Metody Inżynierii Wiedzy

Recurrent Neural Networks – Long Short-Term Memory Networks – Transformers

RNN – LSTM – 

Dr inż. Michał Majewski

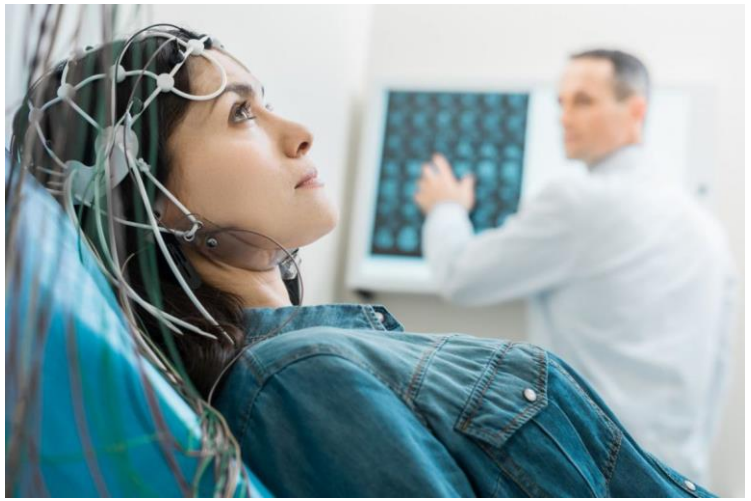
mmajew@pjwstk.edu.pl

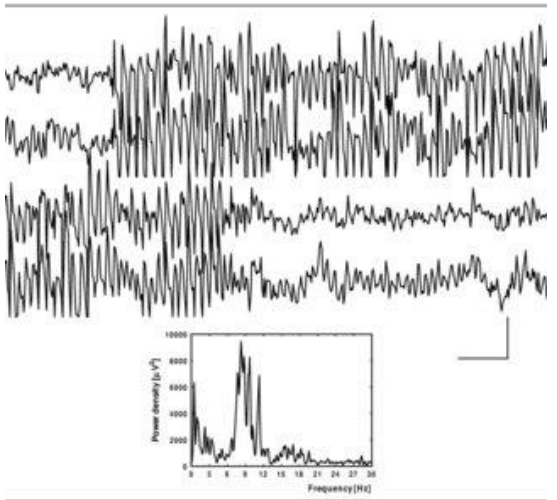
materiały: *ftp*(*public*) : *//mmajew/MIW*

Przetwarzanie sekwencji danych

1. Przetwarzanie języka naturalnego (NLP) :
   - Tłumaczenie maszynowe
   - Analiza sentymentu
   - Generowanie tekstu
2. Przewidywanie szeregów czasowych:
   - Prognozowanie cen akcji
   - Przewidywanie pogody
   - Analiza trendów sprzedaży
3. Rozpoznawanie mowy:
   - Konwersja mowy na tekst
   - Transkrypcja nagrań
   - Asystenci głosowi

4. Przetwarzanie sygnałów:
   - Analiza sygnałów EEG/ECG (zapisy aktywności elektrycznej mózgu/ elektryczną aktywność serca)
   - Rozpoznawanie wzorców w danych sensorycznych
   - Kompresja i rekonstrukcja sygnałów
5. Przetwarzanie wideo:
   - Analiza ruchu
   - Opis i kategoryzacja klipów wideo
   - Rozpoznawanie aktywności wideo

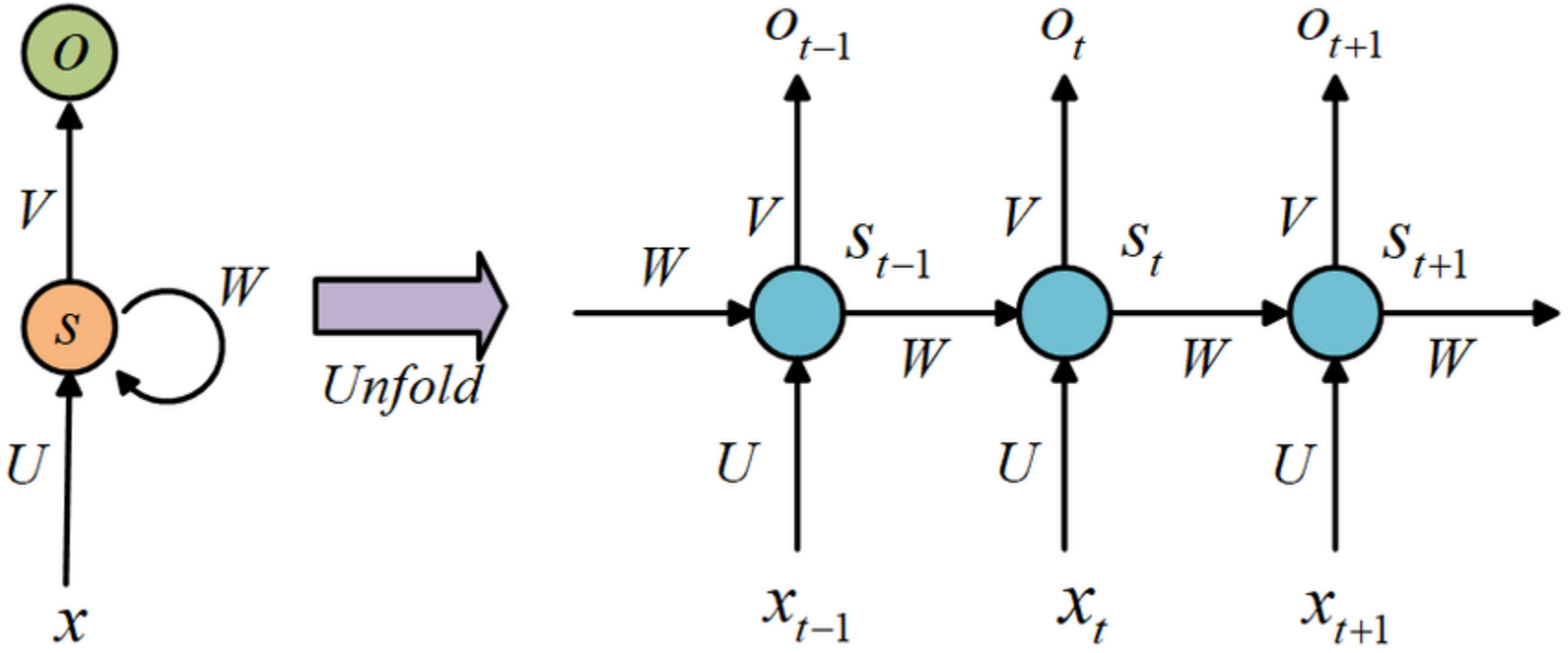




https://www.researchgate.net/profile/Dragan-Djuric/publication/5657047/figure/fig6/AS:667608630173700@1536181709329/Representative-EEG-correlate-and-the-corresponding-power-spectra-bottom-diagram-of_Q320.jpg

Przetwarzanie sekwencji danych

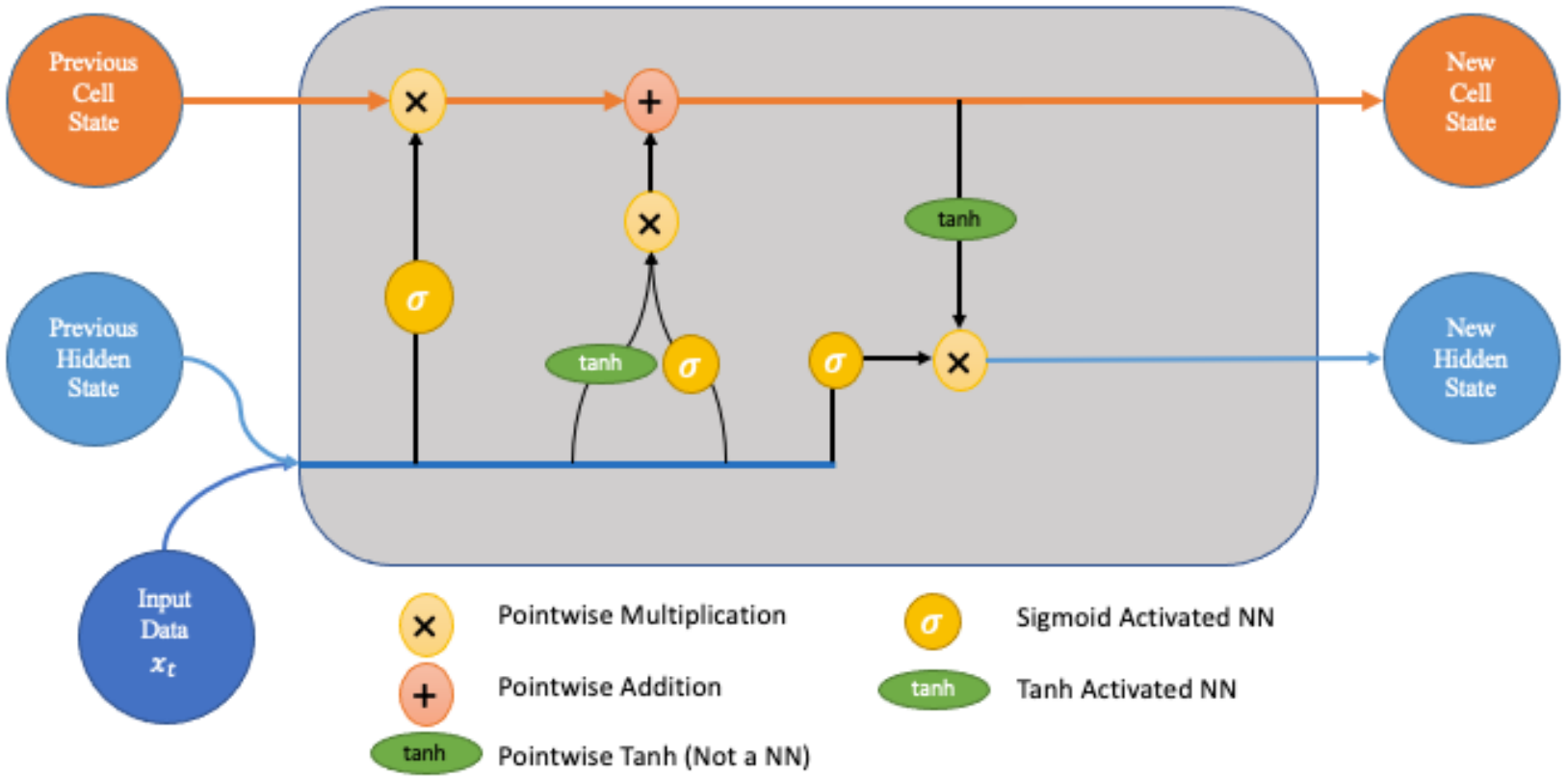
- **Rekurencyjne Sieci Neuronowe (RNN)** są typem sieci neuronowych zaprojektowanych do przetwarzania sekwencji danych, gdzie **wyjście z jednej iteracji jest używane jako wejście do następnej**, co pozwala im na modelowanie zależności czasowych.



https://www.researchgate.net/publication/334464982/figure/fig1/AS:780930788626432@1563199817147/Basic-RNN-structure.ppm

Przetwarzanie sekwencji danych

- **Długoterminowe Pamięci Krótkotrwałe (LSTM)** to ulepszona wersja RNN, która wprowadza specjalne mechanizmy, takie jak **komórki pamięci i bramki**, aby skuteczniej **zarządzać długoterminowymi zależnościami i zapobiegać problemowi znikającego gradientu.**



https://miro.medium.com/v2/resize:fit:674/1*jikKbzFXCq-IYnFZankIMg.png

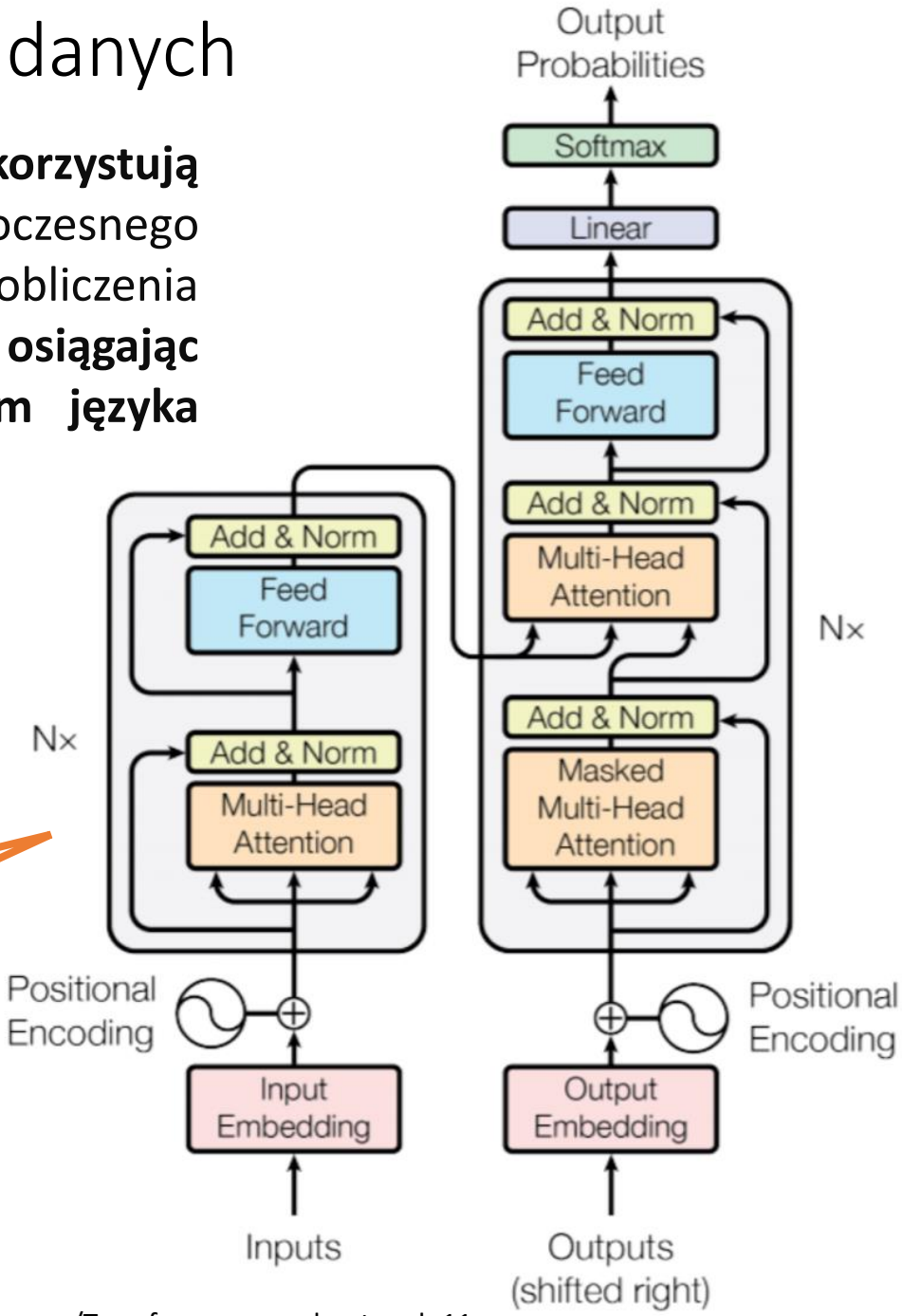
Przetwarzanie sekwencji danych

- **Sieci transformers to zaawansowane architektury**, które **wykorzystują mechanizmy uwagi (attention mechanisms)** do równoczesnego przetwarzania całych sekwencji danych, co umożliwia równoległe obliczenia i efektywne modelowanie zależności długozasięgowych, **osiągając znakomite wyniki** w zadaniach związanych **z przetwarzaniem języka naturalnego** i nie tylko.

Transformers przed labami z MIW

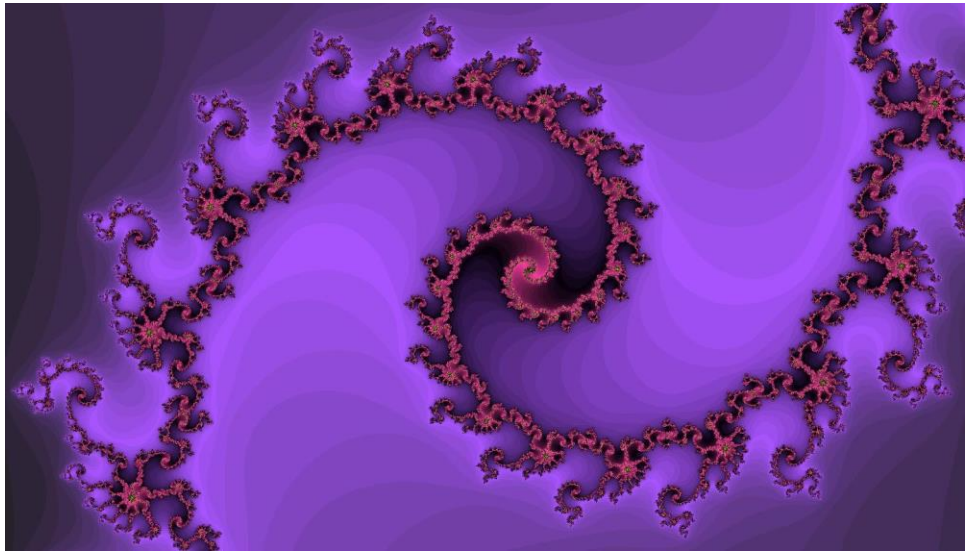
Transformers po

Transformersy na kolejnych MIWach



https://builtin.com/sites/www.builtin.com/files/styles/ckeditor_optimize/public/inline-images/Transformer-neural-network-11.png

Rekurencyjne Sieci Neuronowe (RNN)



Przykłady rekurencji

$$n! = \begin{cases} 1 \text{ dla } n = 0 \\ n(n-1)! \text{ dla } n \geq 1 \end{cases}$$

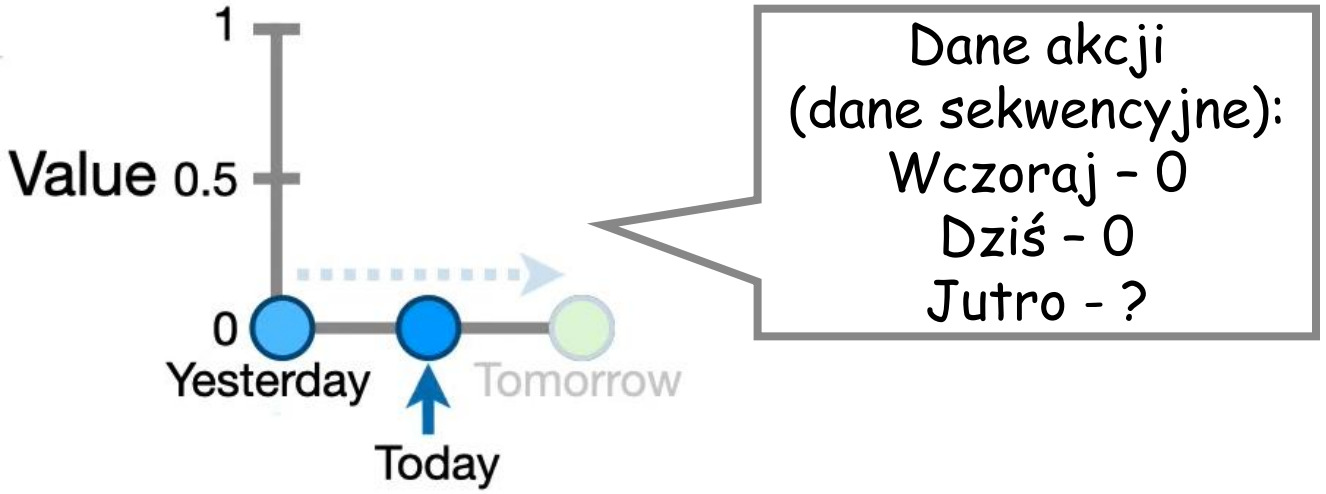
```
void recurse() {
    ... .. ...
    recurse();
    ... .. ...
}

int main() {
    ... .. ...
    recurse();
    ... .. ...
}
```

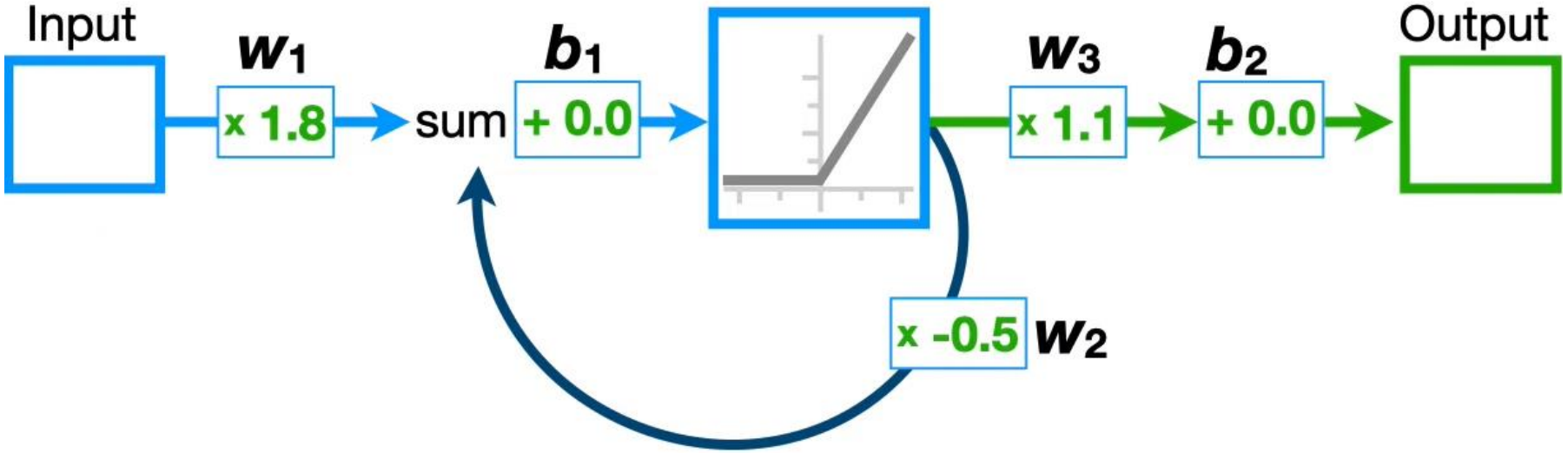
recursive call

function call

https://www.youtube.com/watch?v=AsNTP8Kwu80&list=PLblh5JKOoLUIxGDQs4LFFD--41Vzf-ME1&index=15 Recurrent Neural Networks (RNNs), Clearly Explained!!!

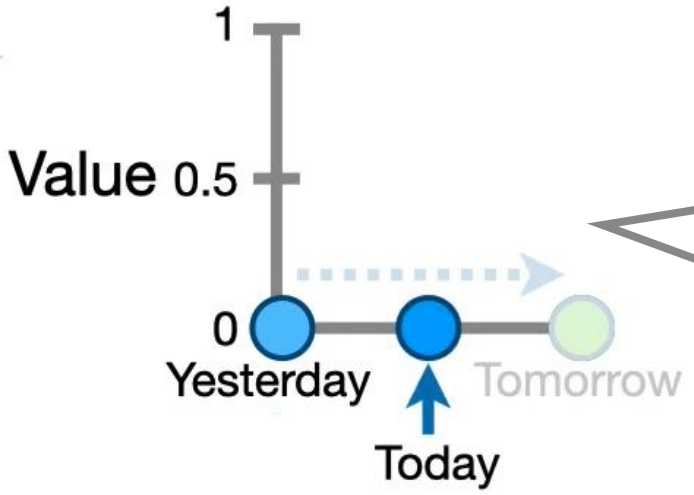
Rekurencyjne Sieci Neuronowe (RNN)





https://www.youtube.com/watch?v=AsNTP8Kwu80&list=PLblh5JKOoLUIxGDQs4LFFD--41Vzf-ME1&index=15 Recurrent Neural Networks (RNNs), Clearly Explained!!!

Rekurencyjne Sieci Neuronowe (RNN)



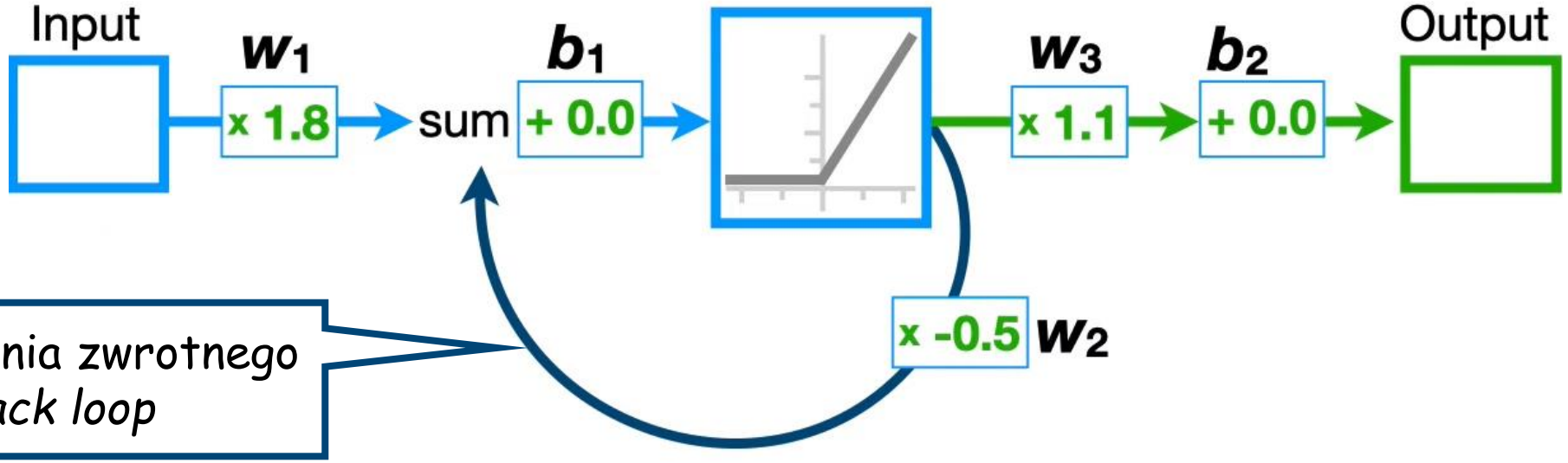
Dane akcji
(dane sekwencyjne):
Wczoraj - 0
Dziś - 0
Jutro - ?

RNN:

1 **Warstwa wejściowa** z 1 węzłem

2 **Warstwa ukryta** z 1 neuronem (w1,b1,ReLU) i pętlą sprzężenia zwrotnego (w2)

3 **Warstaw wyjściowa** z 1 neuronem (w3, b2, bez funkcji aktywacji)

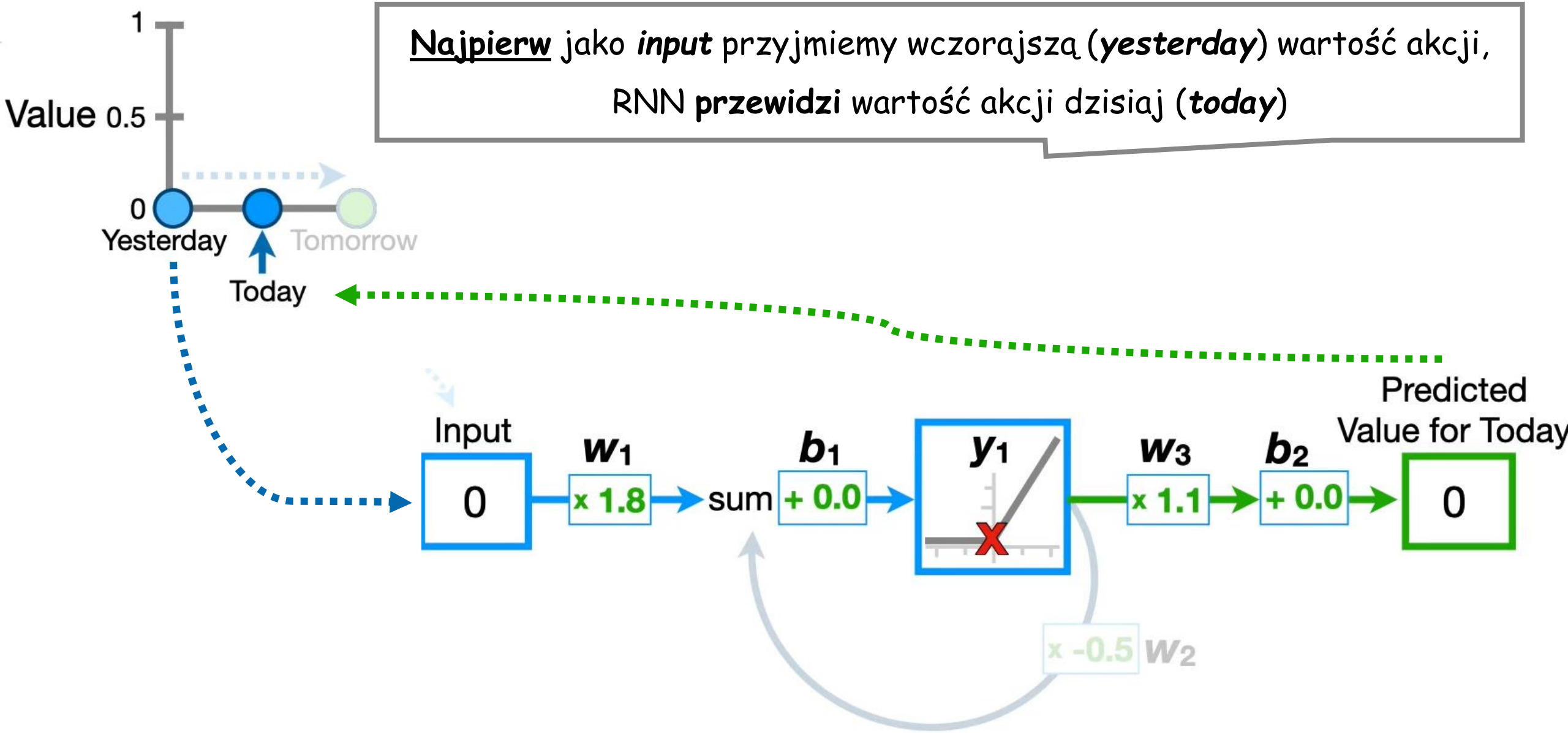


Pętla sprzężenia zwrotnego
feedback loop

https://www.youtube.com/watch?v=AsNTP8Kwu80&list=PLblh5JKOoLUIxGDQs4LFFD--41Vzf-ME1&index=15 Recurrent Neural Networks (RNNs), Clearly Explained!!!

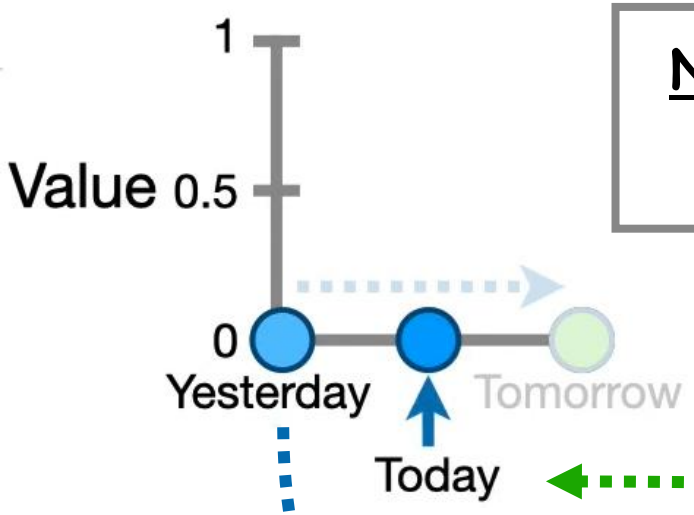
Rekurencyjne Sieci Neuronowe (RNN)

<u>Najpierw</u> jako ***input*** przyjmiemy wczorajszą (***yesterday***) wartość akcji,
RNN **przewidzi** wartość akcji dzisiaj (***today***)

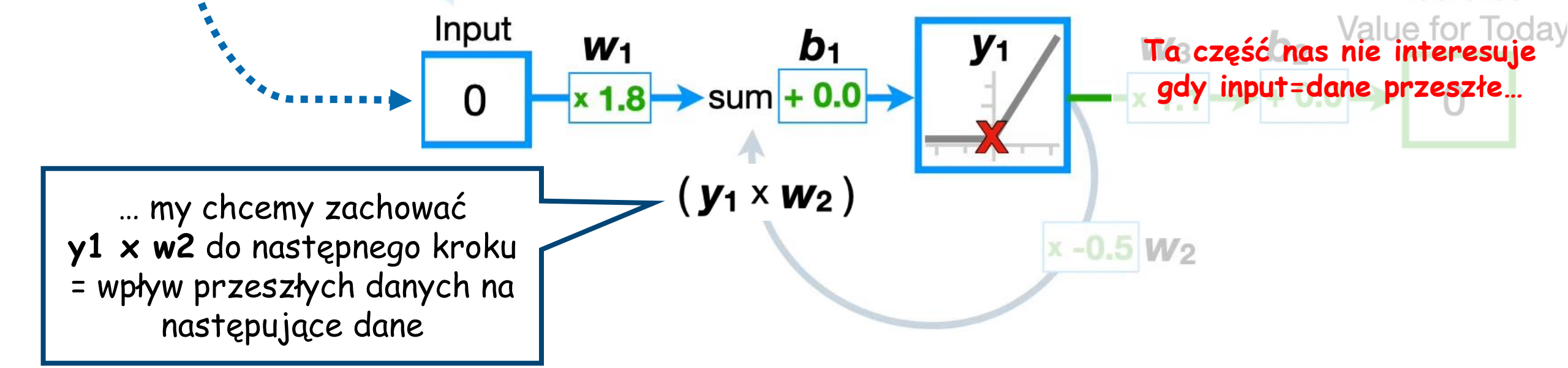


https://www.youtube.com/watch?v=AsNTP8Kwu80&list=PLblh5JKOoLUIxGDQs4LFFD--41Vzf-ME1&index=15 Recurrent Neural Networks (RNNs), Clearly Explained!!!

Rekurencyjne Sieci Neuronowe (RNN)



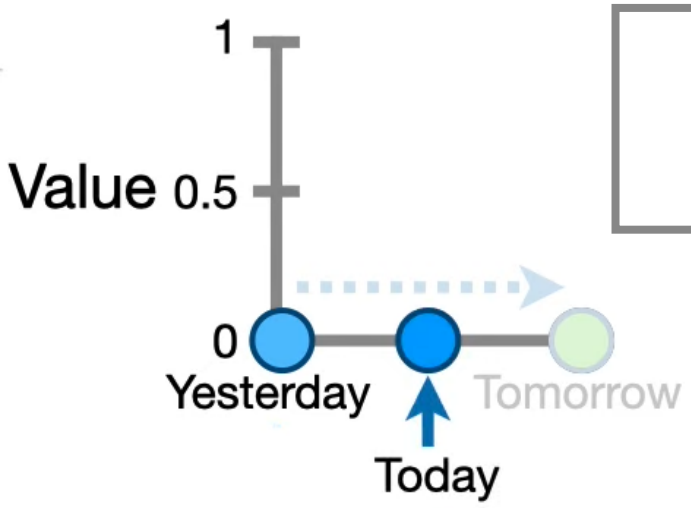
<u>Najpierw</u> jako ***input*** przyjmiemy wczorajszą (***yesterday***) wartość akcji,
RNN **przewidzi** wartość akcji dzisiaj (***today***)



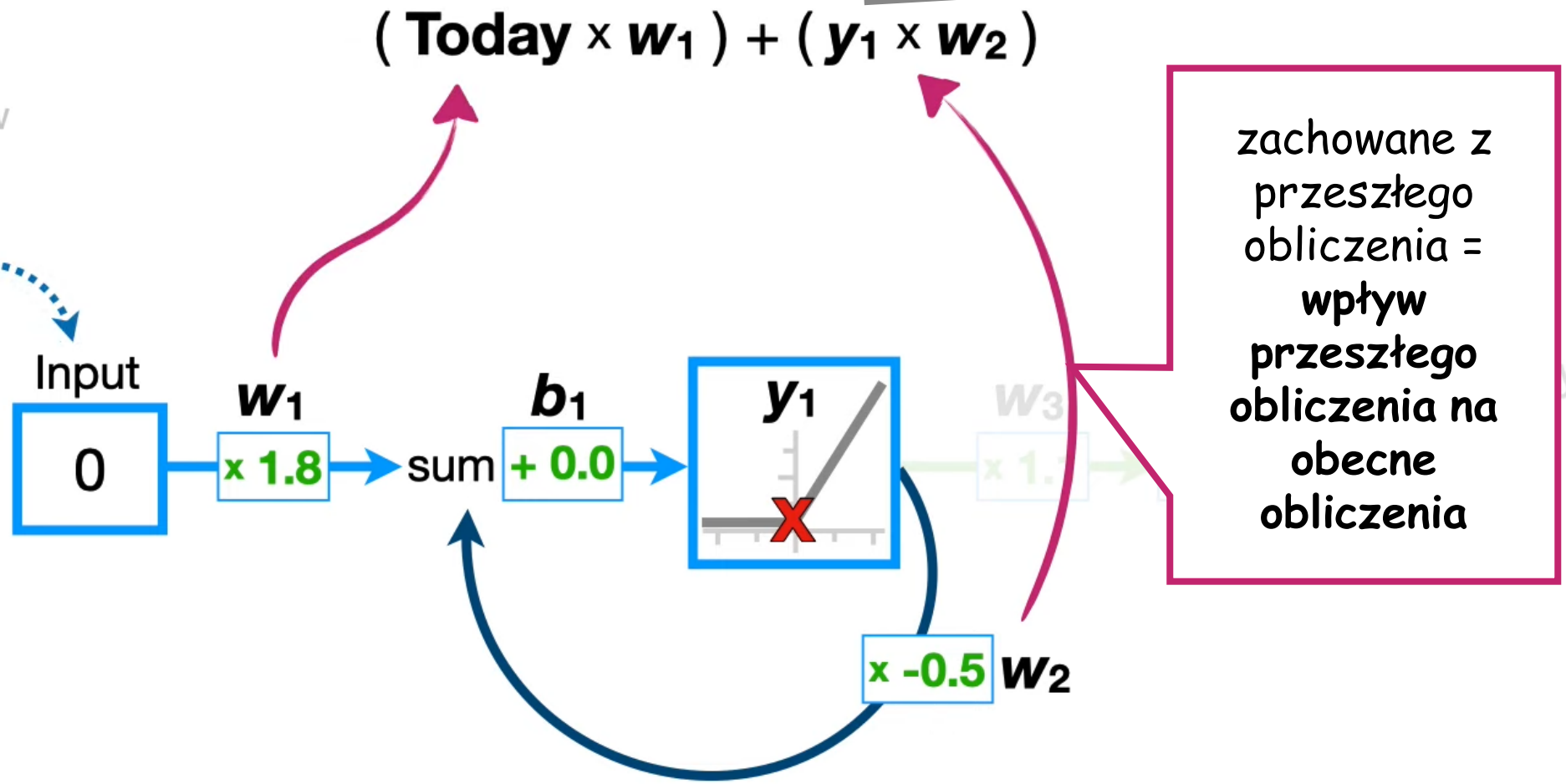
… my chcemy zachować
y1 x w2 do następnego kroku
= wpływ przeszłych danych na
następujące dane

https://www.youtube.com/watch?v=AsNTP8Kwu80&list=PLblh5JKOoLUIxGDQs4LFFD--41Vzf-ME1&index=15 Recurrent Neural Networks (RNNs), Clearly Explained!!!

Rekurencyjne Sieci Neuronowe (RNN)



Następnie jako ***input*** przyjmiemy dzisiejszą (***today***) wartość akcji,
RNN **przewidzi** wartość akcji jutro (***tomorrow***)



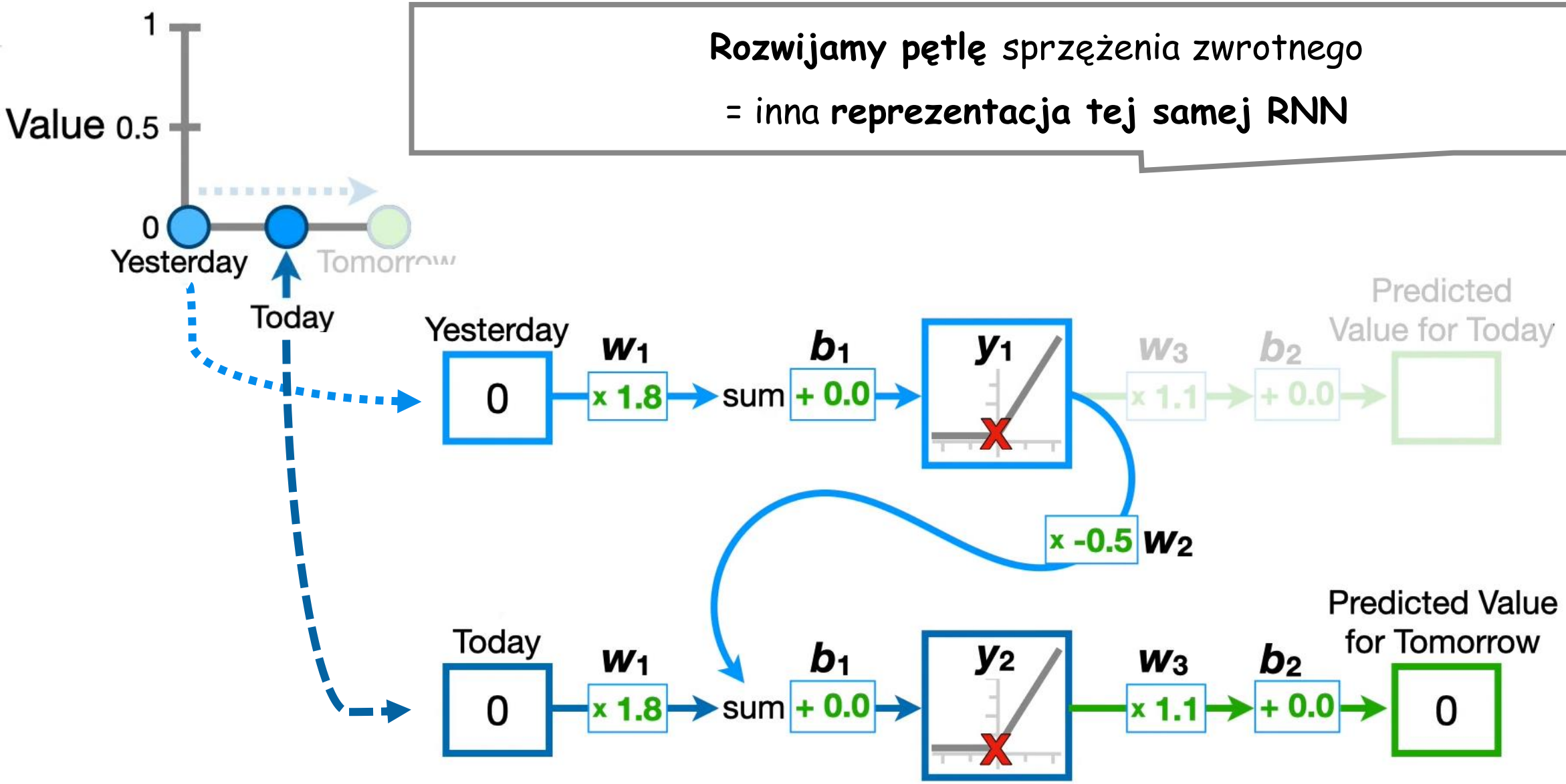
zachowane z przeszłego obliczenia = **wpływ przeszłego obliczenia na obecne obliczenia**

https://www.youtube.com/watch?v=AsNTP8Kwu80&list=PLblh5JKOoLUIxGDQs4LFFD--41Vzf-ME1&index=15 Recurrent Neural Networks (RNNs), Clearly Explained!!!

Rekurencyjne Sieci Neuronowe (RNN)

Rozwijamy pętlę sprzężenia zwrotnego

= inna **reprezentacja tej samej RNN**

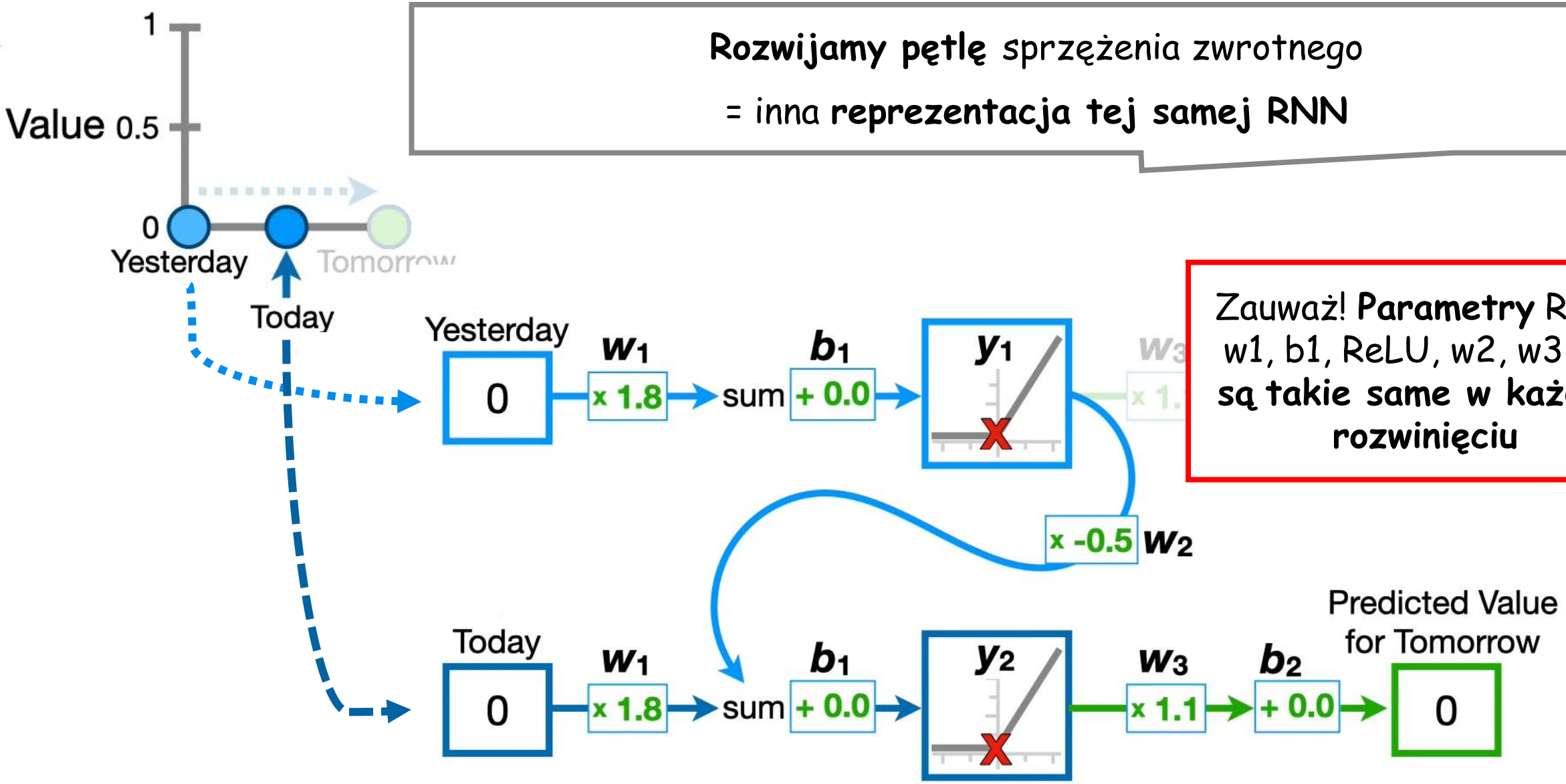


https://www.youtube.com/watch?v=AsNTP8Kwu80&list=PLblh5JKOoLUIxGDQs4LFFD--41Vzf-ME1&index=15 Recurrent Neural Networks (RNNs), Clearly Explained!!!

Rekurencyjne Sieci Neuronowe (RNN)

Rozwijamy pętlę sprzężenia zwrotnego

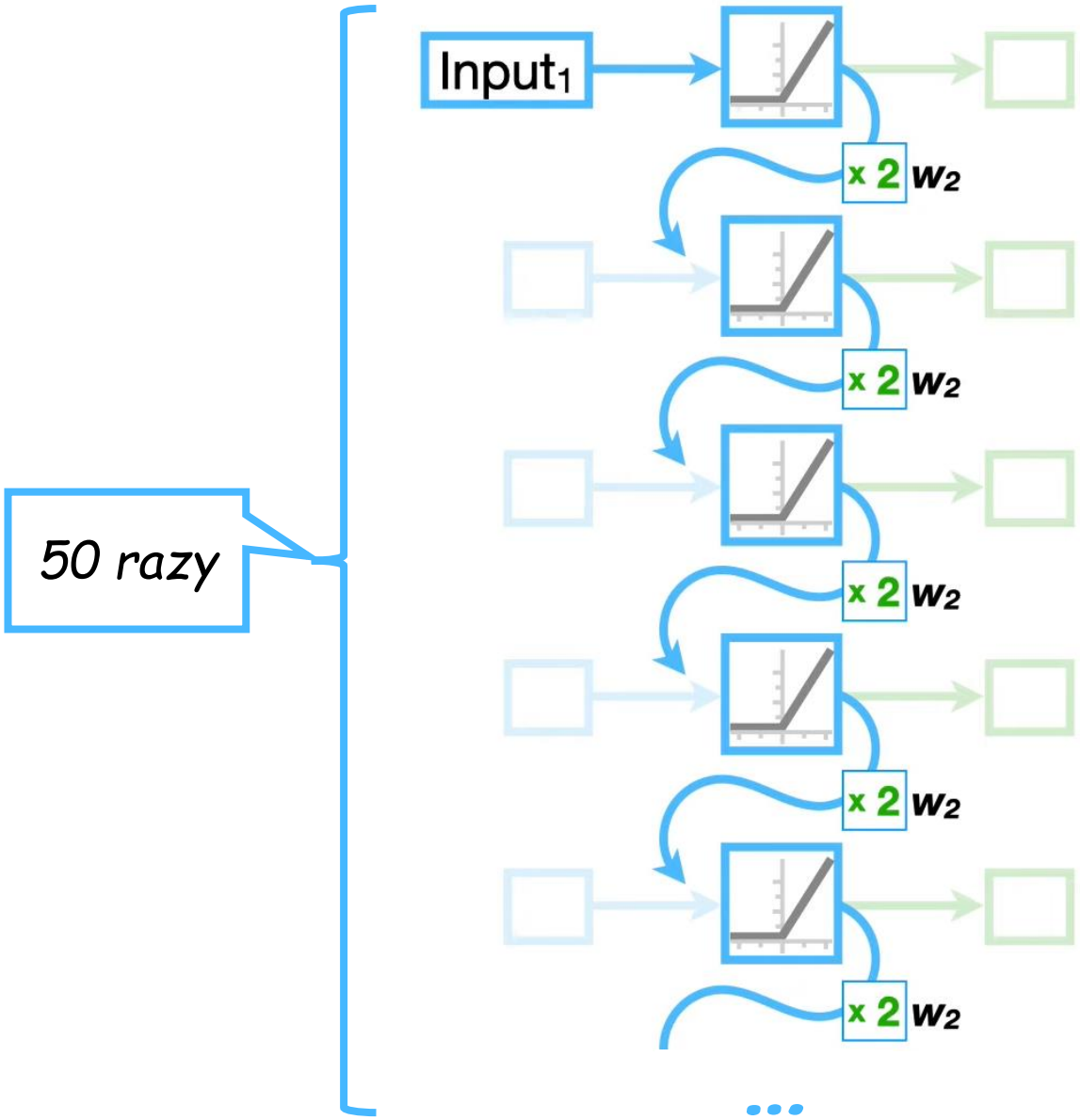
= inna **reprezentacja tej samej RNN**



Zauważ! **Parametry** RNN: w1, b1, ReLU, w2, w3, b3 **są takie same w każdym rozwinięciu**

https://www.youtube.com/watch?v=AsNTP8Kwu80&list=PLblh5JKOoLUIxGDQs4LFFD--41Vzf-ME1&index=15 Recurrent Neural Networks (RNNs), Clearly Explained!!!

Rekurencyjne Sieci Neuronowe (RNN) – *a w czym problem?*



Zakładamy, że uwzględniami ***50 sekwencyjnych danych****:*

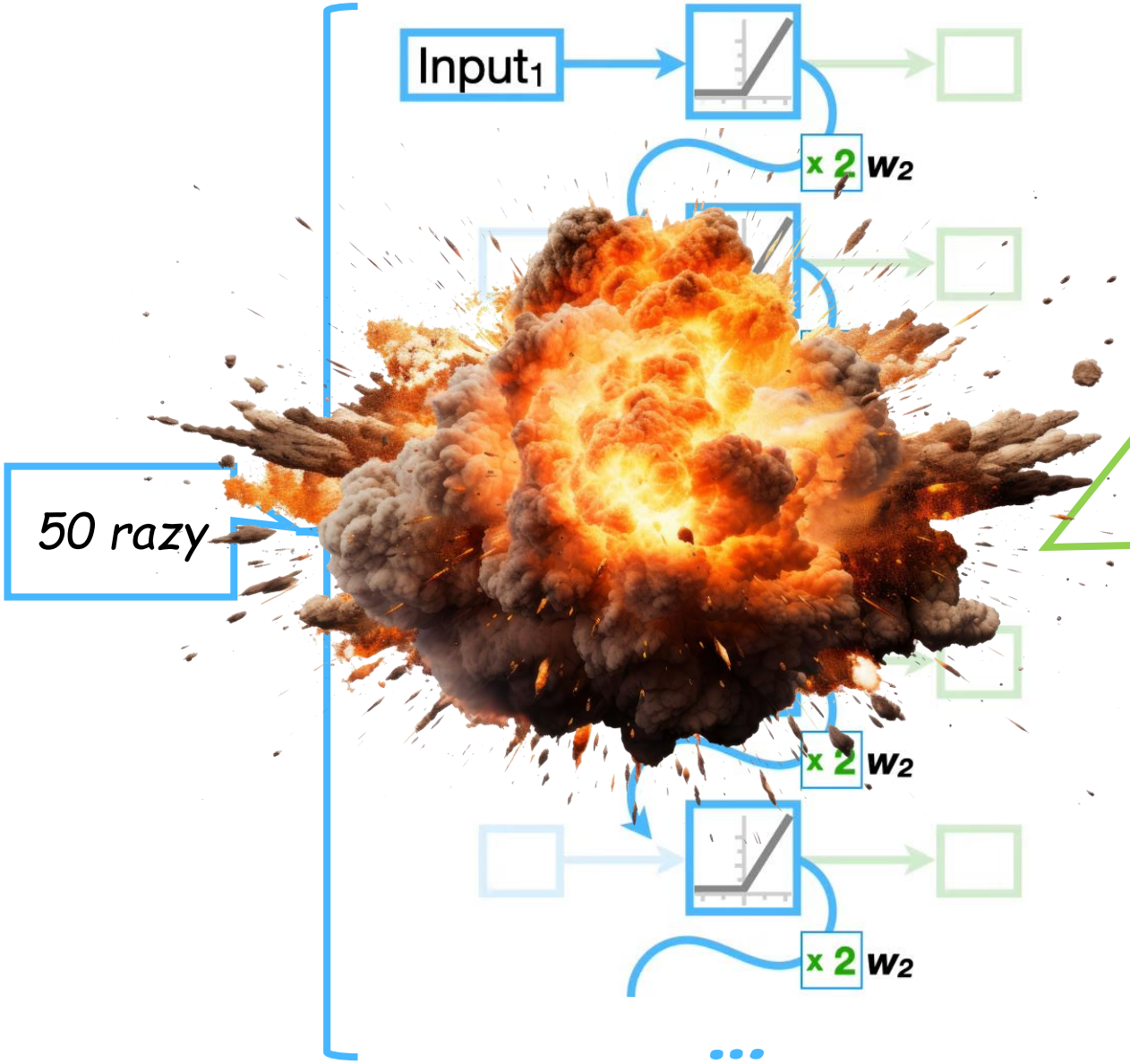
propagacja wsteczna *(backpropagation) ->*

pochodna funkcji złożonej (chain rule) ->

Input1 x **(w2=2.0)^50** ->

https://www.youtube.com/watch?v=AsNTP8Kwu80&list=PLblh5JKOoLUIxGDQs4LFFD--41Vzf-ME1&index=15 Recurrent Neural Networks (RNNs), Clearly Explained!!!

Rekurencyjne Sieci Neuronowe (RNN) – *a w czym problem?*



Zakładamy, że uwzględniami ***50 sekwencyjnych danych****:*

propagacja wsteczna *(backpropagation)* ->

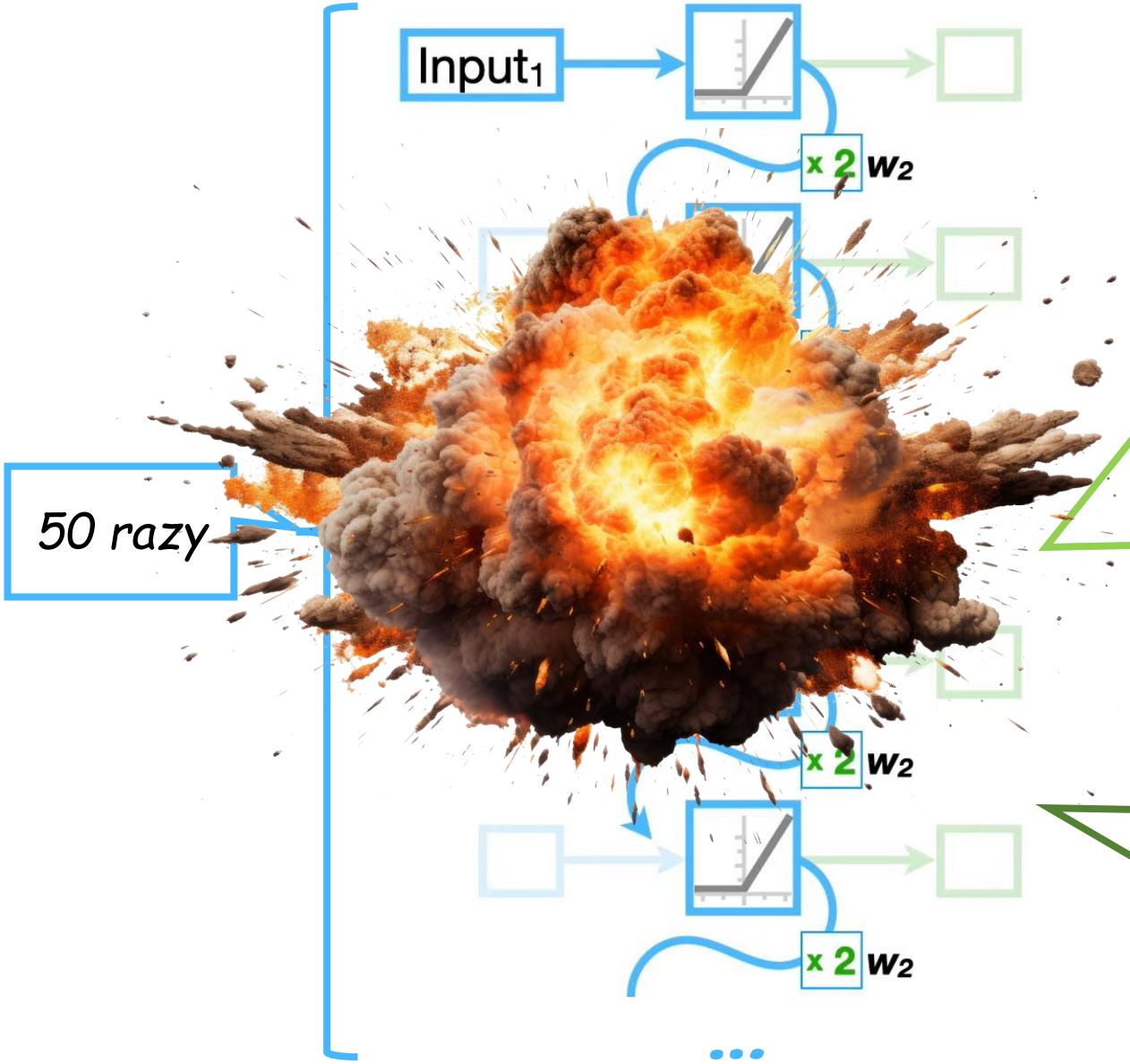
pochodna funkcji złożonej (chain rule) ->

Input1 x **(w2=2.0)^50** ->

eksplodujący gradient

https://www.youtube.com/watch?v=AsNTP8Kwu80&list=PLblh5JKOoLUIxGDQs4LFFD--41Vzf-ME1&index=15 Recurrent Neural Networks (RNNs), Clearly Explained!!!

Rekurencyjne Sieci Neuronowe (RNN) – *a w czym problem?*



Zakładamy, że uwzględniami ***50 sekwencyjnych danych****:*

propagacja wsteczna *(backpropagation) ->*

pochodna funkcji złożonej (chain rule) ->

Input1 x **(w2=2.0)^50** ->

eksplodujący gradient

Analogicznie dla

Input1 x **(w2=0.5)^50** ->

znikający gradient

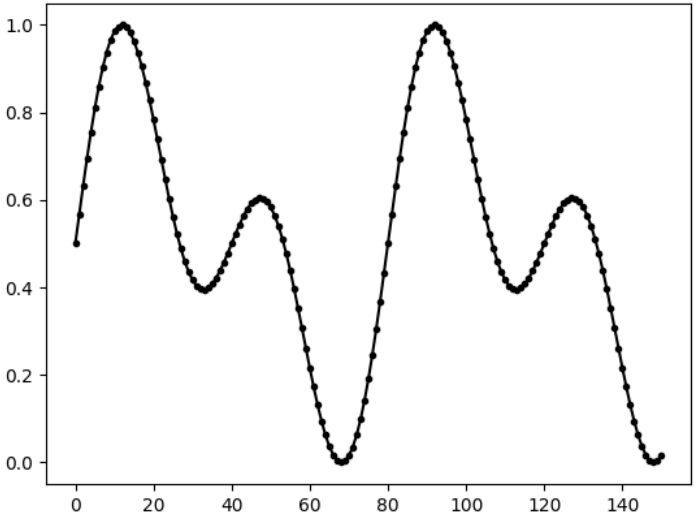
https://www.youtube.com/watch?v=AsNTP8Kwu80&list=PLblh5JKOoLUIxGDQs4LFFD--41Vzf-ME1&index=15 Recurrent Neural Networks (RNNs), Clearly Explained!!!

Kodujemy

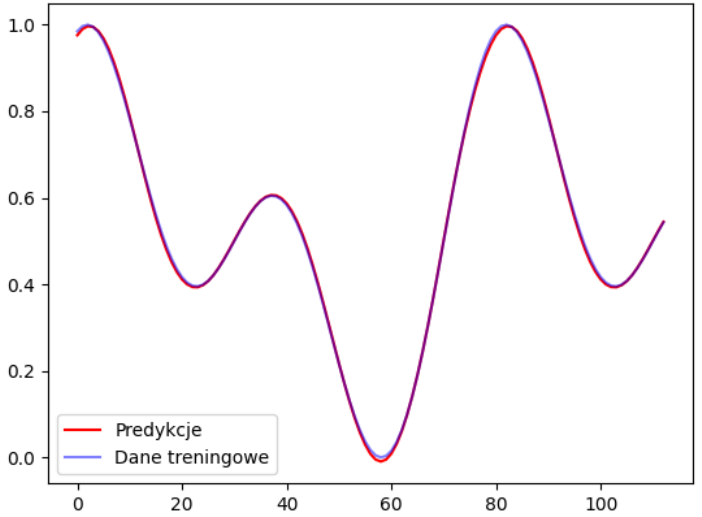
Źródła inspiracji : https://keras.io/api/layers/recurrent_layers/

public/mmajew/MIW/10/ *00_rnn_function .py / .ipynb*

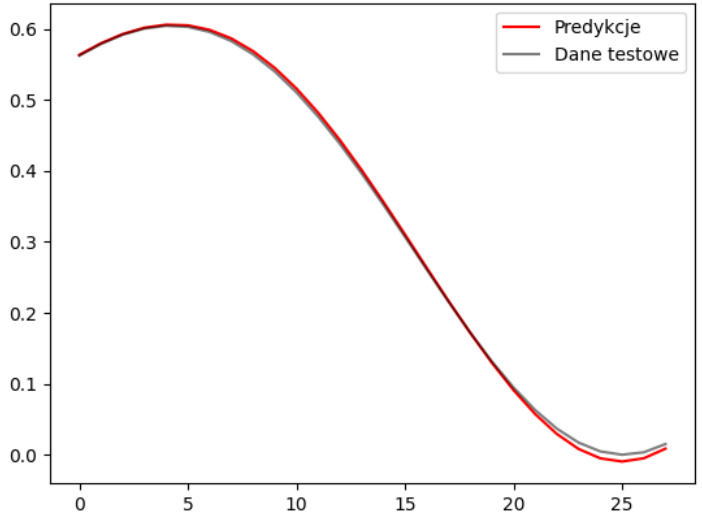
Ciąg czasowy



Dane treningowe



Dane testowe



UWAGA! Jako input wchodzą ciągi wartości ‚y' (X nas NIE obchodzi!)

Keras

Google colab

Kodujemy

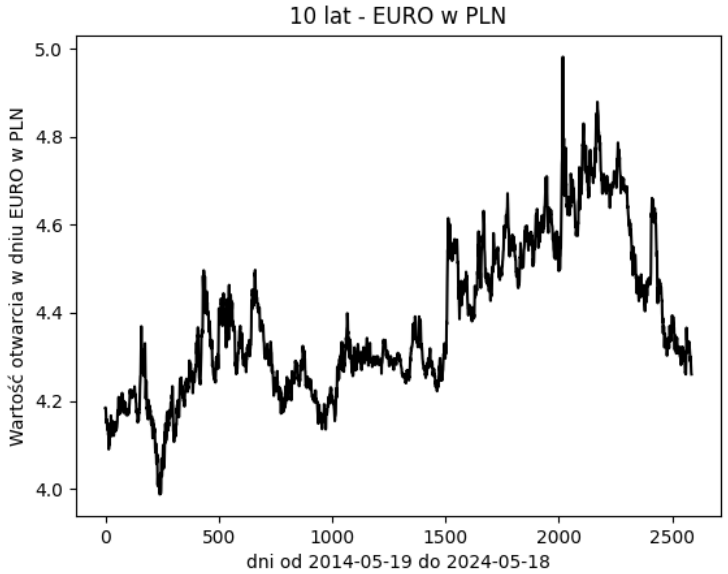
Źródła inspiracji : https://keras.io/api/layers/recurrent_layers/rnn/

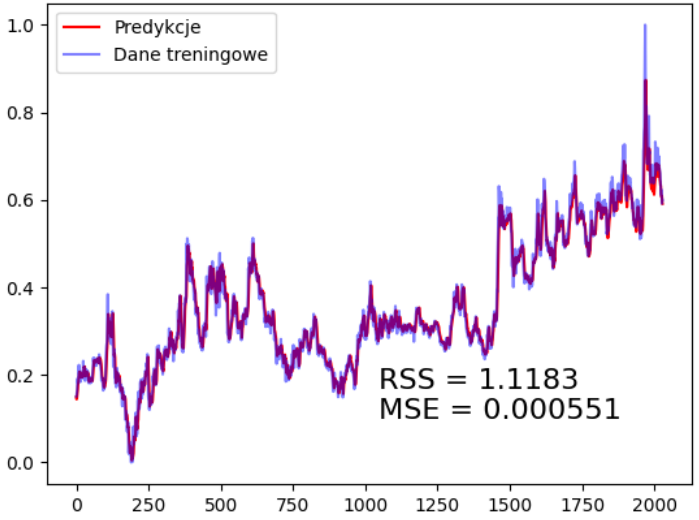
public/mmajew/MIW/10/ *01_rnn_EURtoPLN .py / .ipynb*

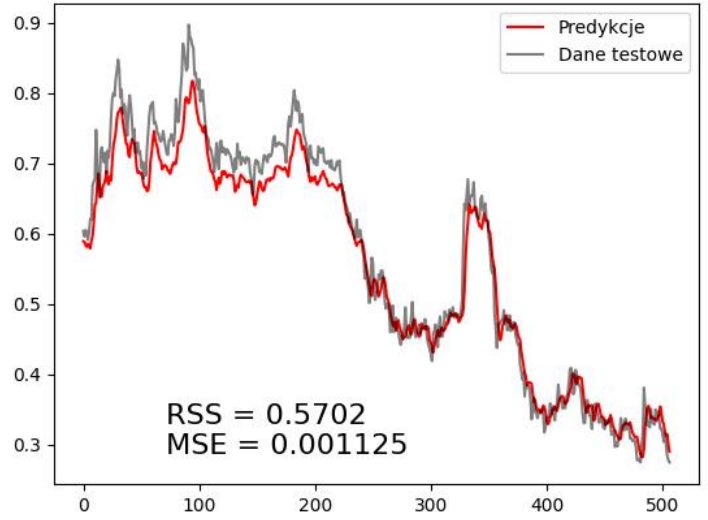
Ciąg czasowy

Dane treningowe

Dane testowe



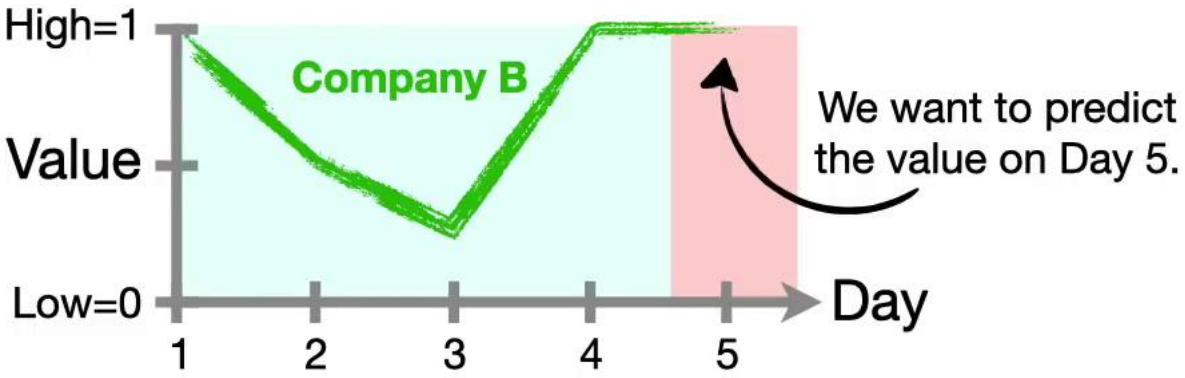




Keras

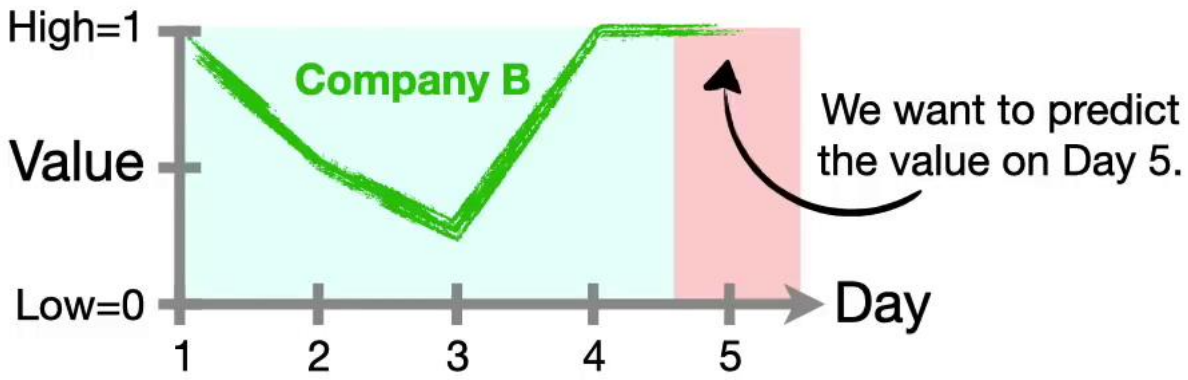
Google colab

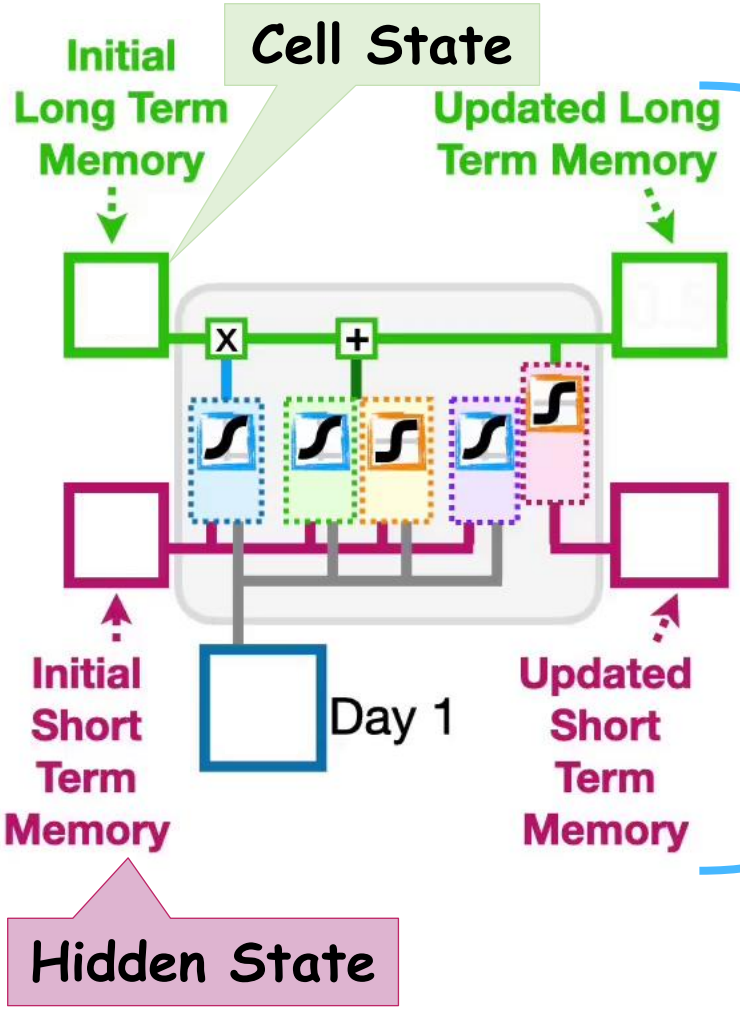
Long Short-Term Memory (LSTM)



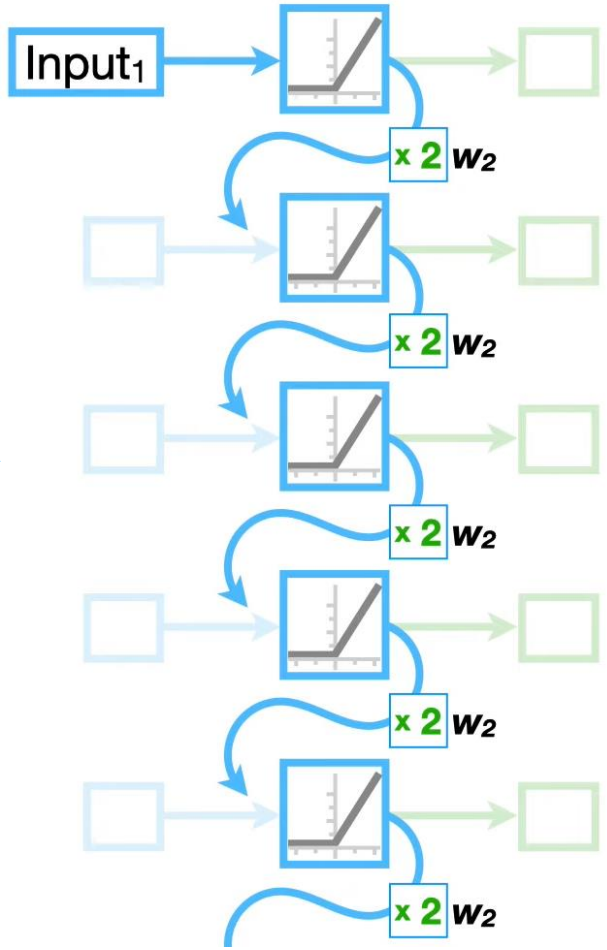
https://www.youtube.com/watch?v=YCzL96nL7j0& Long Short-Term Memory (LSTM), Clearly Explained

Long Short-Term Memory (LSTM)



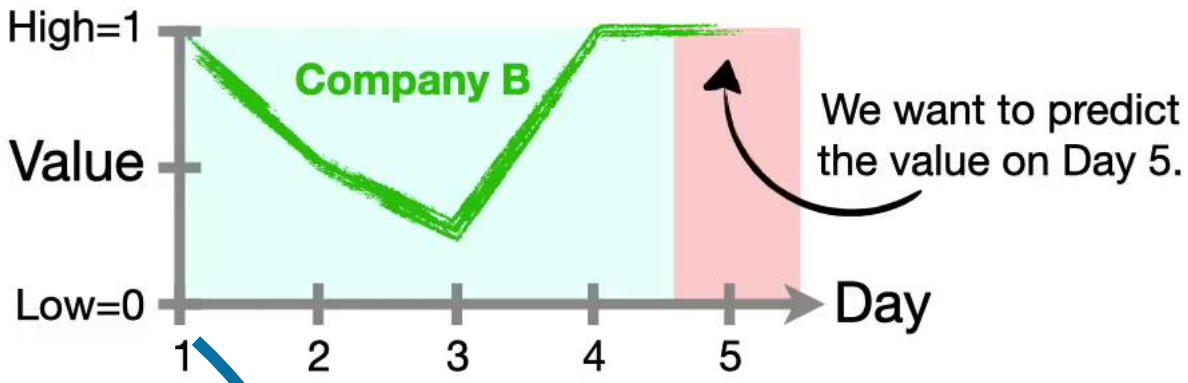


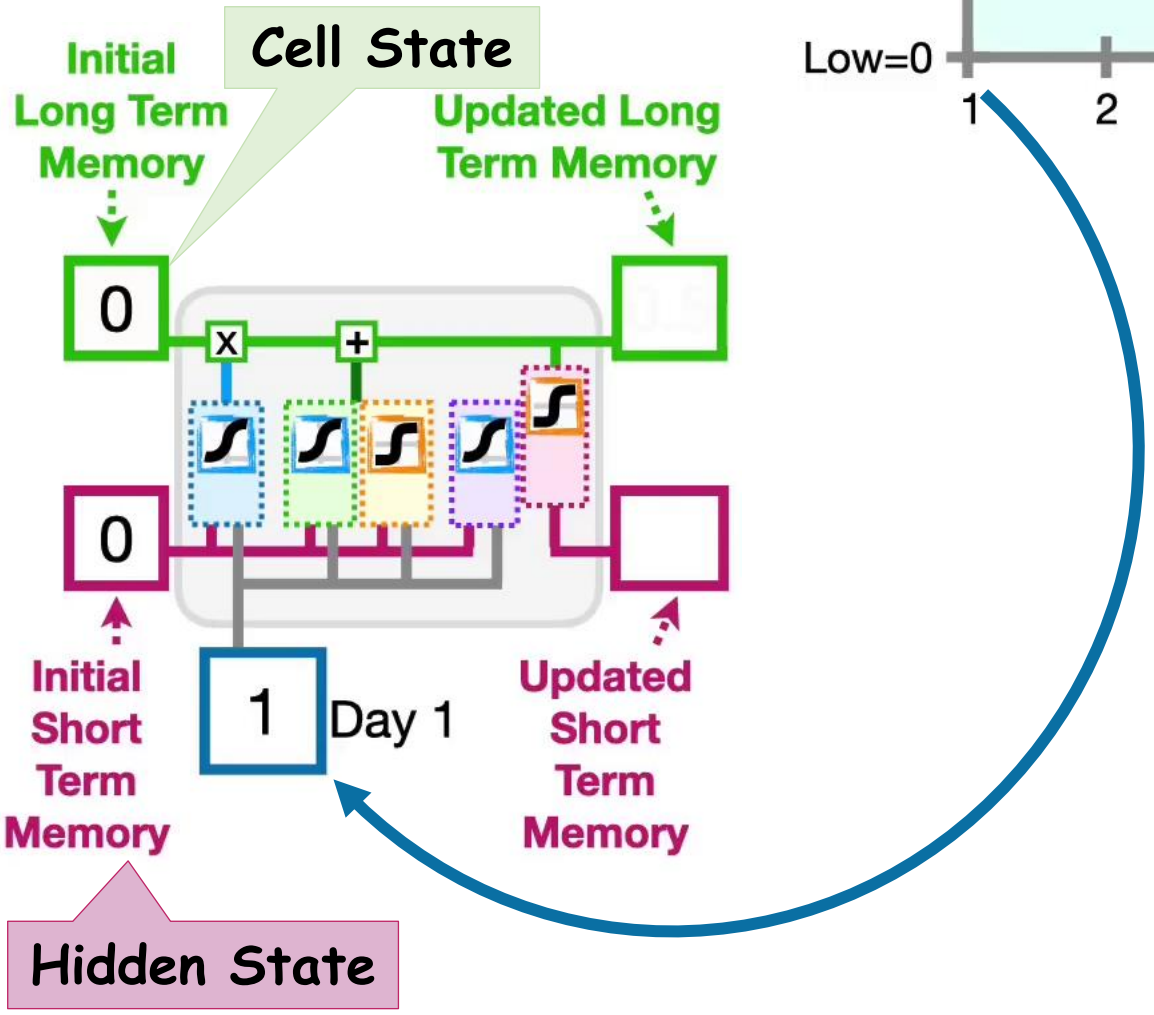
Jednostka węzła **LSTM** (rozwijamy tak jak warstwę rekurencyjną)



https://www.youtube.com/watch?v=YCzL96nL7j0& Long Short-Term Memory (LSTM), Clearly Explained

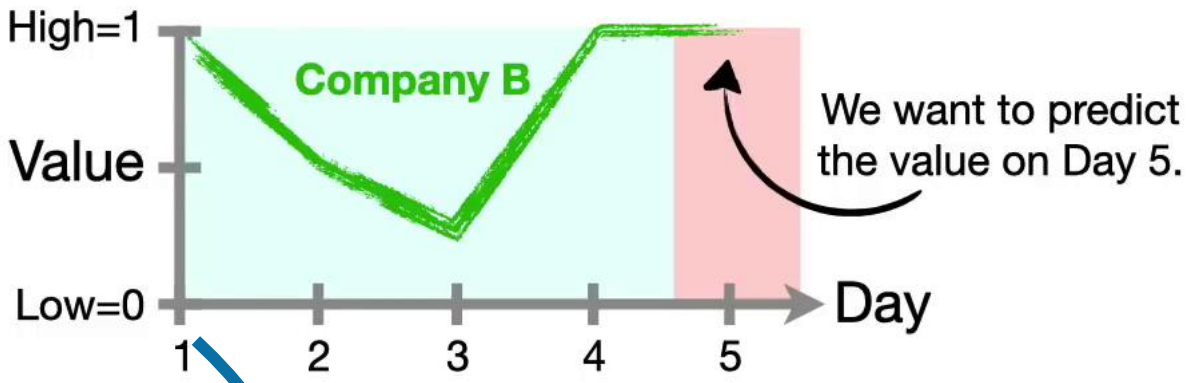
Long Short-Term Memory (LSTM)

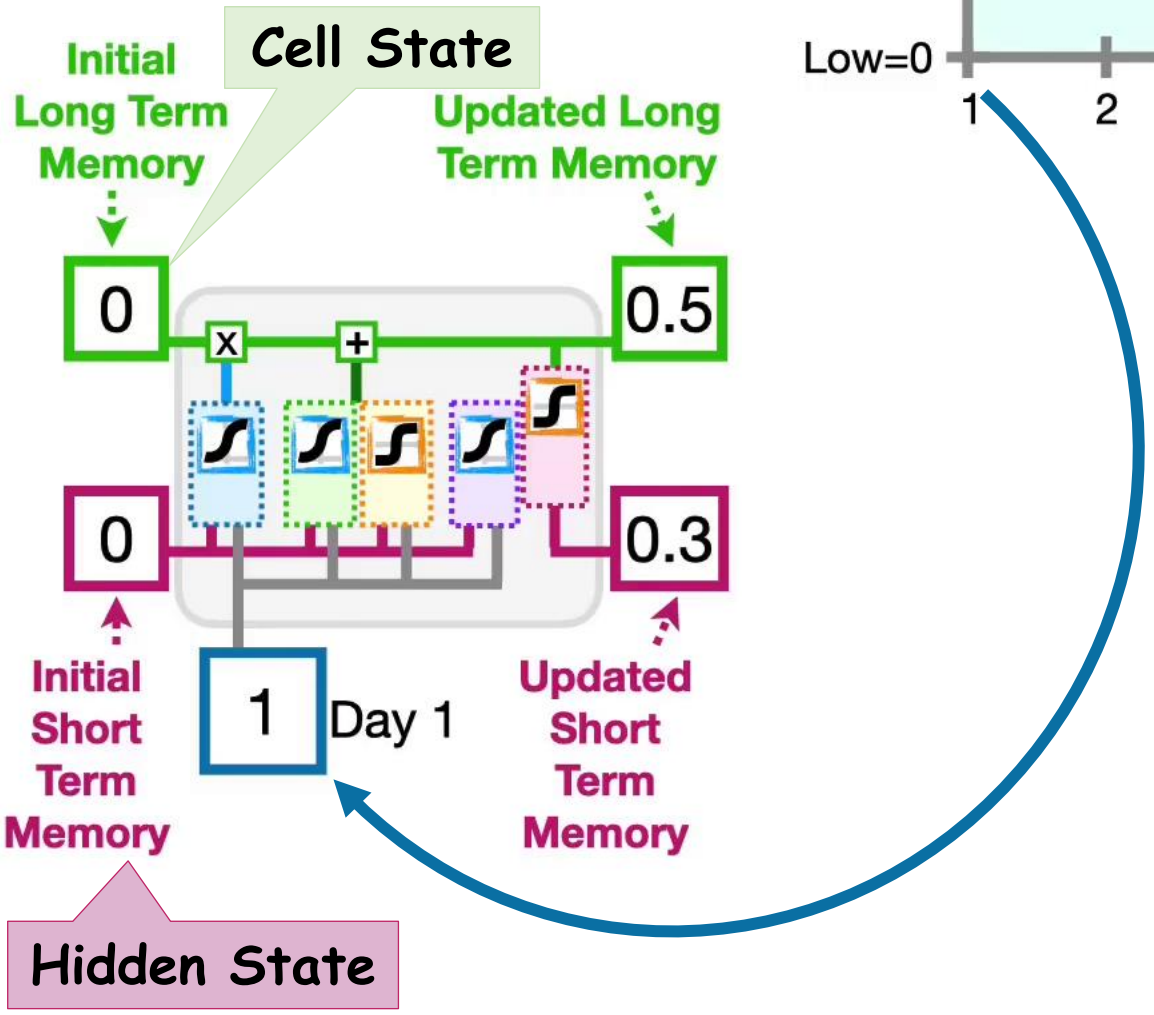




https://www.youtube.com/watch?v=YCzL96nL7j0& Long Short-Term Memory (LSTM), Clearly Explained

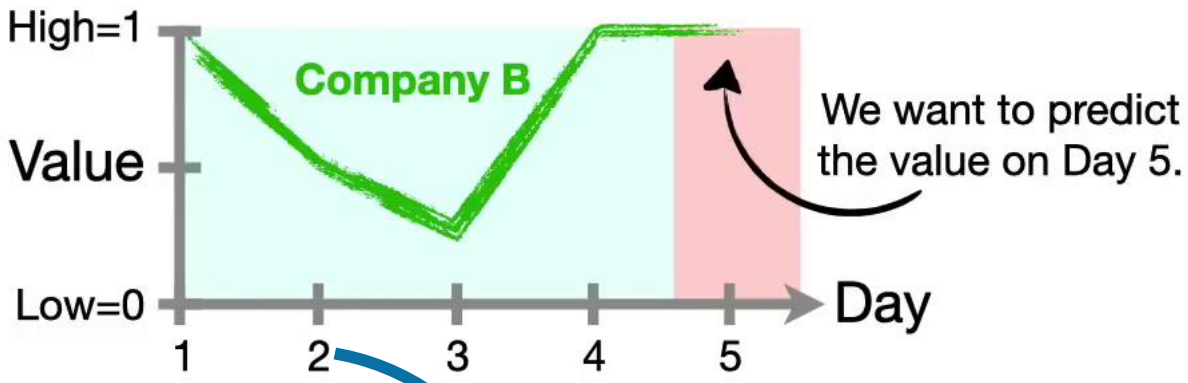
Long Short-Term Memory (LSTM)

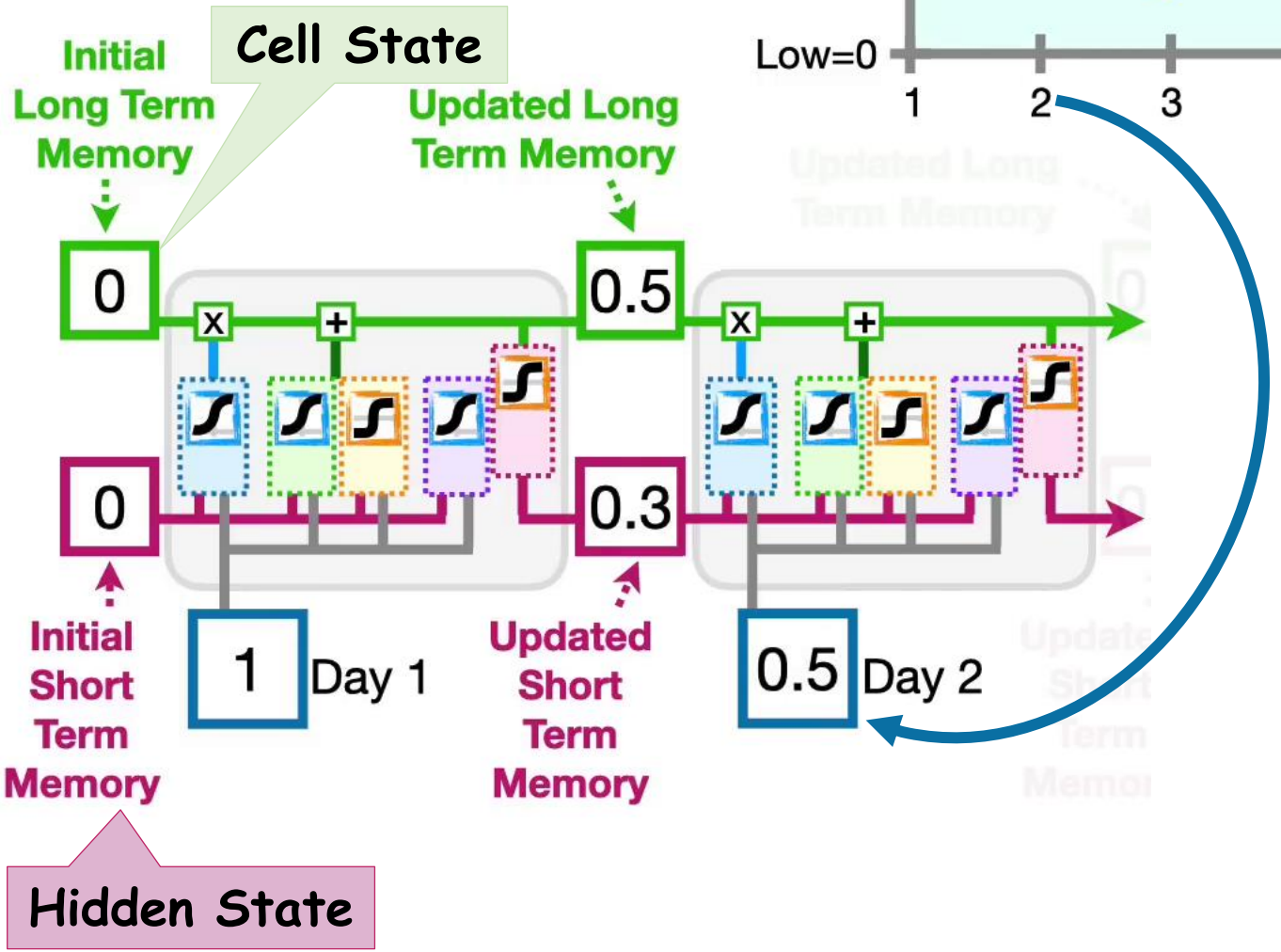




https://www.youtube.com/watch?v=YCzL96nL7j0& Long Short-Term Memory (LSTM), Clearly Explained

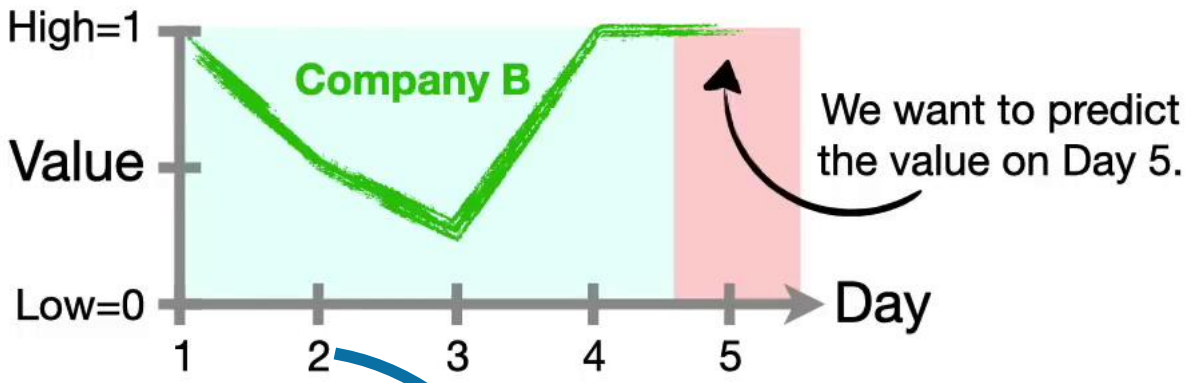
Long Short-Term Memory (LSTM)

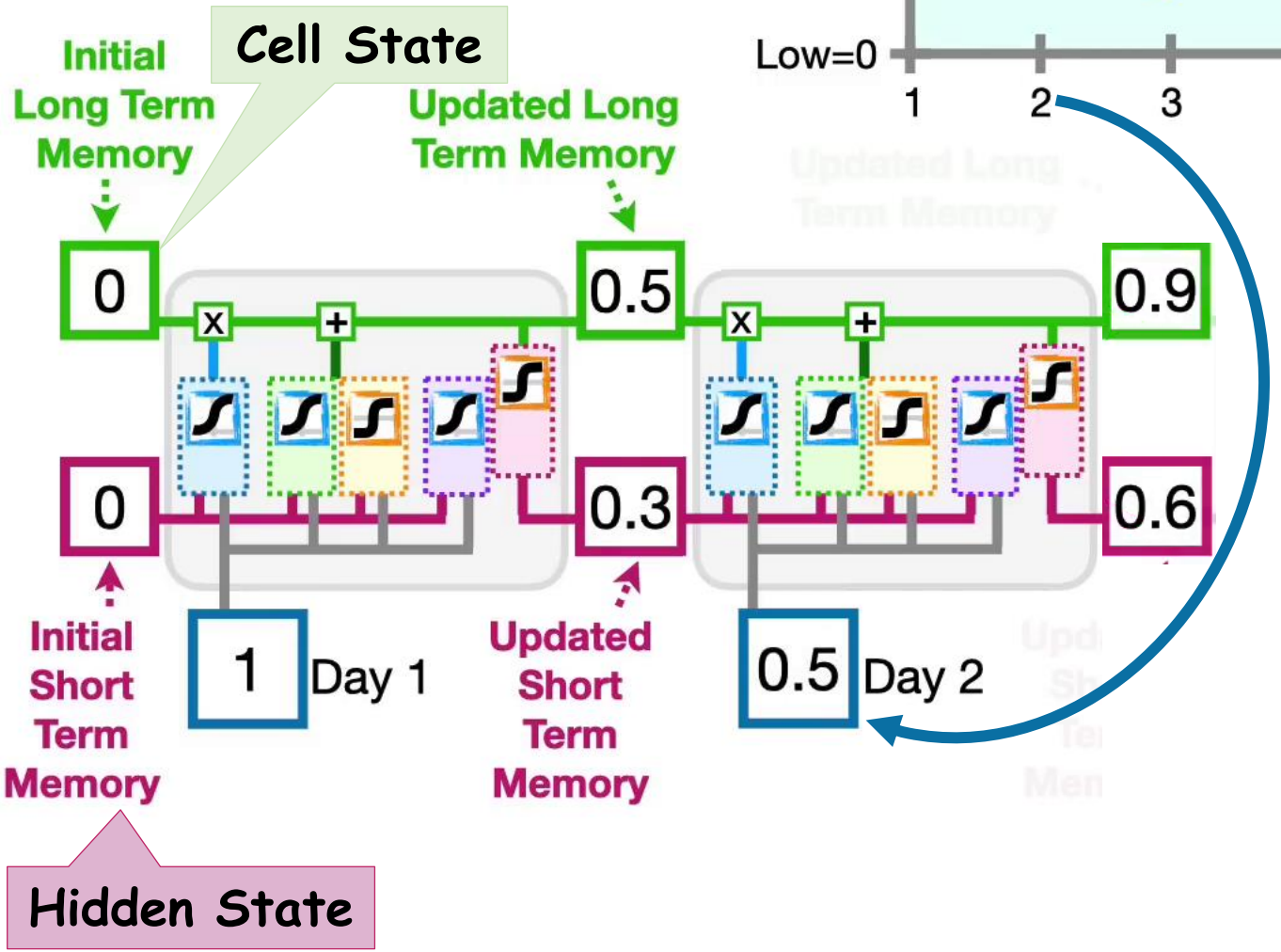




https://www.youtube.com/watch?v=YCzL96nL7j0& Long Short-Term Memory (LSTM), Clearly Explained

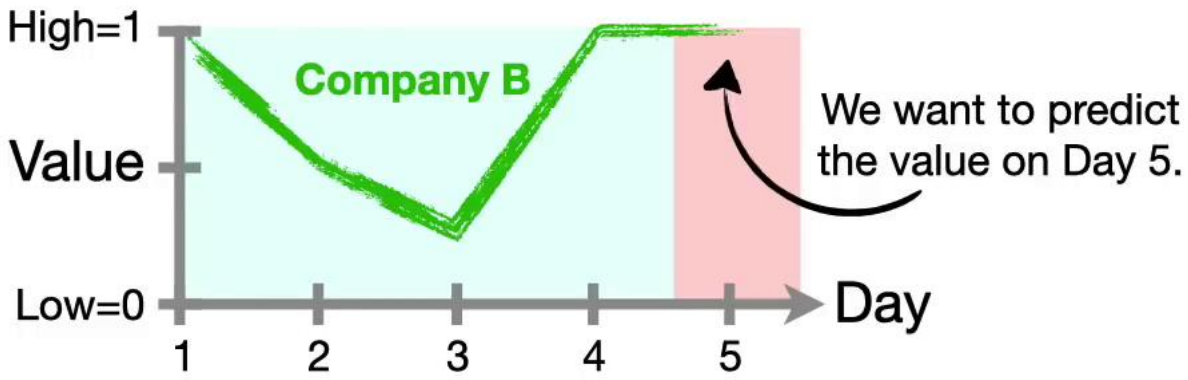
Long Short-Term Memory (LSTM)

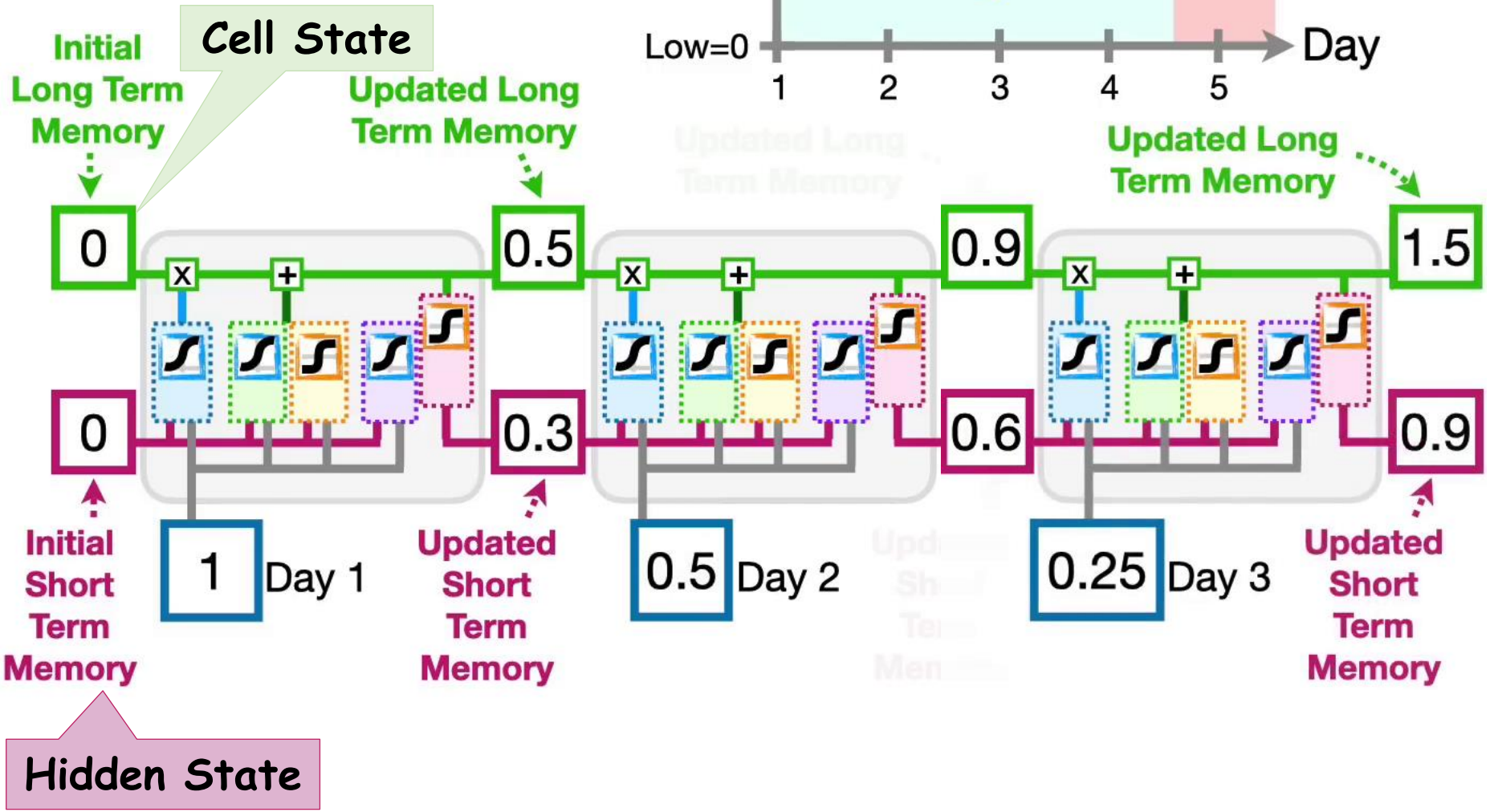




https://www.youtube.com/watch?v=YCzL96nL7j0& Long Short-Term Memory (LSTM), Clearly Explained

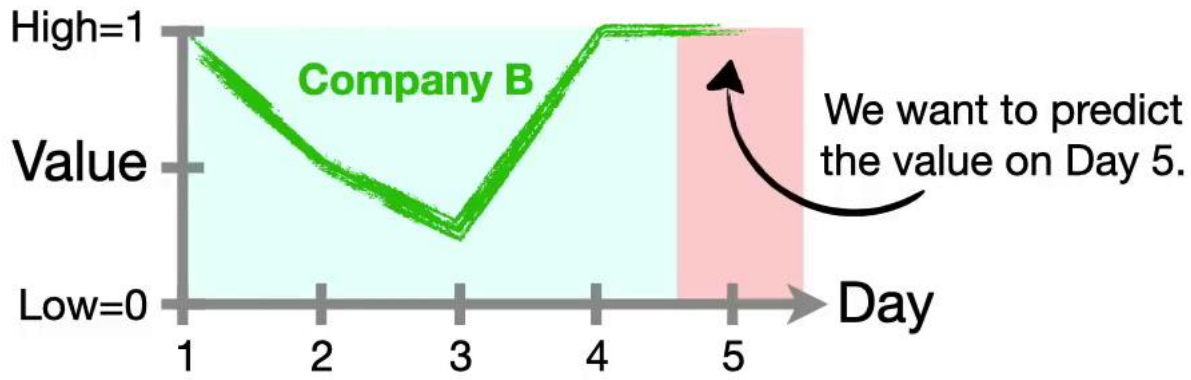
Long Short-Term Memory (LSTM)

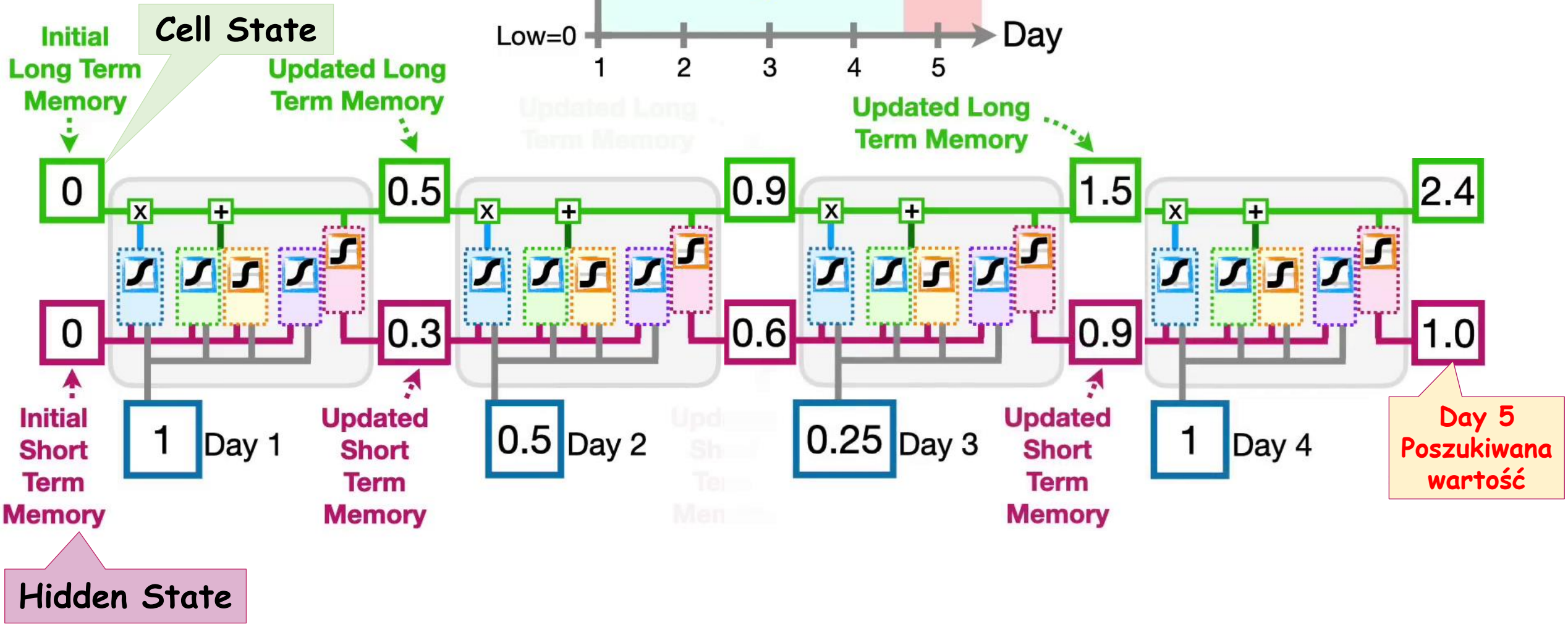




https://www.youtube.com/watch?v=YCzL96nL7j0& Long Short-Term Memory (LSTM), Clearly Explained

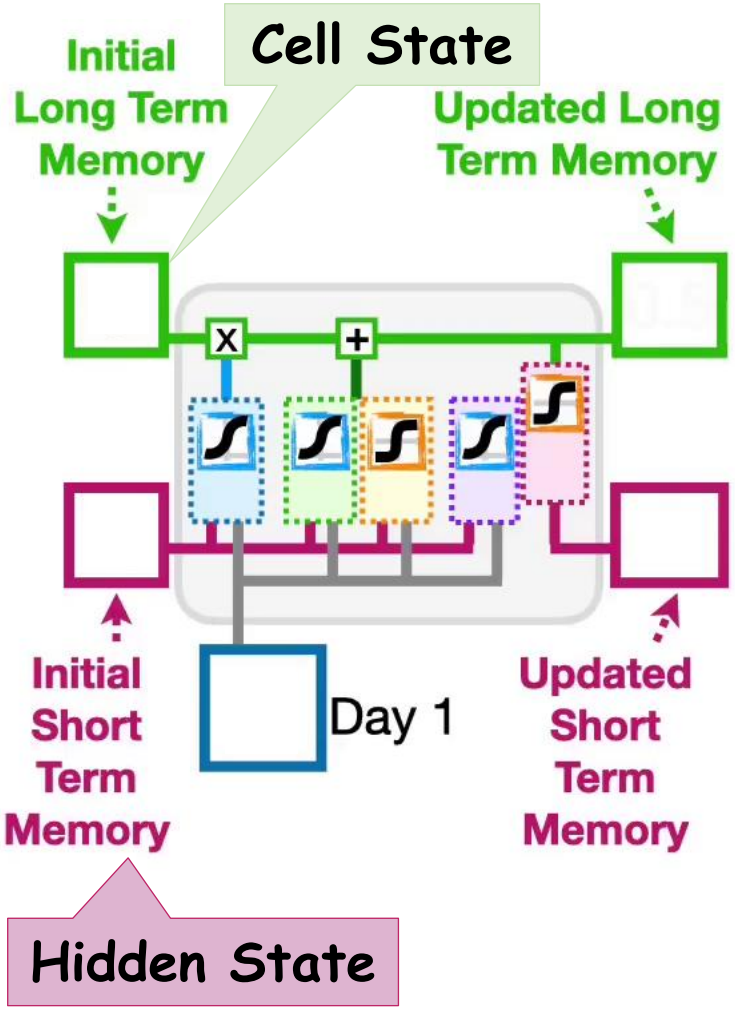
Long Short-Term Memory (LSTM)





https://www.youtube.com/watch?v=YCzL96nL7j0& Long Short-Term Memory (LSTM), Clearly Explained

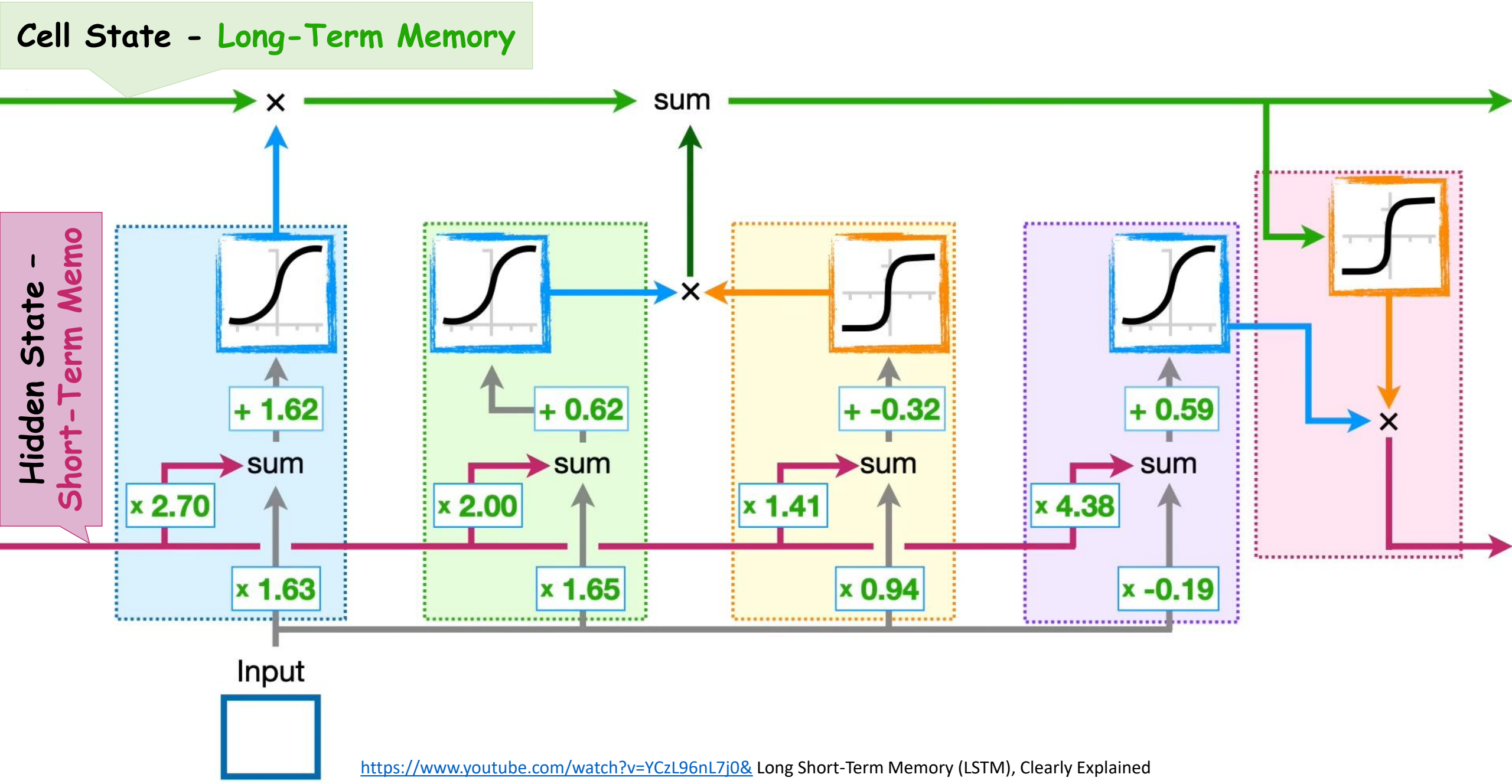
Long Short-Term Memory (LSTM)



A co się dzieje w jednostce LSTM?

https://www.youtube.com/watch?v=YCzL96nL7j0& Long Short-Term Memory (LSTM), Clearly Explained

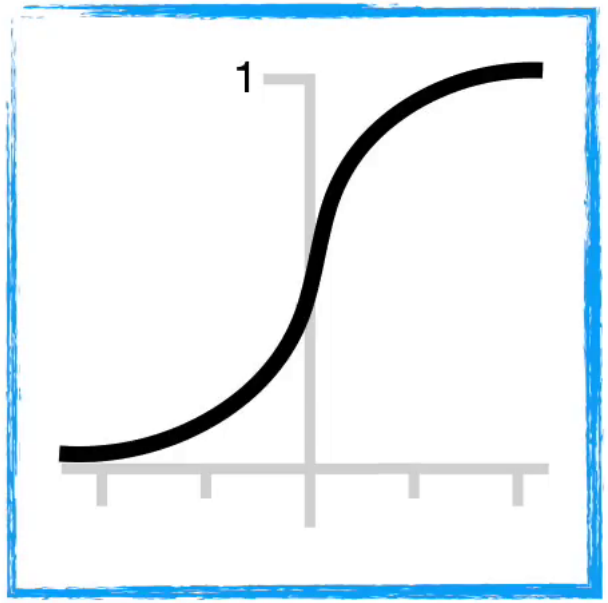
Long Short-Term Memory unit (jednostka LSTM)



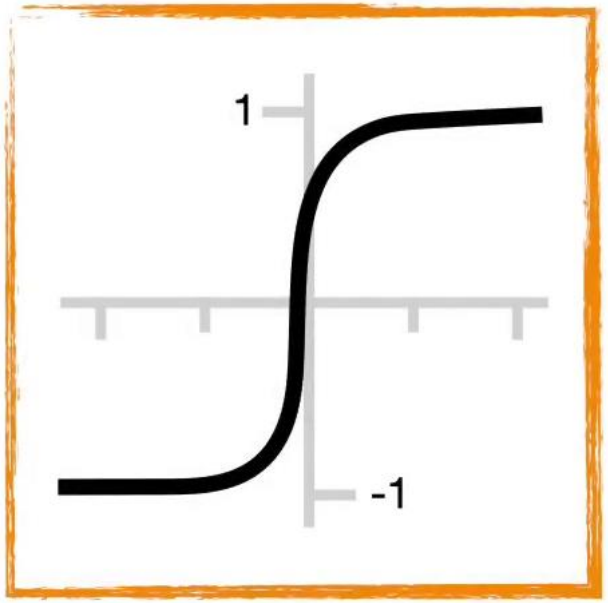
https://www.youtube.com/watch?v=YCzL96nL7j0& Long Short-Term Memory (LSTM), Clearly Explained

Long Short-Term Memory unit (jednostka LSTM)

Sigmoid – od 0 do 1



Hyperbolic tangent – od -1 do 1



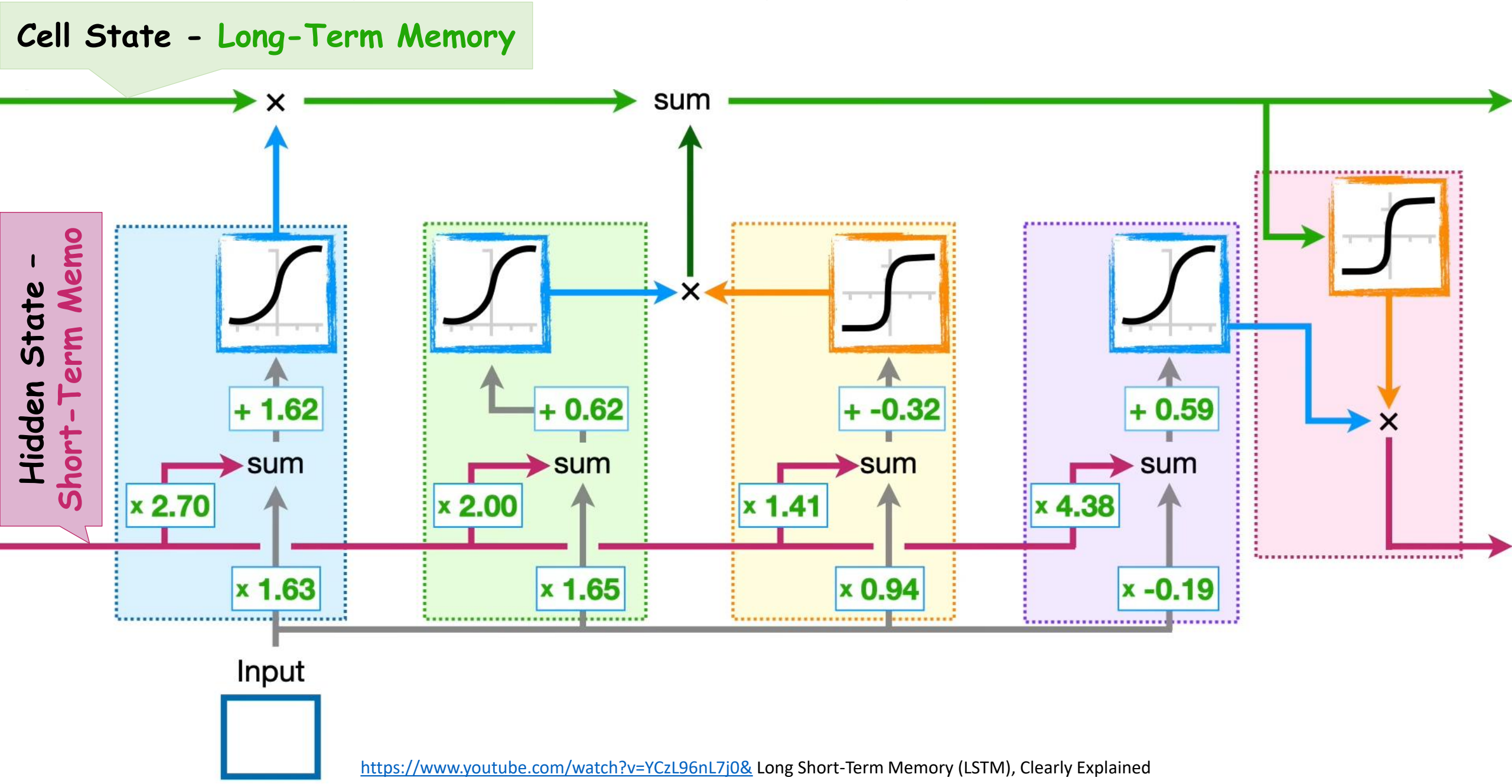
$$f(x) = \frac{e^x}{e^x + 1} \qquad f(x) = \frac{e^x}{e^x + 1} \qquad f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \qquad f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

$$f(10) = \frac{e^{10}}{e^{10} + 1} \qquad f(-5) = \frac{e^{-5}}{e^{-5} + 1} \qquad f(2) = \frac{e^2 - e^{-2}}{e^2 + e^{-2}} \qquad f(-5) = \frac{e^{-5} - e^5}{e^{-5} + e^5}$$

$$= 0.99995 \qquad = 0.01 \qquad = 0.96 \qquad = -1$$

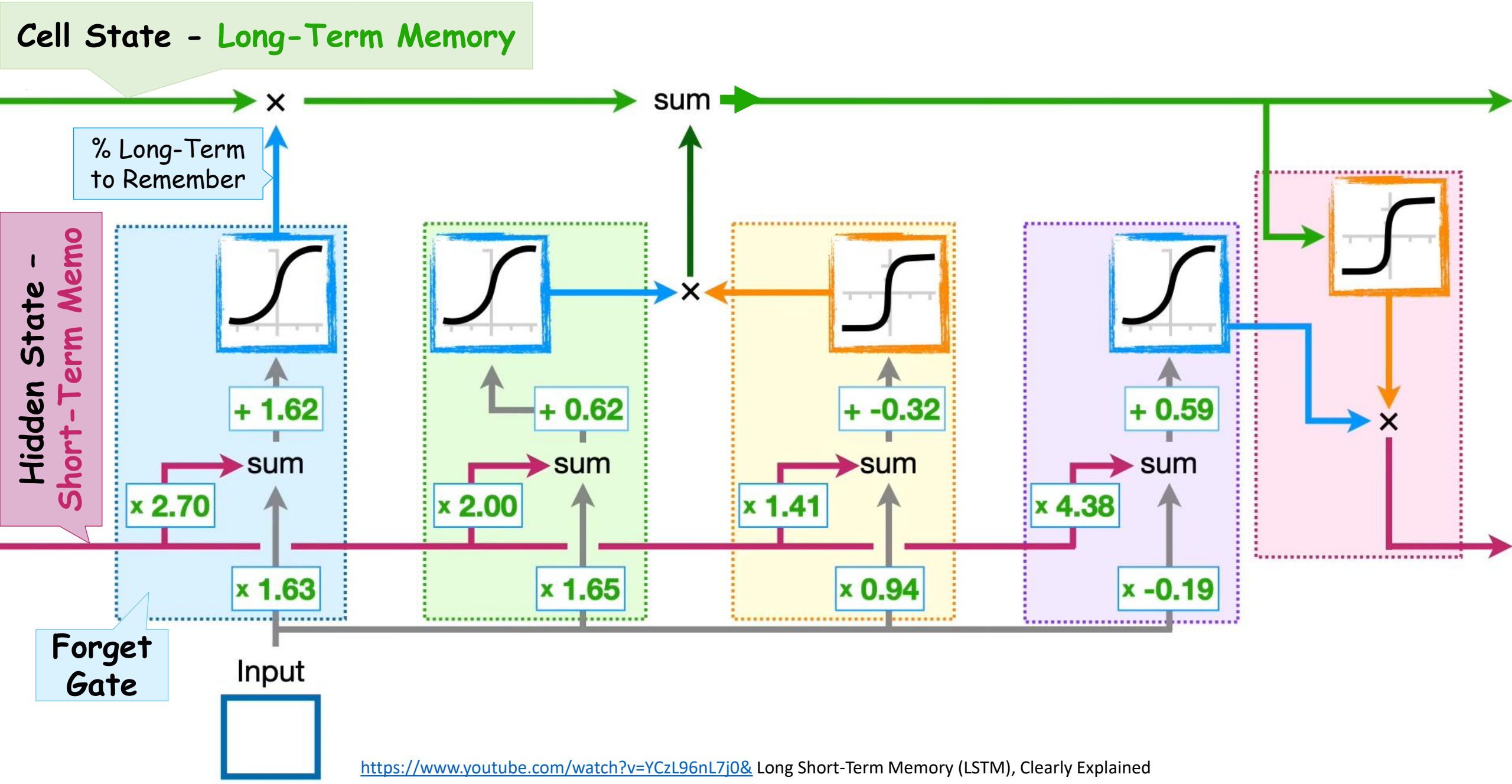
https://www.youtube.com/watch?v=YCzL96nL7j0& Long Short-Term Memory (LSTM), Clearly Explained

Long Short-Term Memory unit (jednostka LSTM)



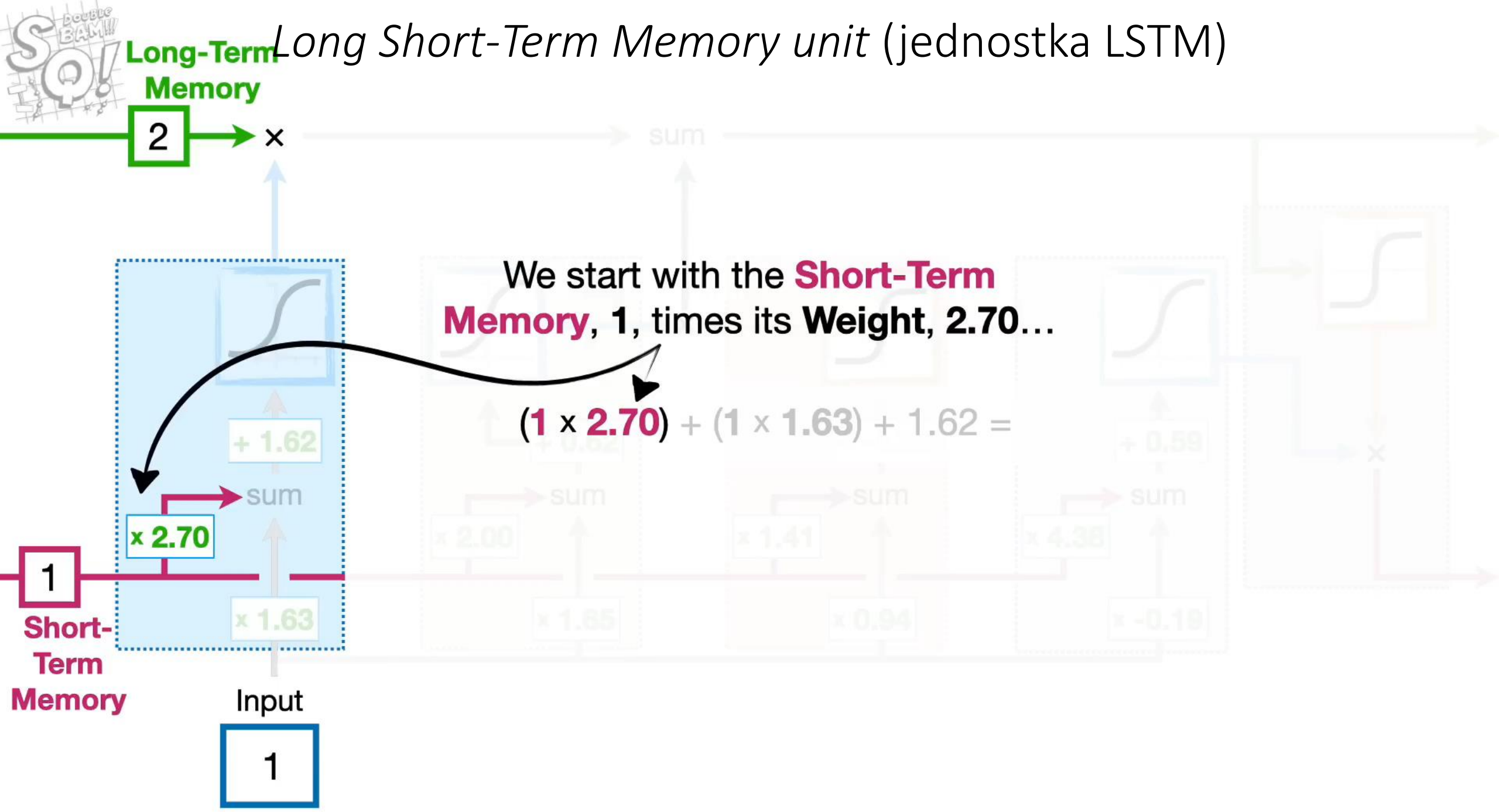
https://www.youtube.com/watch?v=YCzL96nL7j0& Long Short-Term Memory (LSTM), Clearly Explained

Long Short-Term Memory unit (jednostka LSTM)



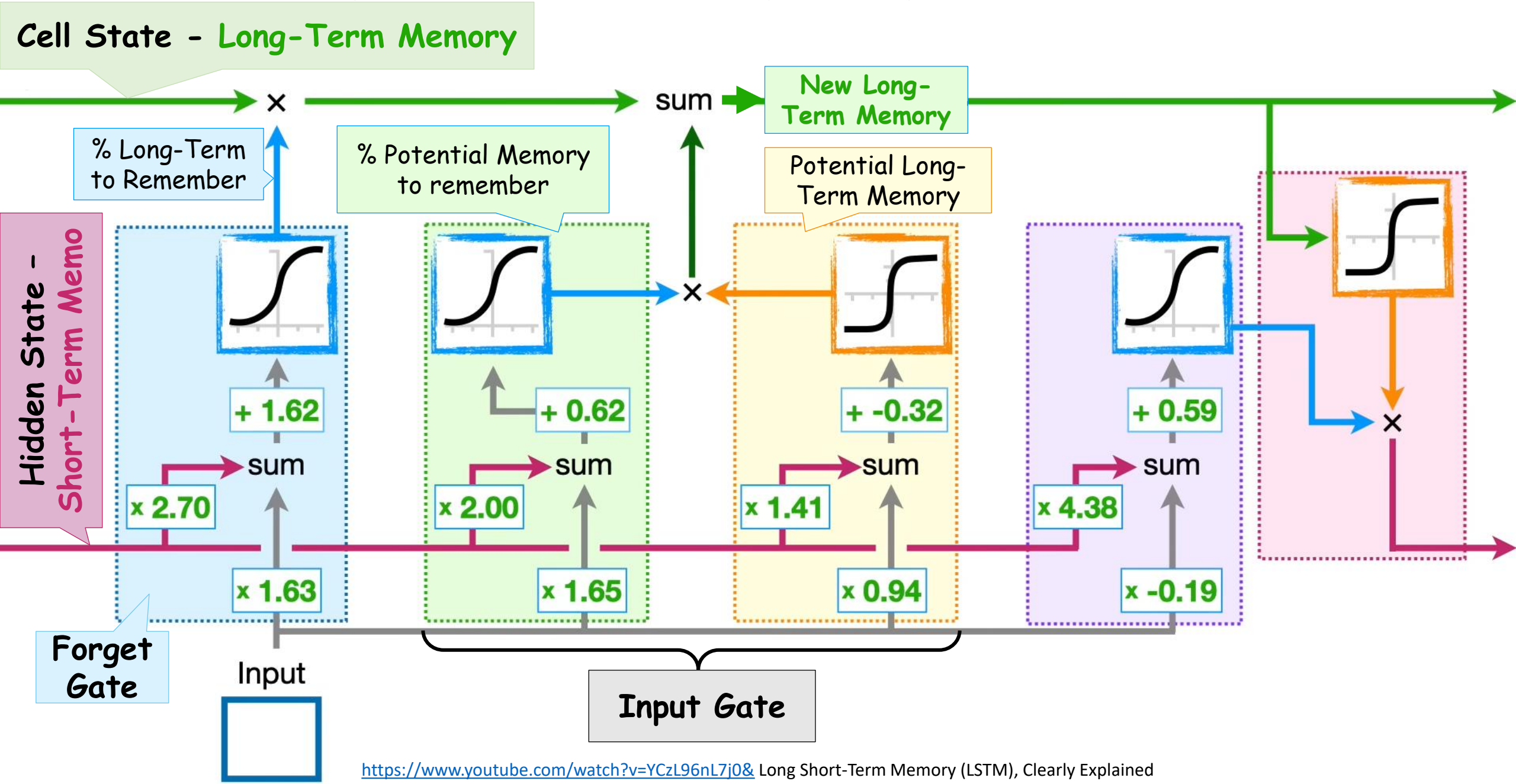
https://www.youtube.com/watch?v=YCzL96nL7j0& Long Short-Term Memory (LSTM), Clearly Explained

Long Short-Term Memory unit (jednostka LSTM)



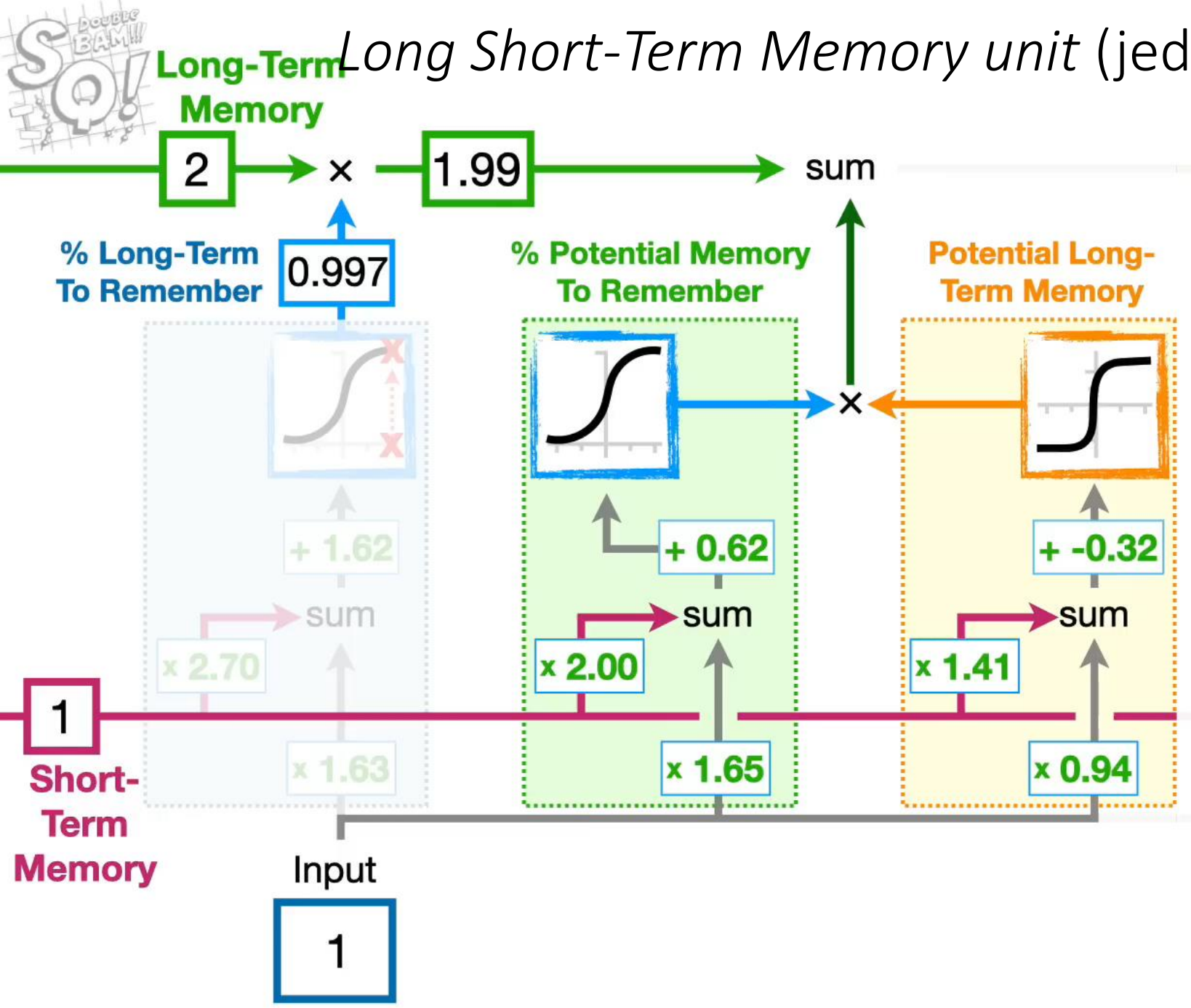
https://www.youtube.com/watch?v=YCzL96nL7j0& Long Short-Term Memory (LSTM), Clearly Explained

Long Short-Term Memory unit (jednostka LSTM)



https://www.youtube.com/watch?v=YCzL96nL7j0& Long Short-Term Memory (LSTM), Clearly Explained

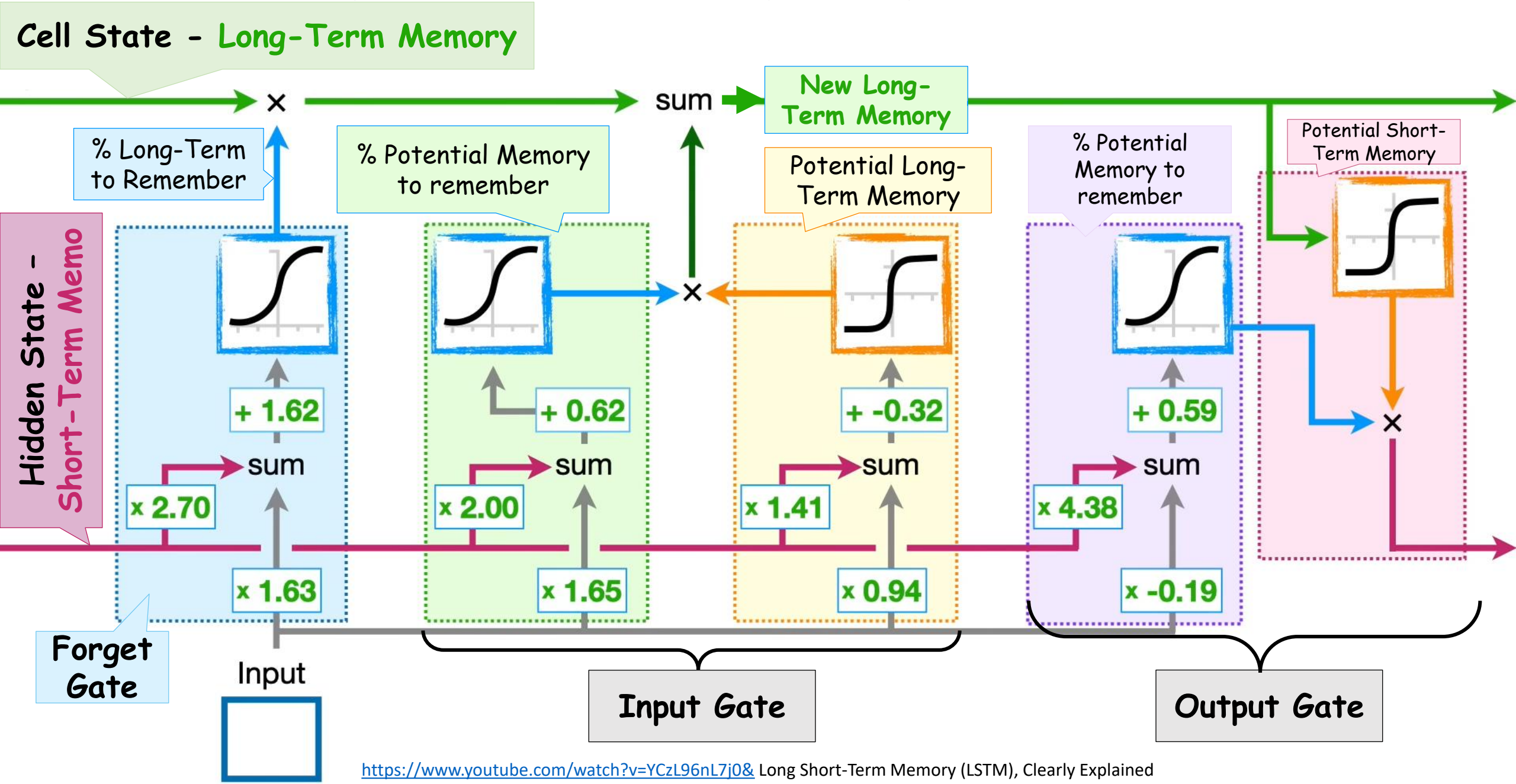
Long Short-Term Memory unit (jednostka LSTM)



So let's plug the numbers in and do the math to see how a **Potential Memory** is created and how much of it is added to the **Long-Term Memory**.

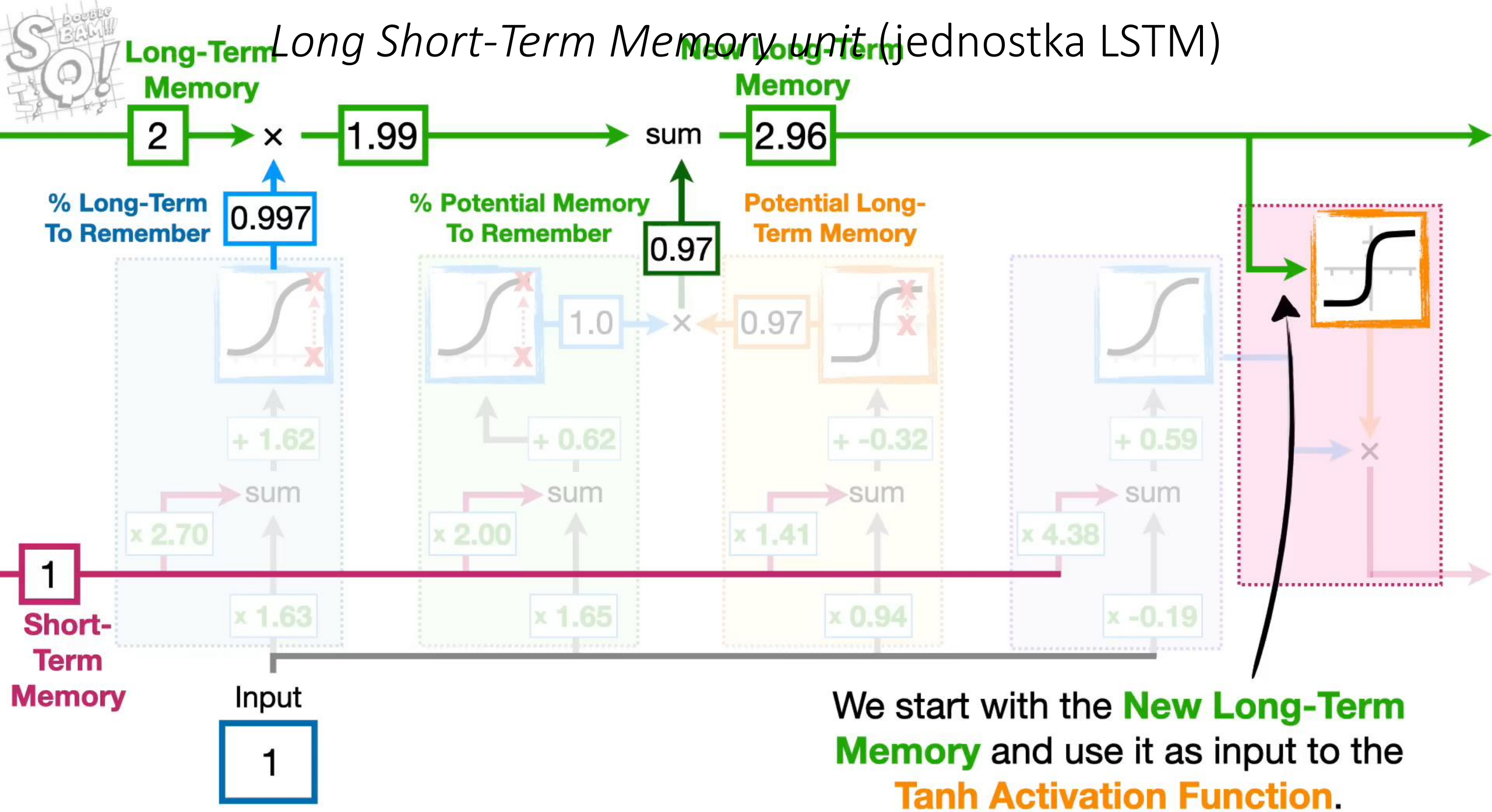
https://www.youtube.com/watch?v=YCzL96nL7j0& Long Short-Term Memory (LSTM), Clearly Explained

Long Short-Term Memory unit (jednostka LSTM)



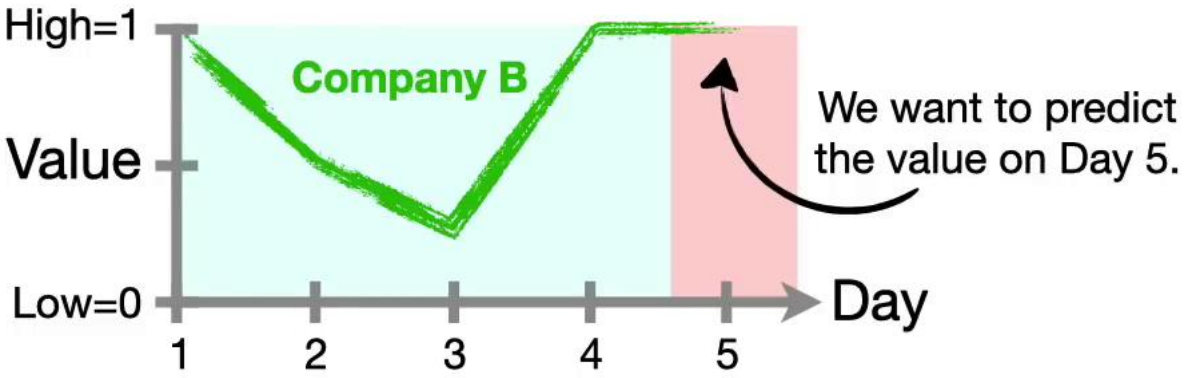
https://www.youtube.com/watch?v=YCzL96nL7j0& Long Short-Term Memory (LSTM), Clearly Explained

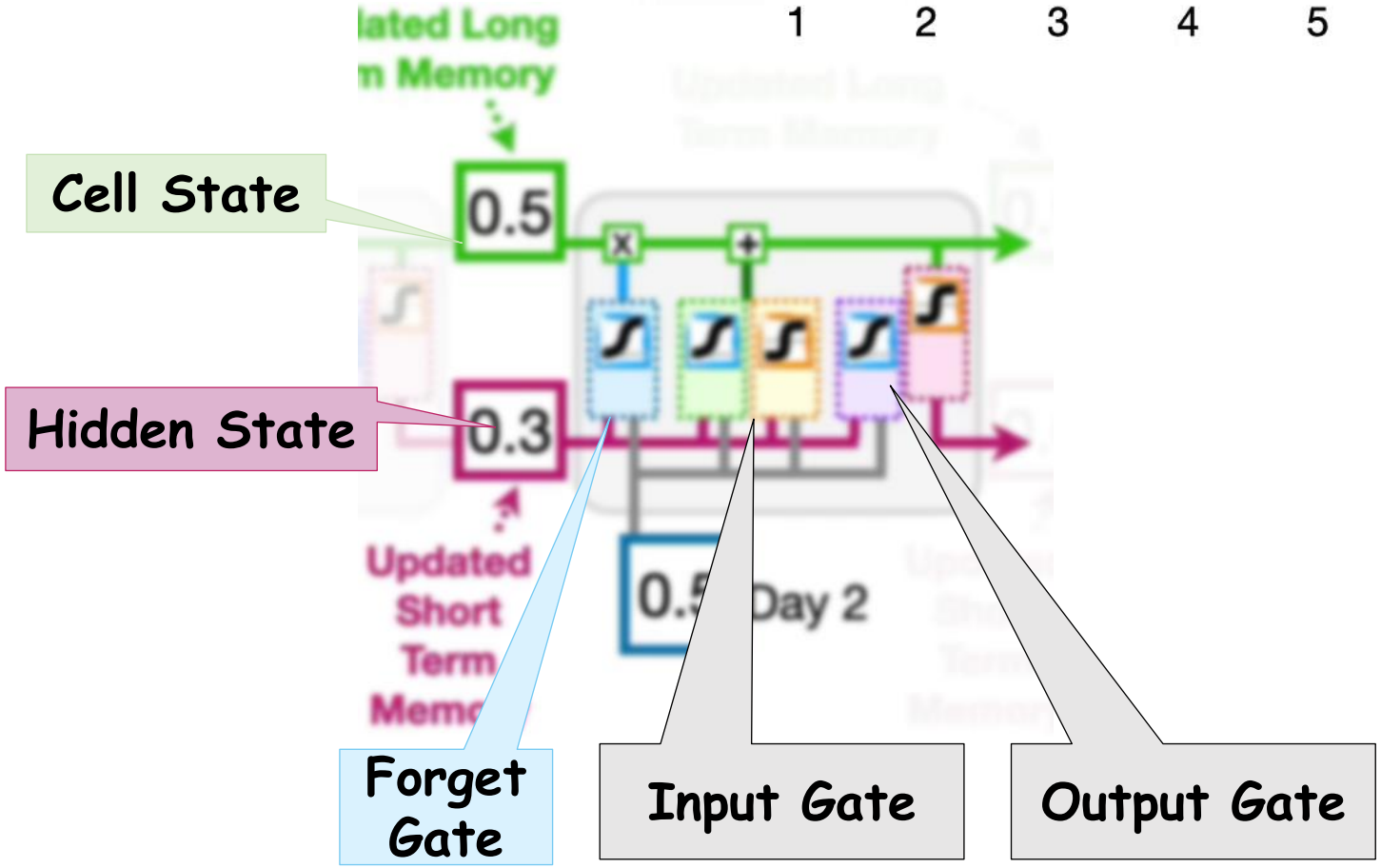
Long Short-Term Memory unit (jednostka LSTM)



https://www.youtube.com/watch?v=YCzL96nL7j0& Long Short-Term Memory (LSTM), Clearly Explained

Long Short-Term Memory (LSTM)





https://www.youtube.com/watch?v=YCzL96nL7j0& Long Short-Term Memory (LSTM), Clearly Explained

Projekt 6

Zaprogramuj sieć neuronową z warstwami LSTM przewidującą przebieg danych sekwencyjnych na następny dzień. Jako dane sekwencyjne możesz wykorzystać dane z https://stooq.pl/ :

1. wybieramy interesejący nas instrument finansowy/kursy walut/notowania akcji,
2. klikamy po lewo „dane historyczne”,
3. wybieramy zakres dat jaki nas interesuje, interwał czasowy, wyłączenia,
4. klikamy „Pobierz dane w pliku csv...”

Pobrane dane podziel w stosunku: około 60-70 % na dane treningowe, pozostałe 40-30% dane testowe.

Wygeneruj wykres przewidywania sieci neuronowej dla danych treningowych i dla danych testowych.

Keras

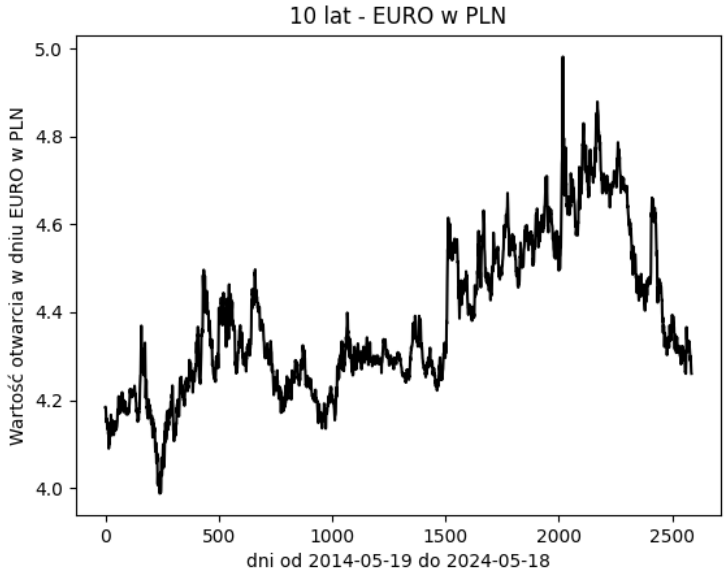
Google colab

Projekt 6 - przykładowy wynik korzystając z sieci LSTM

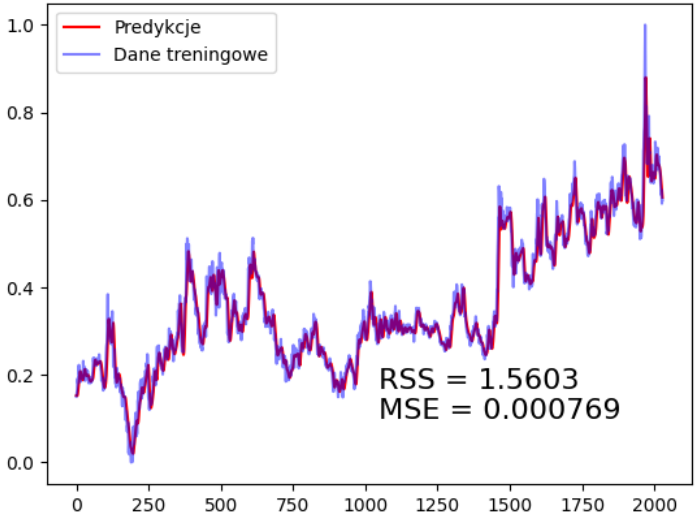
Źródła inspiracji : https://keras.io/2.16/api/layers/recurrent_layers/lstm/

public/mmajew/MIW/10/ *02_lstm_EURtoPLN .py / .ipynb*

Ciąg czasowy



Dane treningowe



Dane testowe