente iperbolica



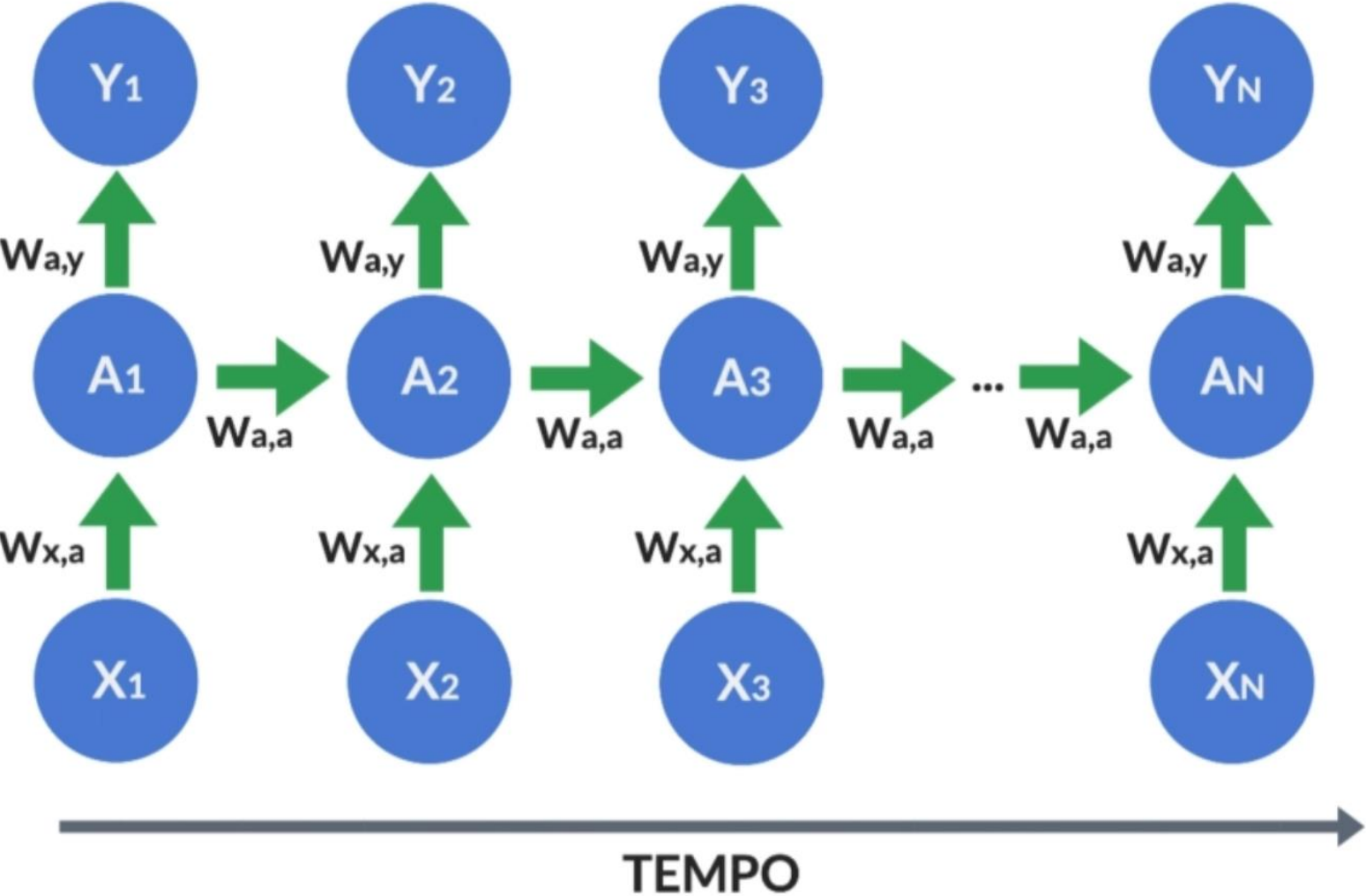
$$\phi(z) = \frac{1 - e^{-2z}}{1 + e^{-2z}}$$

Una rete ricorrente è esposta al problema di scomparsa/esplosione del gradiente tra le diverse esecuzioni della stessa rete
l'utilizzo della funzione di attivazione tangente iperbolica ci aiuta a limitare il problema (di più in seguito)

Relazione Uno a Uno



Relazione Molti a Molti



Relazione Molti a Molti

[Tutti](#) [Notizie](#) [Immagini](#) [Video](#) [Maps](#) [Altro](#) [Impostazioni](#) [Strumenti](#)

Circa 3.770.000.000 risultati (0,41 secondi)

Inglese



Italiano

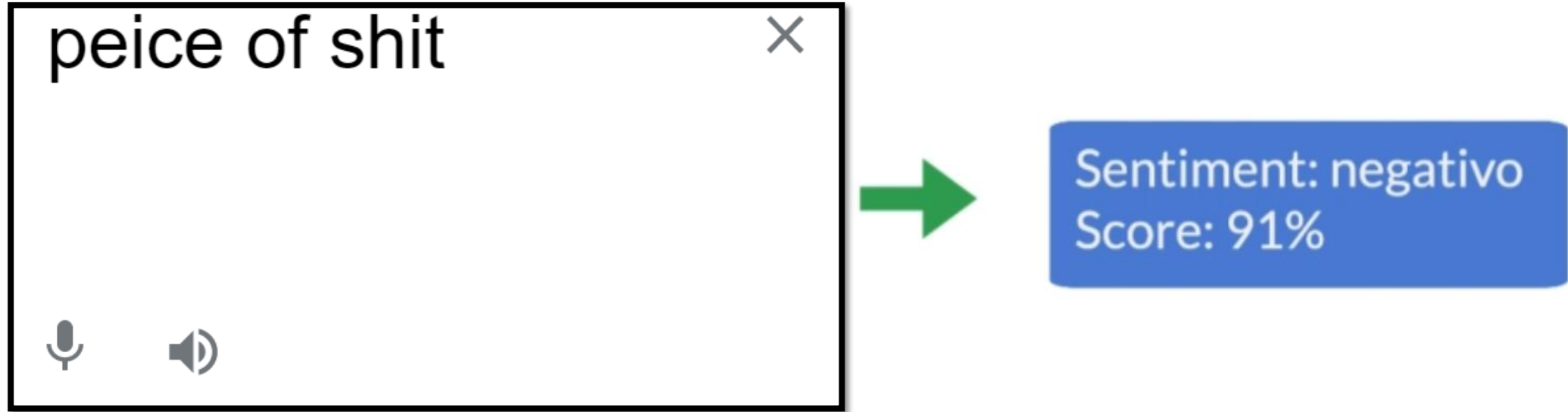
Hey Jude, don't
make it bad.
Take a sad song
and make it
better.
Remember to let
her into your
heart,
Then you can
start to make it
better.



Ehi Jude, non
metterlo male.
Prendi una canzone
triste e rendila
migliore.
Ricorda di lasciarla
entrare nel tuo cuore,
Quindi puoi iniziare a
renderlo migliore.



Relazione Molti a Uno



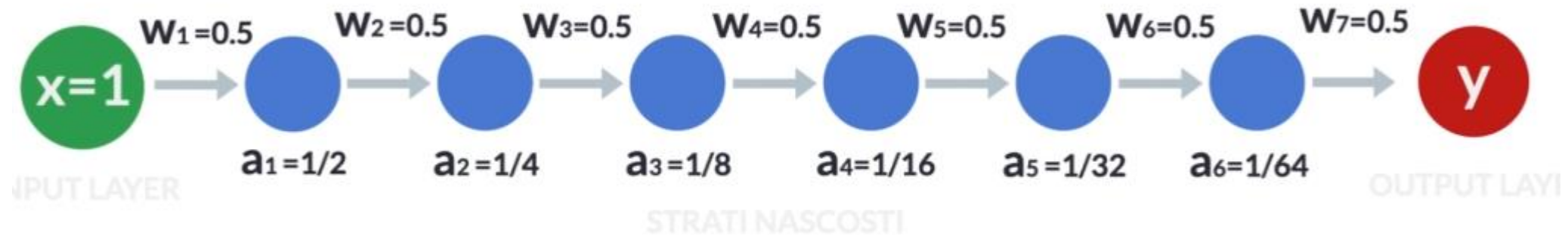
ESEMPIO
Sentiment Analysis

Backpropagation through time (BPTT)

Ci permette di addestrare una rete neurale ricorrente
propagando all'indietro il gradiente
non solo attraverso gli strati
ma anche attraverso le esecuzioni sequenziali della rete

Il problema del Vanishing Gradient nelle RNN

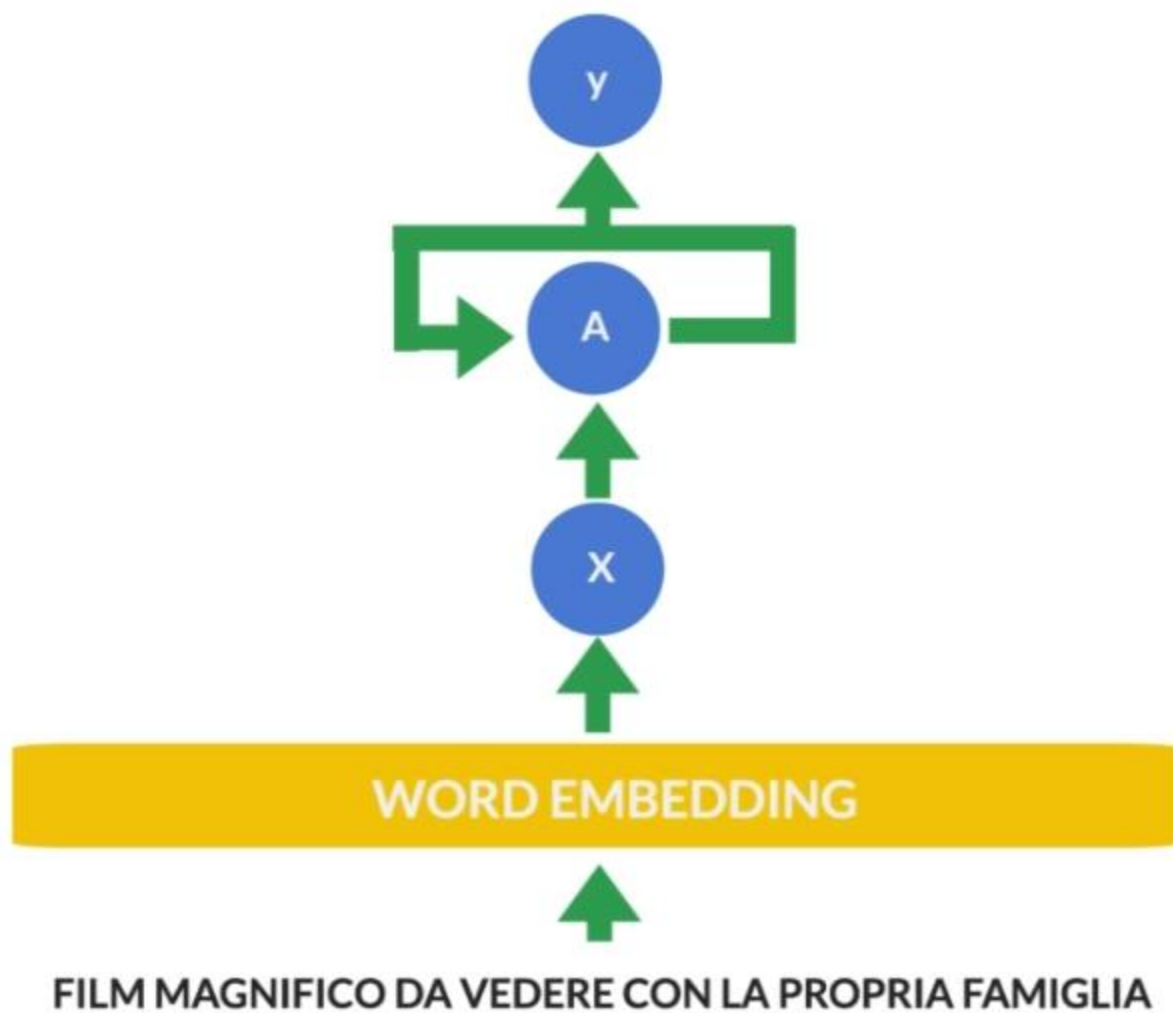
Il problema della scomparsa del Gradiente



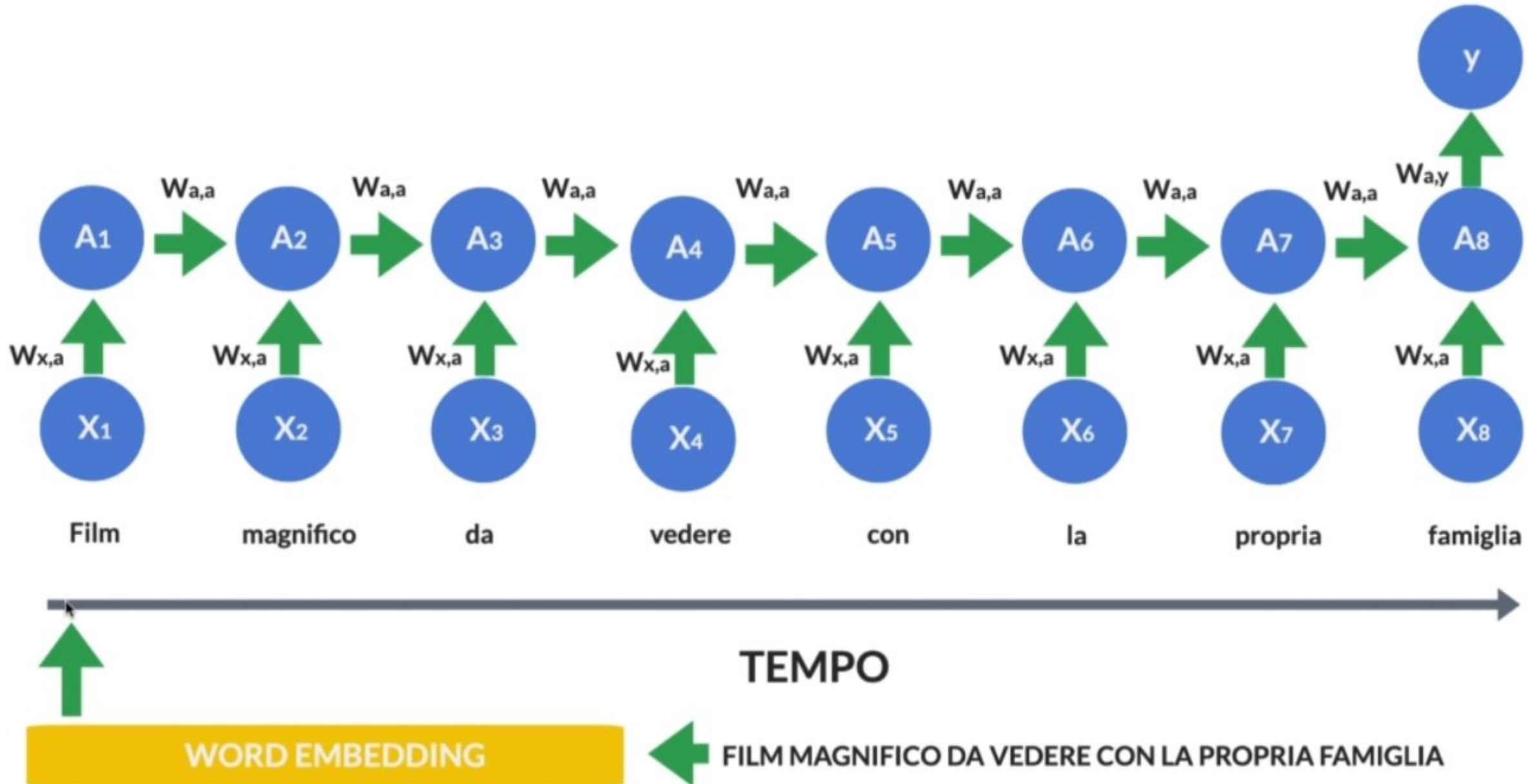
BACKWARD PROPAGATION

Effettuando moltiplicazioni ripetute per valori che tendono allo 0 il gradiente scomparirà (0).

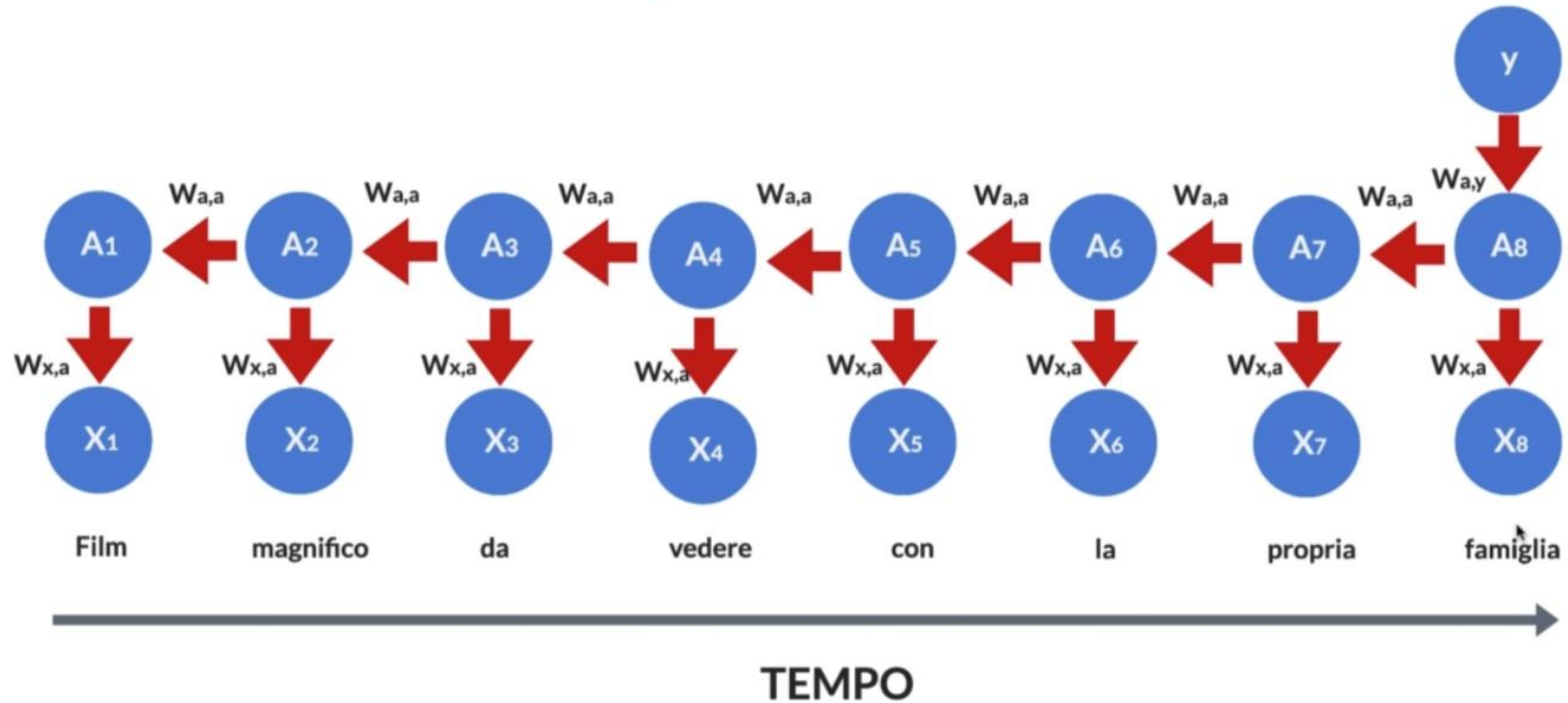
Vanishing Gradient e RNN



Vanishing Gradient e RNN



Vanishing Gradient e RNN



BACKPROPAGATION THROUGH TIME

L'errore viene propagato non solo attraverso gli strati ma anche attraverso le esecuzioni



Le reti ricorrenti semplici (Vanilla RNN) vanno bene per brevi sequenze.

Ma a causa del problema della scomparsa del gradiente sono inadatte per sequenze più lunghe.

Understanding LSTM Networks

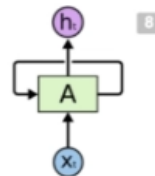
Posted on August 27, 2015

Recurrent Neural Networks

Humans don't start their thinking from scratch every second. As you read this essay, you understand each word based on your understanding of previous words. You don't throw everything away and start thinking from scratch again. Your thoughts have persistence.

Traditional neural networks can't do this, and it seems like a major shortcoming. For example, ⁵ imagine you want to classify what kind of event is happening at every point in a movie. It's unclear how a traditional neural network could use its reasoning about previous events in the film to inform later ones.

Recurrent neural networks address this issue. They are networks with loops in them, allowing ¹ information to persist.



Recurrent Neural Networks have loops.

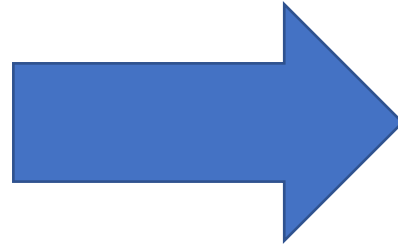
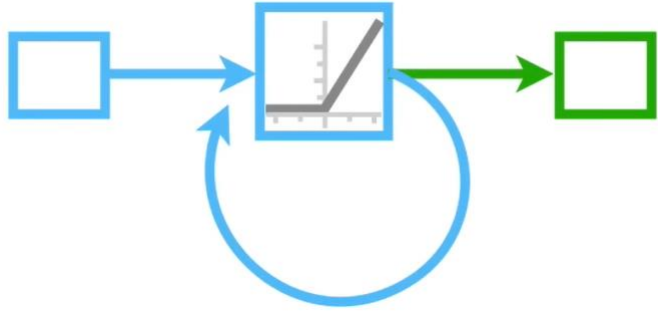
<http://colah.github.io/posts/2015-08-Understanding-LSTMs/>

Cosa fa una LSTM ?

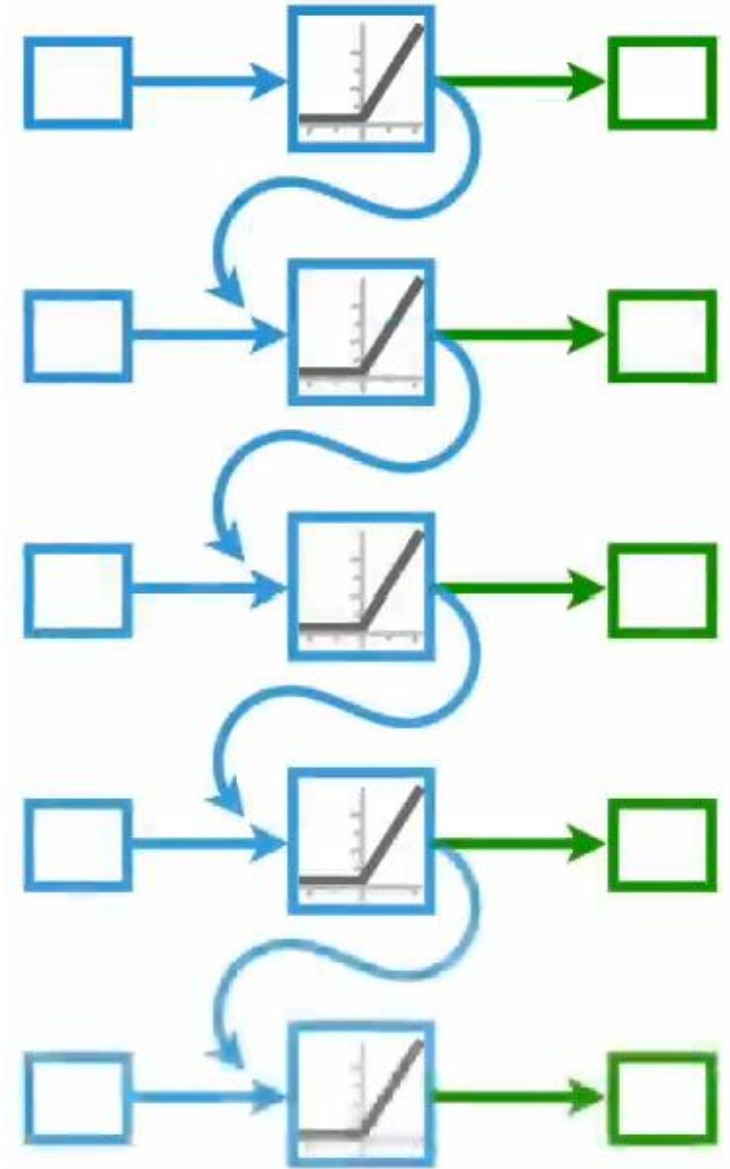
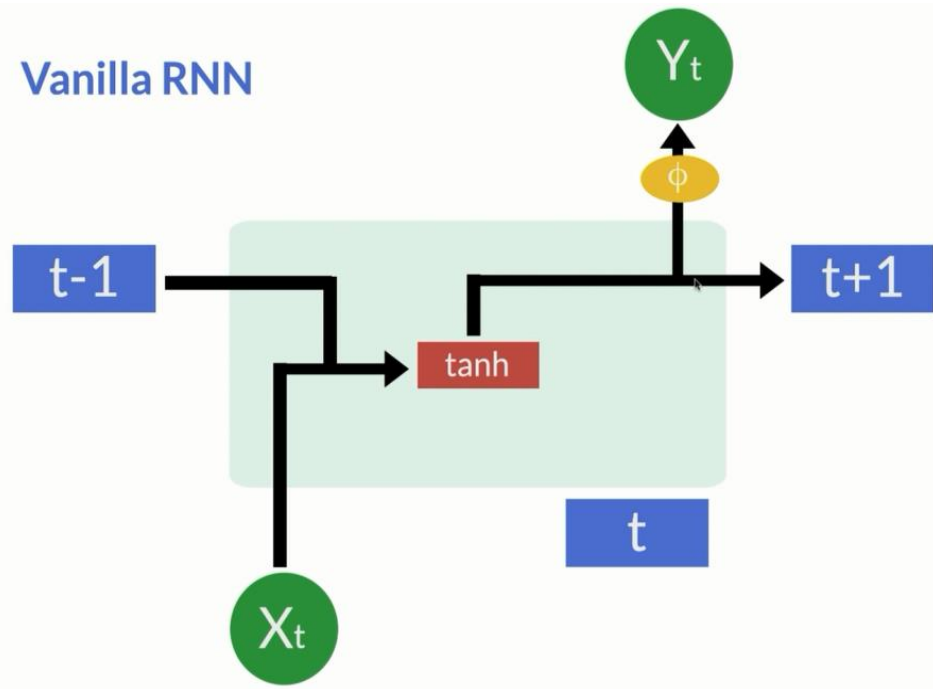
Un capolavoro unico ! Ho portato la mia ragazza al cinema per il suo compleanno, non avevo grosse aspettative ma devo ammettere che ho amato questo film fino all'ultima, anche se purtroppo mi sono perso i 5 minuti iniziali. Lo consiglio a tutti.

Una LSTM memorizza le informazioni importanti e scarta tutto il resto

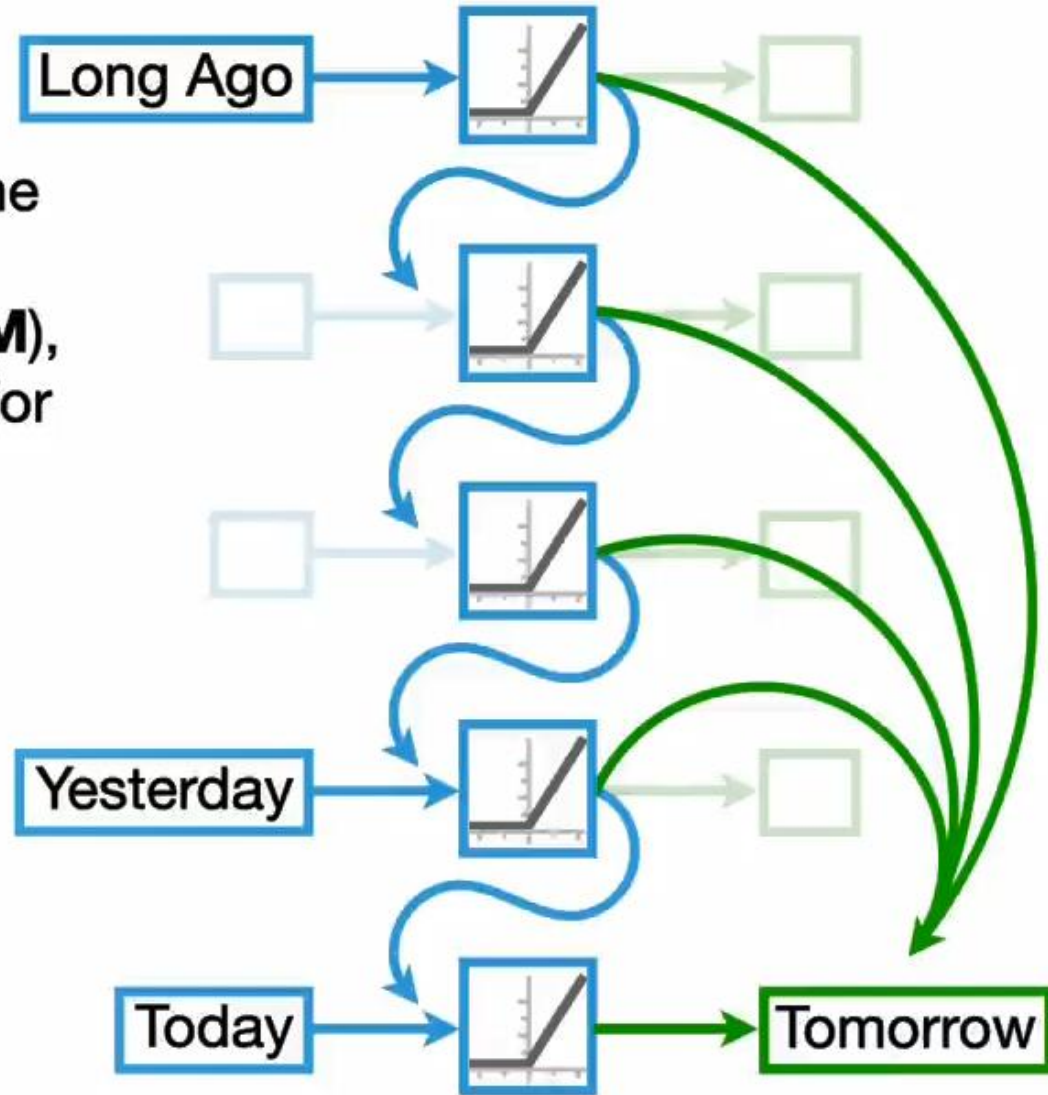
RNN

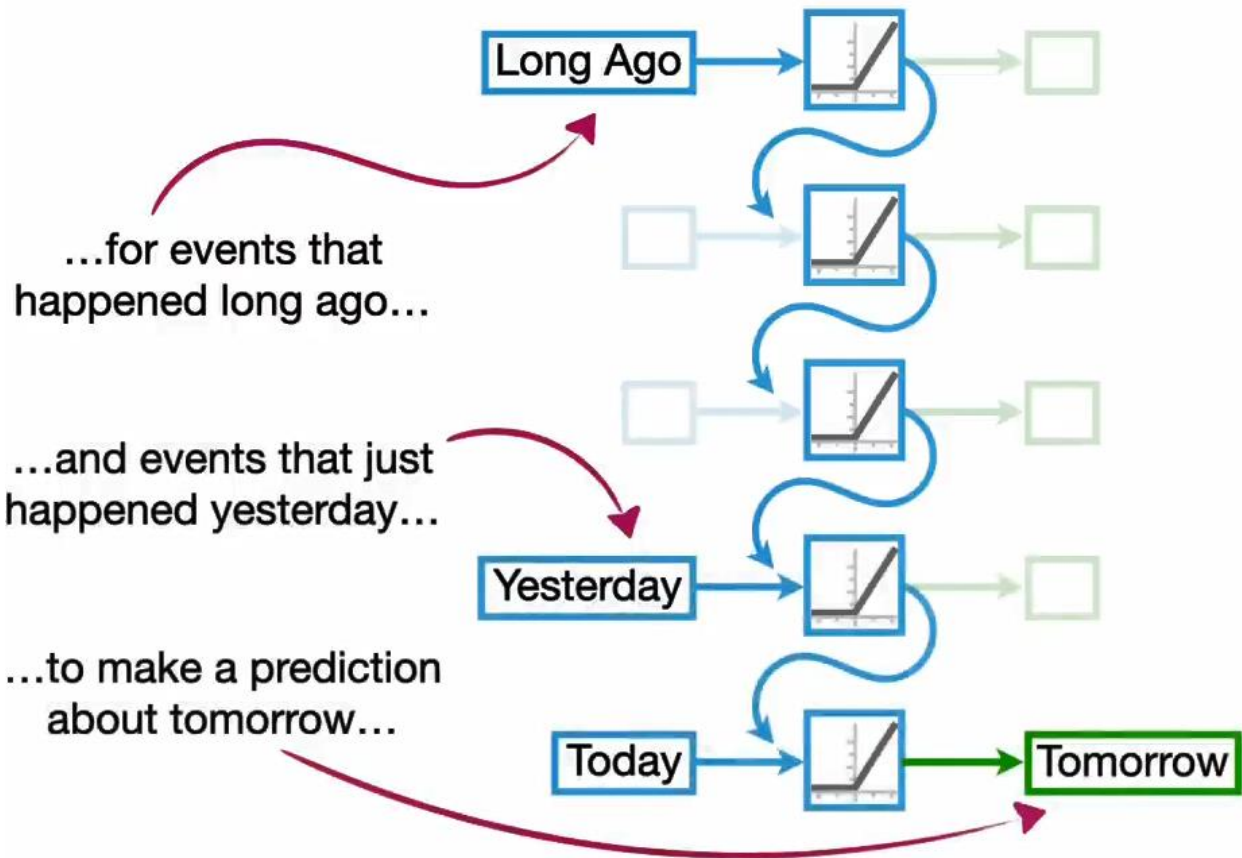


Vanilla RNN

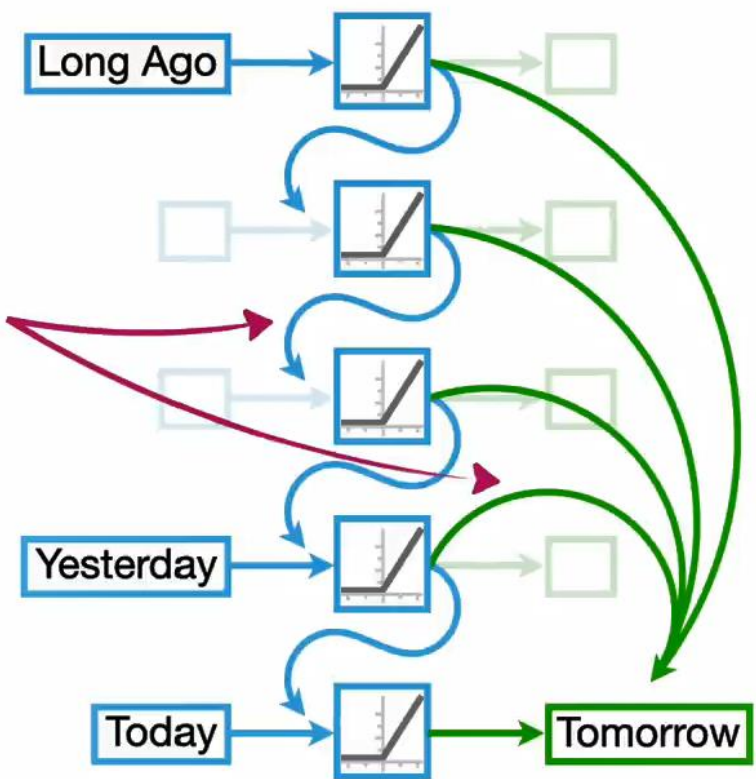


Now that we understand the **main idea** behind **Long Short-Term Memory (LSTM)**, that it uses different paths for **Long** and **Short-Term Memories**...

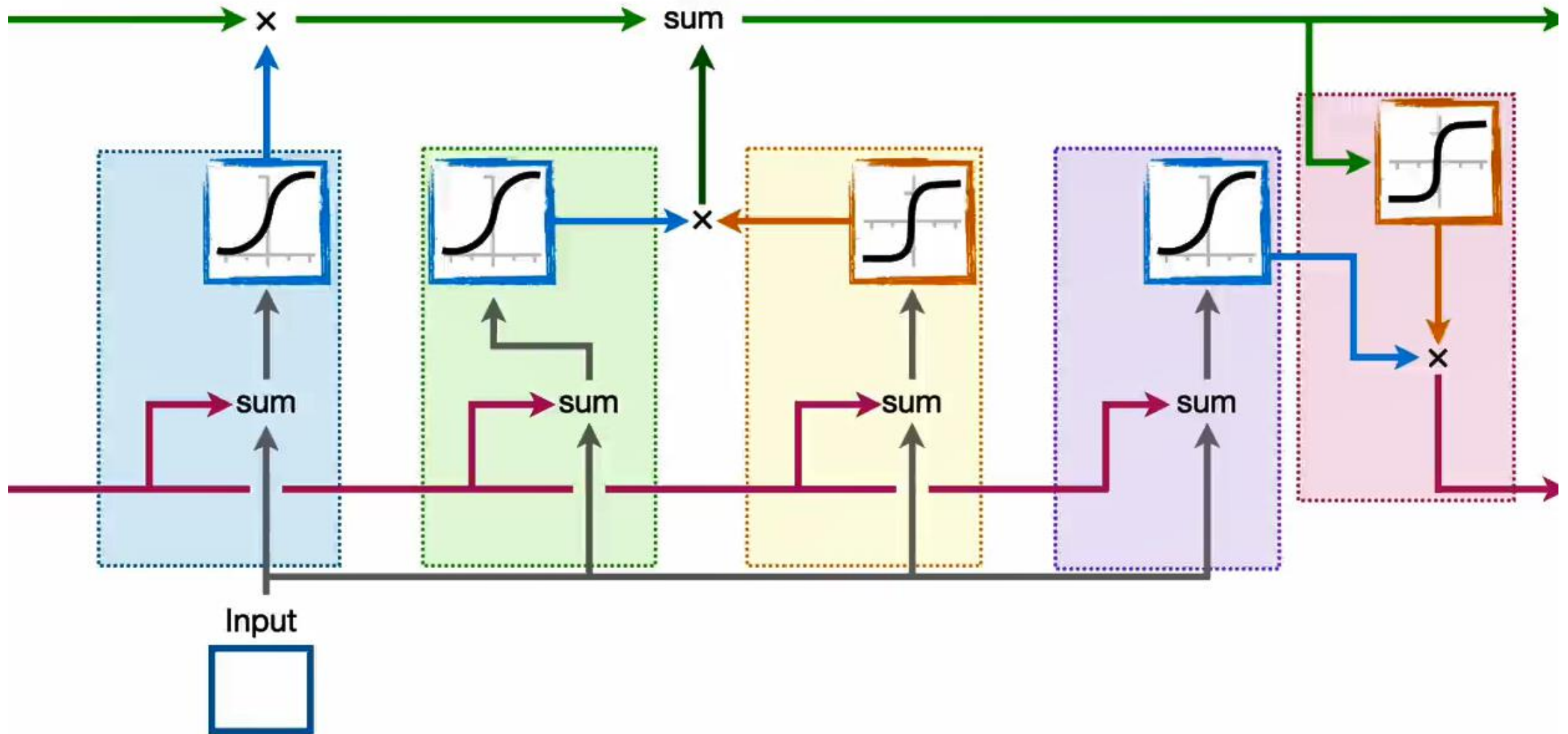




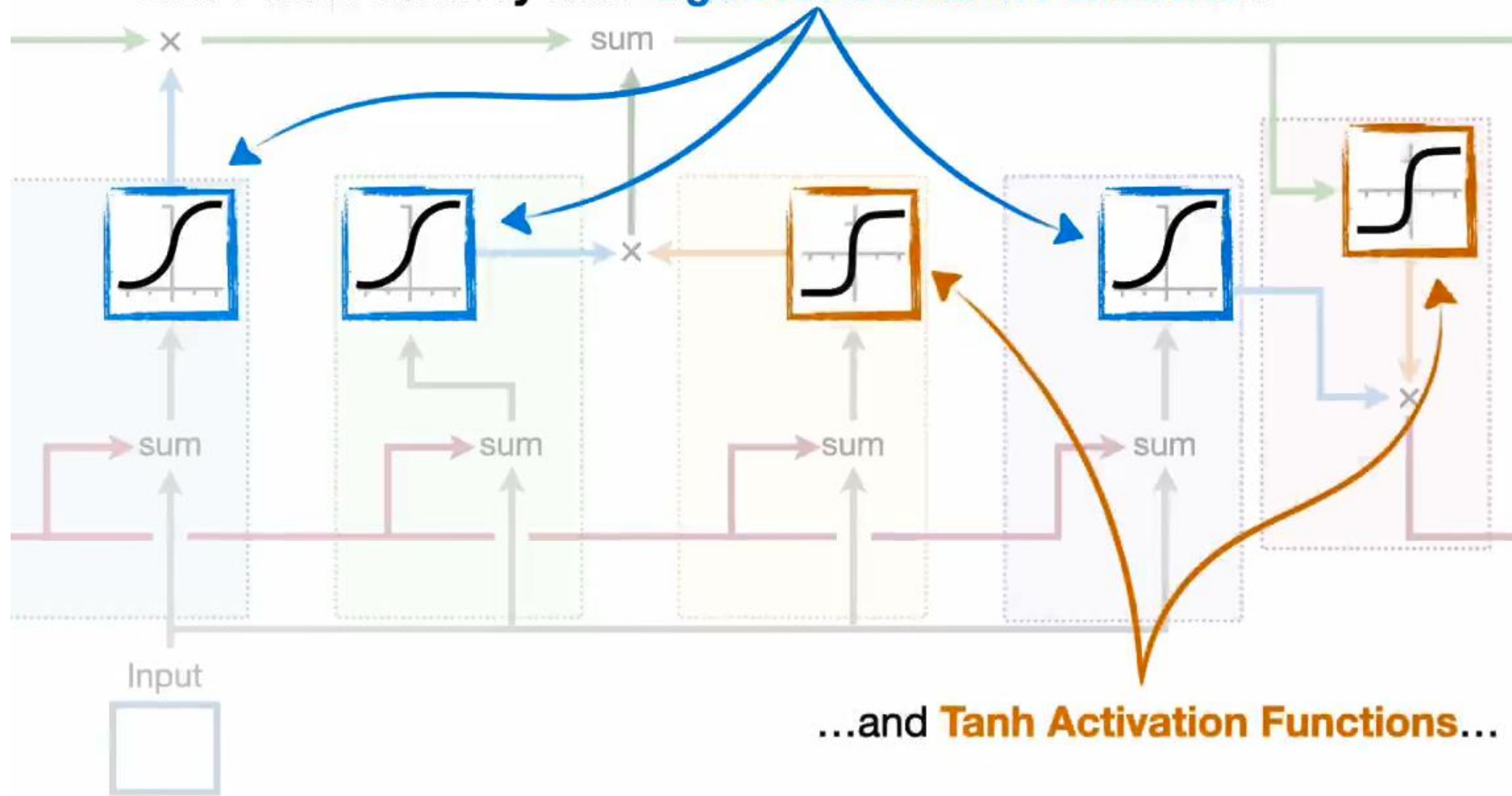
...Long Short-Term
Memory uses two
separate paths to make
predictions about
tomorrow.



...**Long Short-Term Memory** is based
on a much more complicated unit.



NOTE: Unlike the networks we've used before in this series, **Long Short-Term Memory** uses **Sigmoid Activation Functions**...



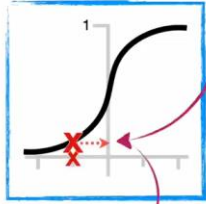
...and **Tanh Activation Functions**...

...then we get **0.01** as the y-axis coordinate.

$$f(x) = \frac{e^x}{e^x + 1}$$

$$f(-5) = \frac{e^{-5}}{e^{-5} + 1}$$

$$= 0.01$$

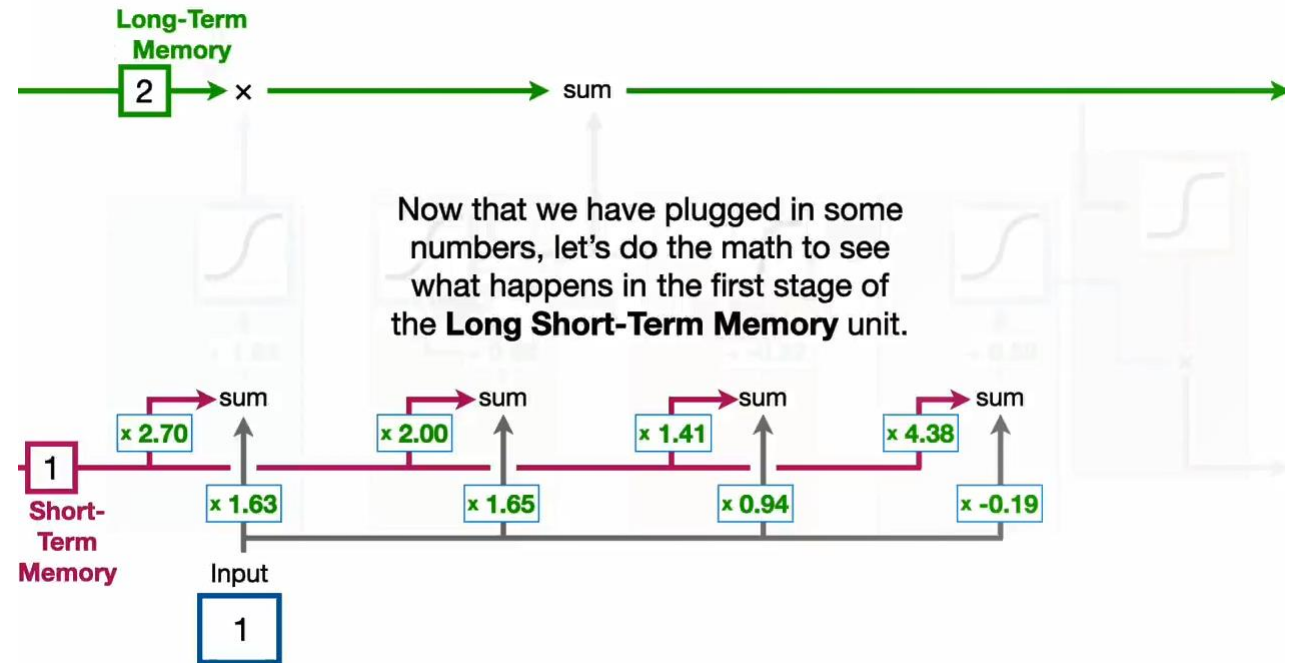
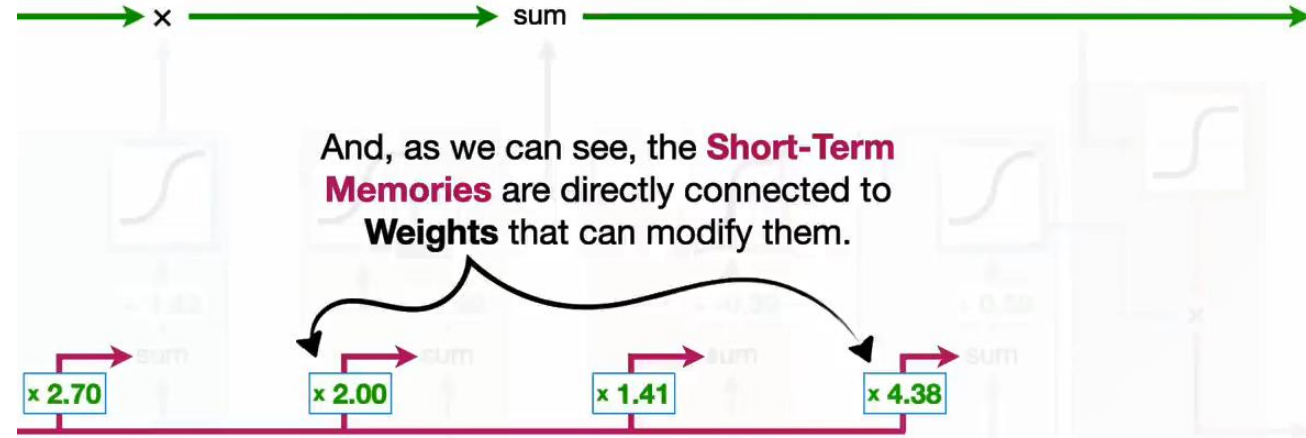
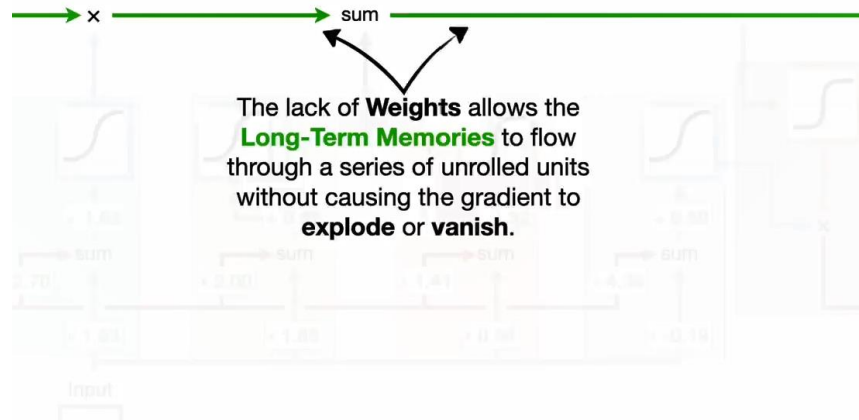
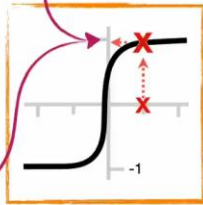


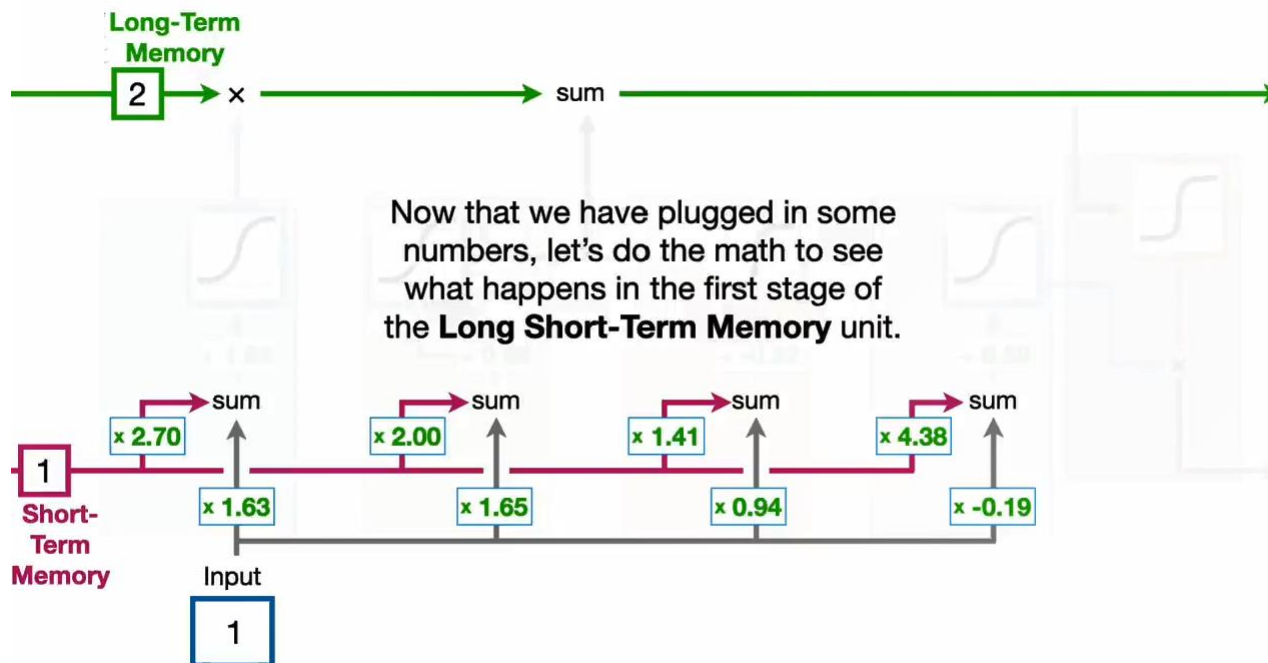
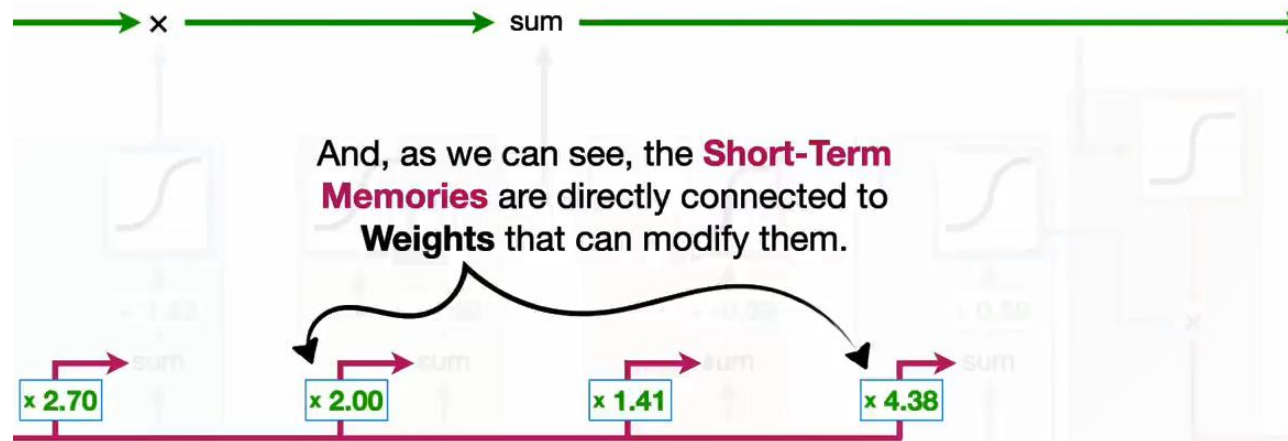
...we get **0.96** as the y-axis coordinate.

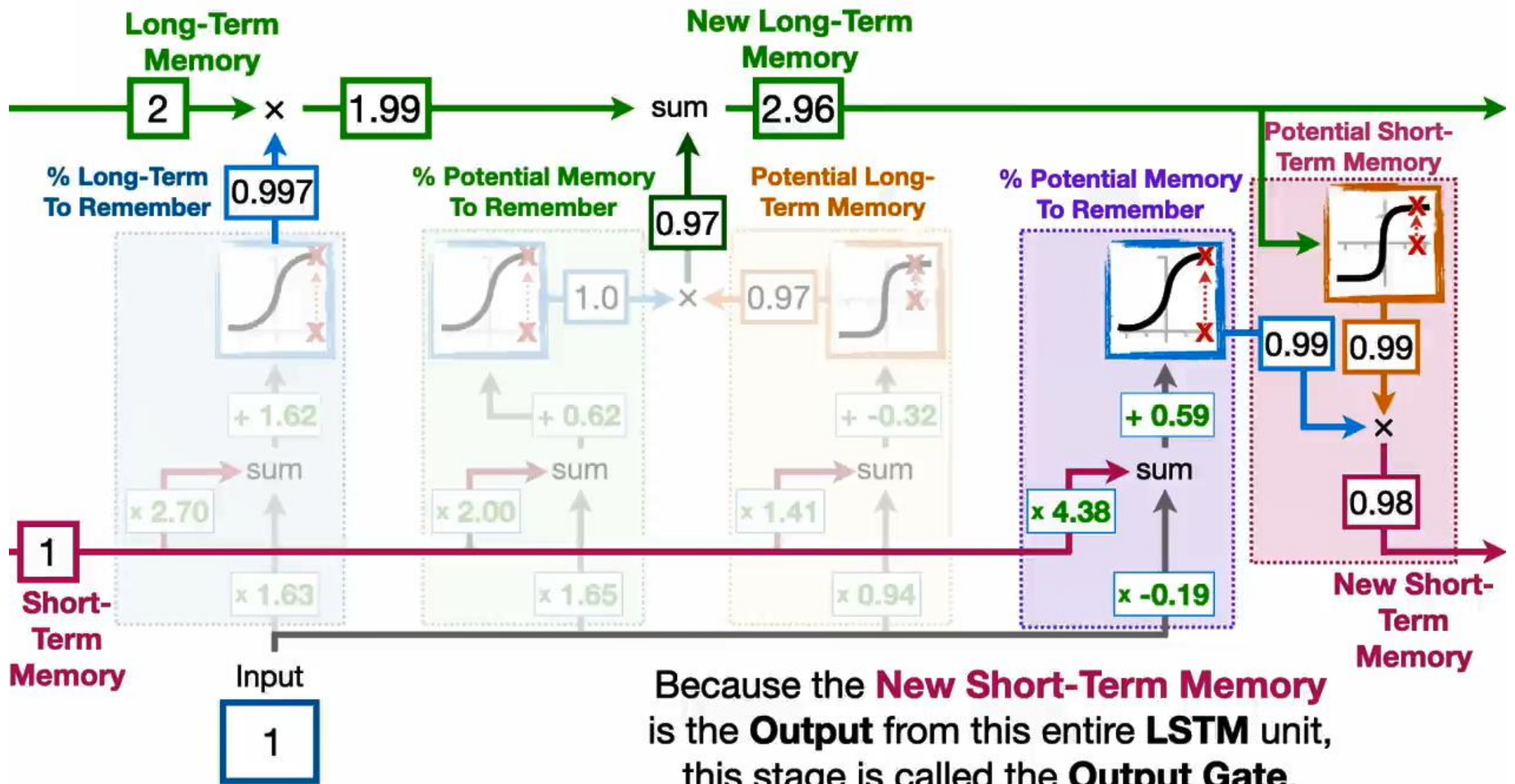
$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

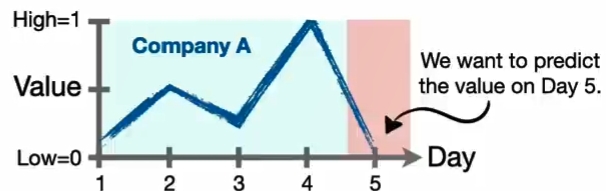
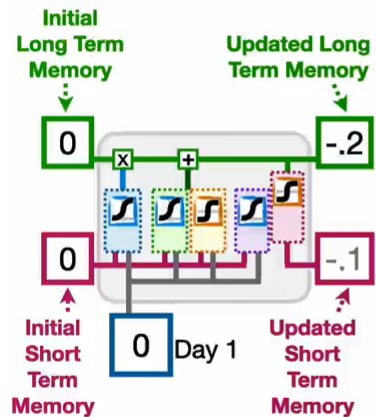
$$f(2) = \frac{e^2 - e^{-2}}{e^2 + e^{-2}}$$

$$= 0.96$$

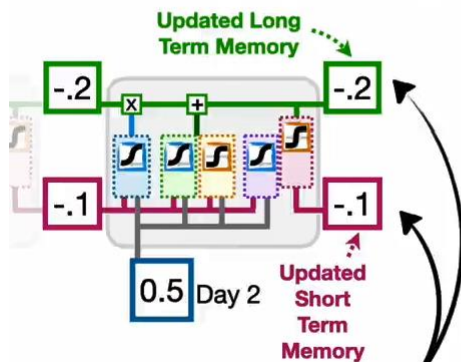
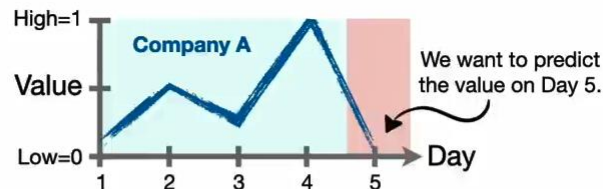




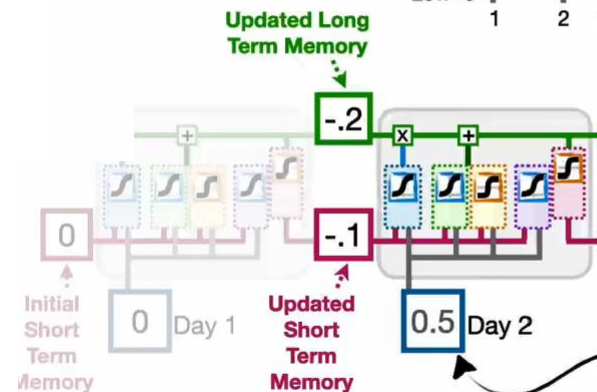




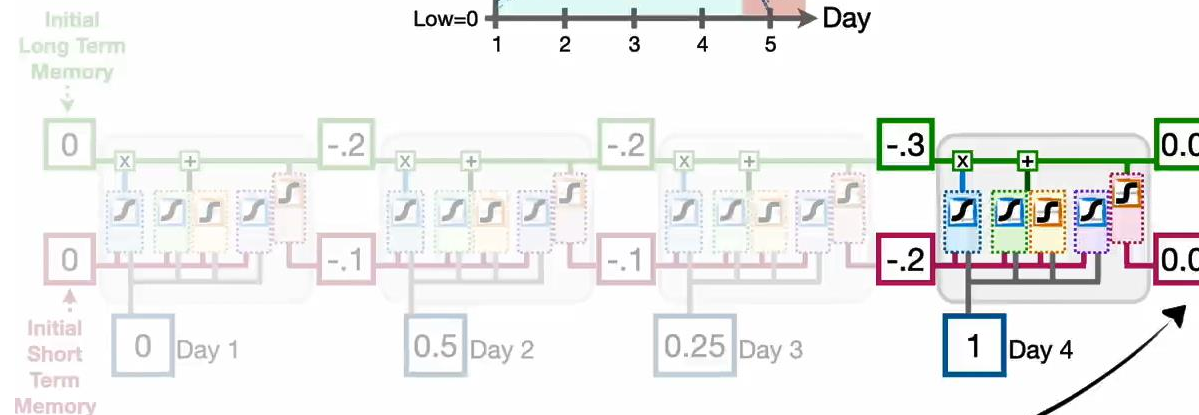
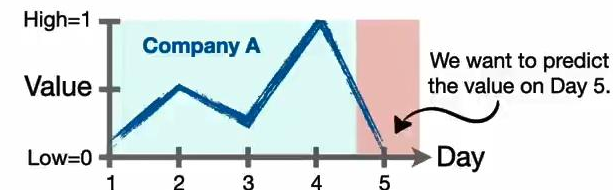
...and **-0.1** (rounded) for the updated **Short-Term Memory**.



...and we end up with these updated **Long and Short-Term Memories**.

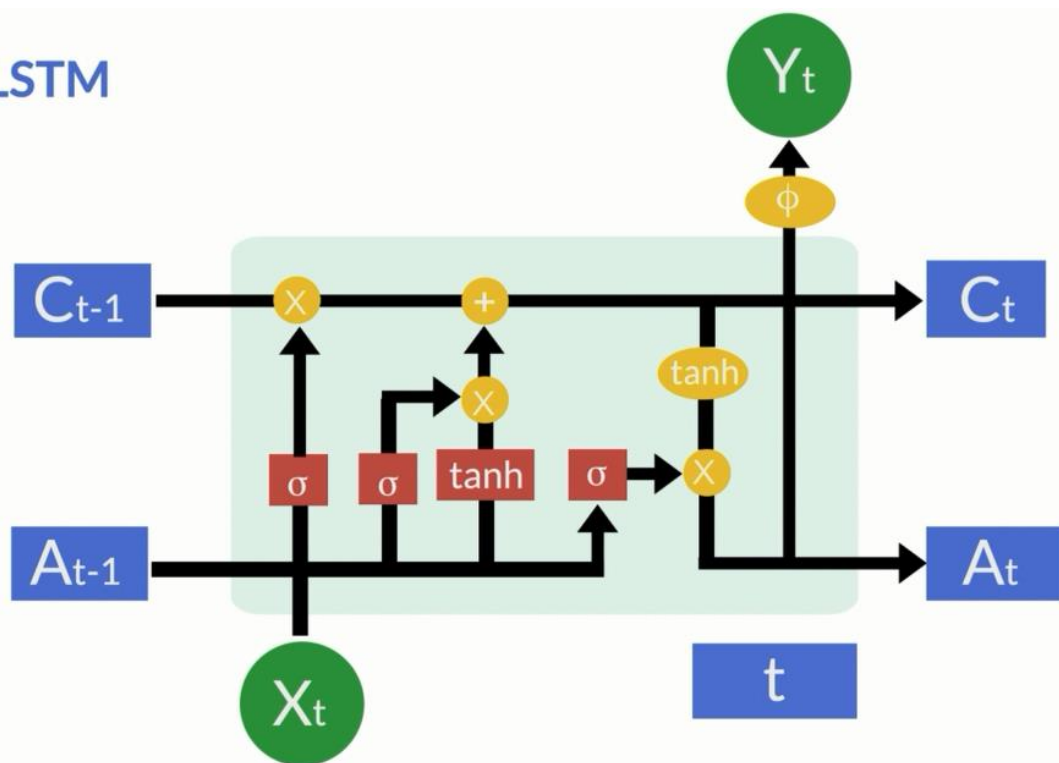


...and plug the value from **Day 2, 0.5**, into the **Input**.



And the final **Short-Term Memory, 0.0**, is the **Output** from the unrolled **LSTM**.

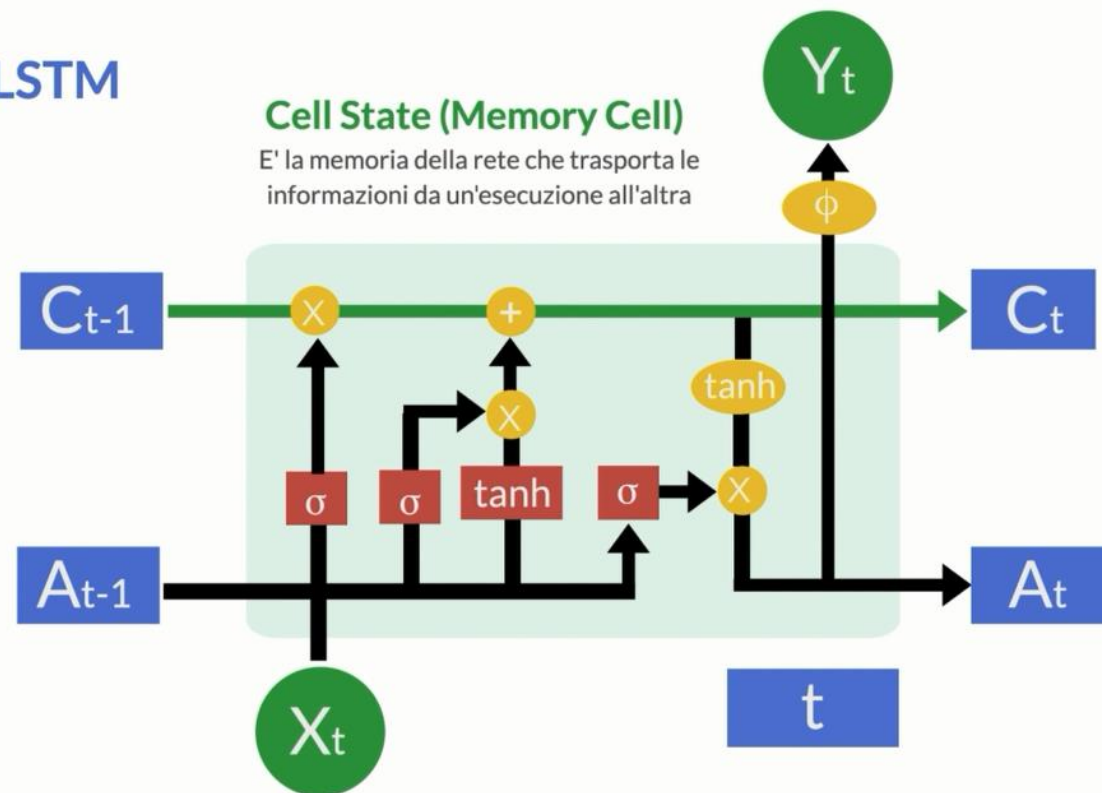
LSTM



LSTM

Cell State (Memory Cell)

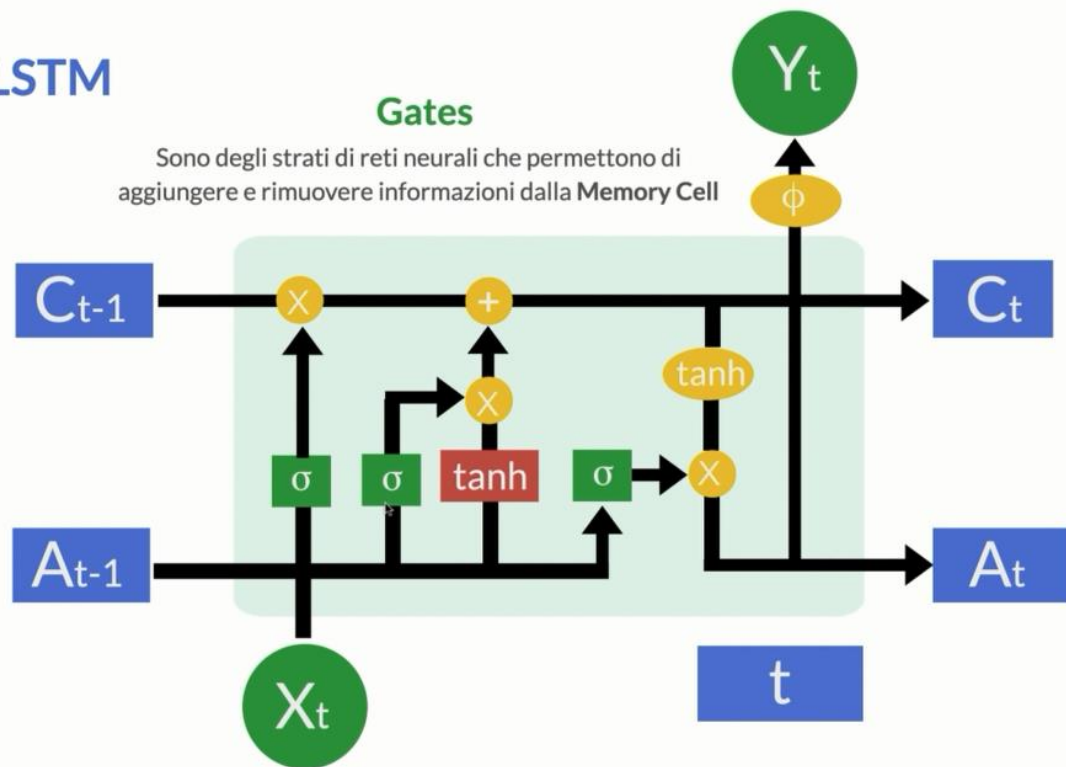
E' la memoria della rete che trasporta le informazioni da un'esecuzione all'altra



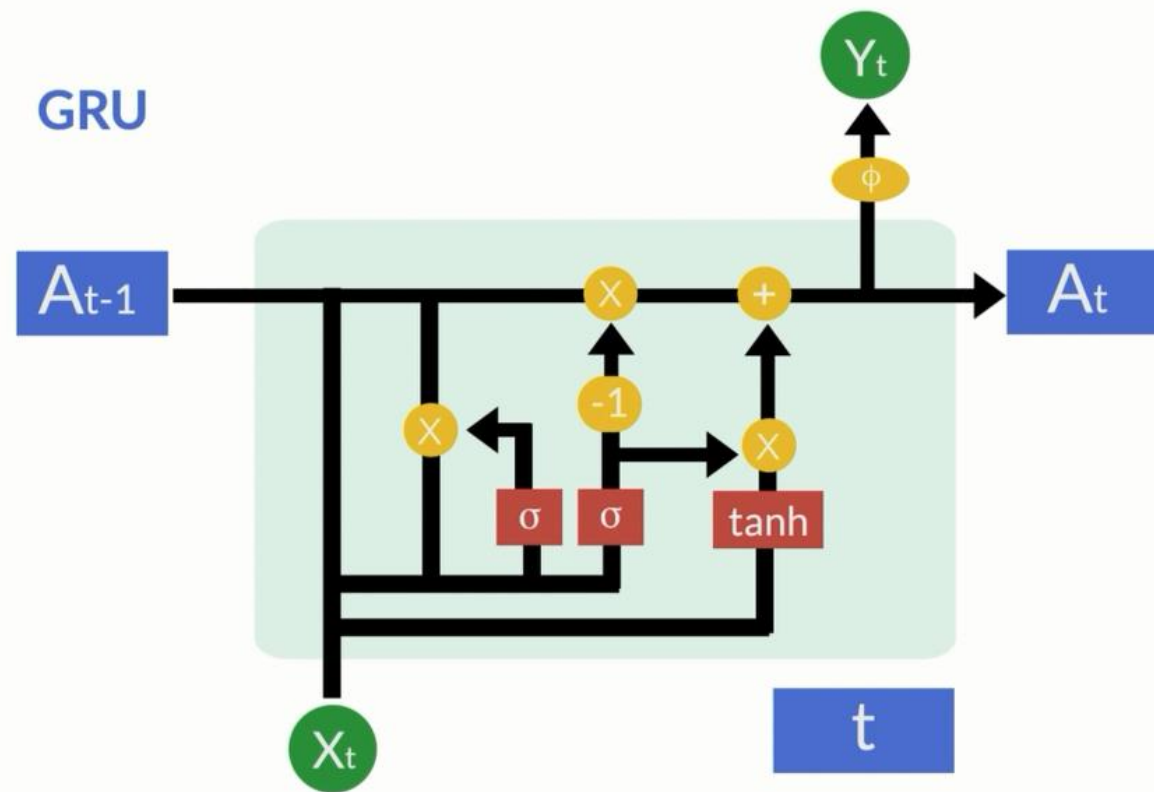
LSTM

Gates

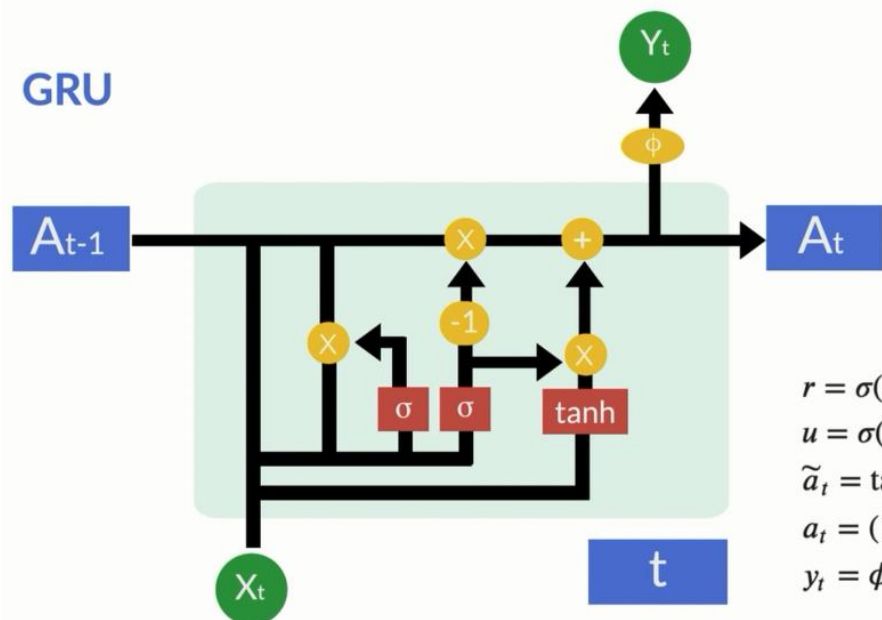
Sono degli strati di reti neurali che permettono di aggiungere e rimuovere informazioni dalla Memory Cell



GRU



GRU



RIEPILOGO

$$\begin{aligned}
 r &= \sigma(W_r \cdot [a_{t-1}, x_t] + b_r) \\
 u &= \sigma(W_u \cdot [a_{t-1}, x_t] + b_u) \\
 \tilde{a}_t &= \tanh(W_a \cdot [r * a_{t-1}, x_t] + b_a) \\
 a_t &= (1 - u) * a_{t-1} + u * \tilde{a}_t \\
 y_t &= \phi(a_t)
 \end{aligned}$$

GRU

Richiedono meno calcoli tra tensori, quindi l'addestramento richiede meno tempo.

vs

LSTM

Solitamente portano a risultati migliori e ricordano sequenze più lunghe.