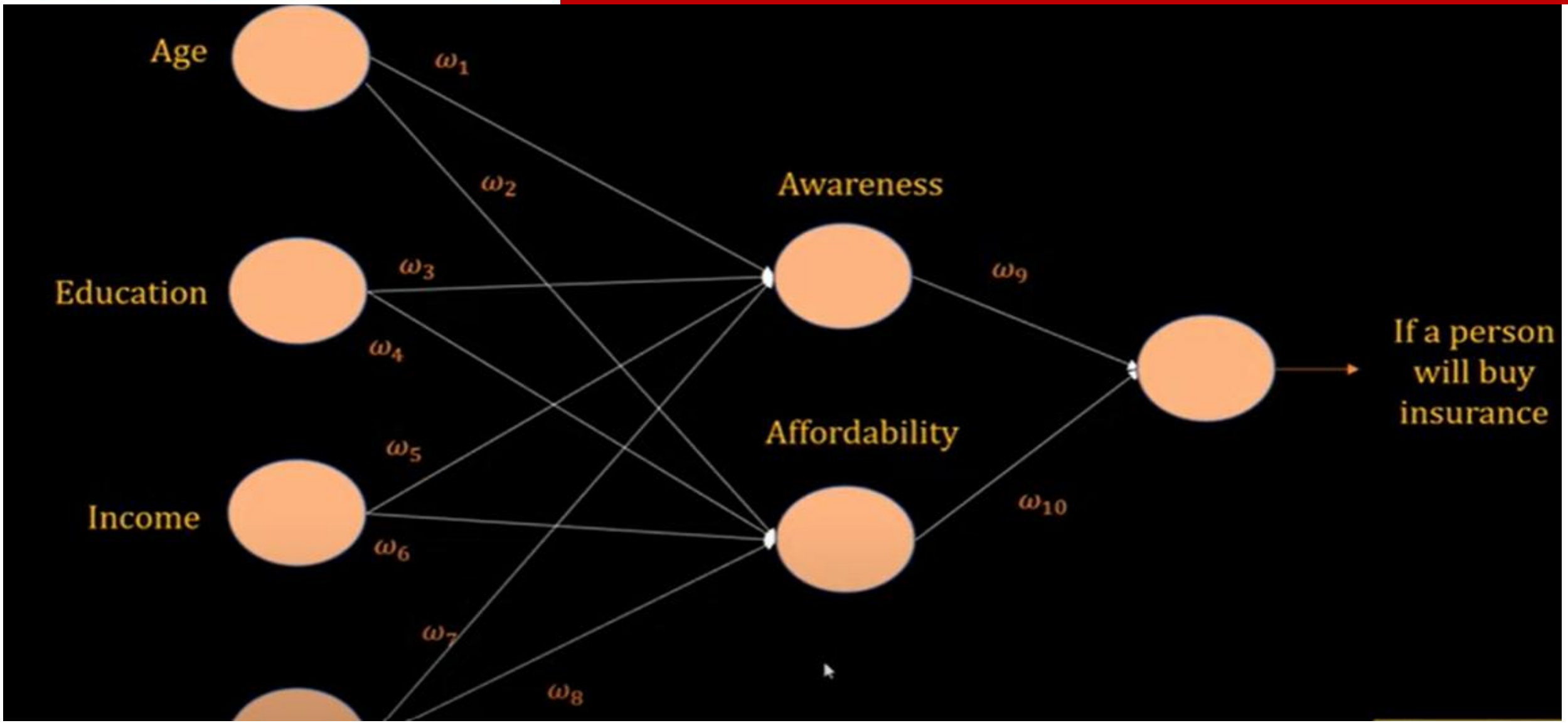
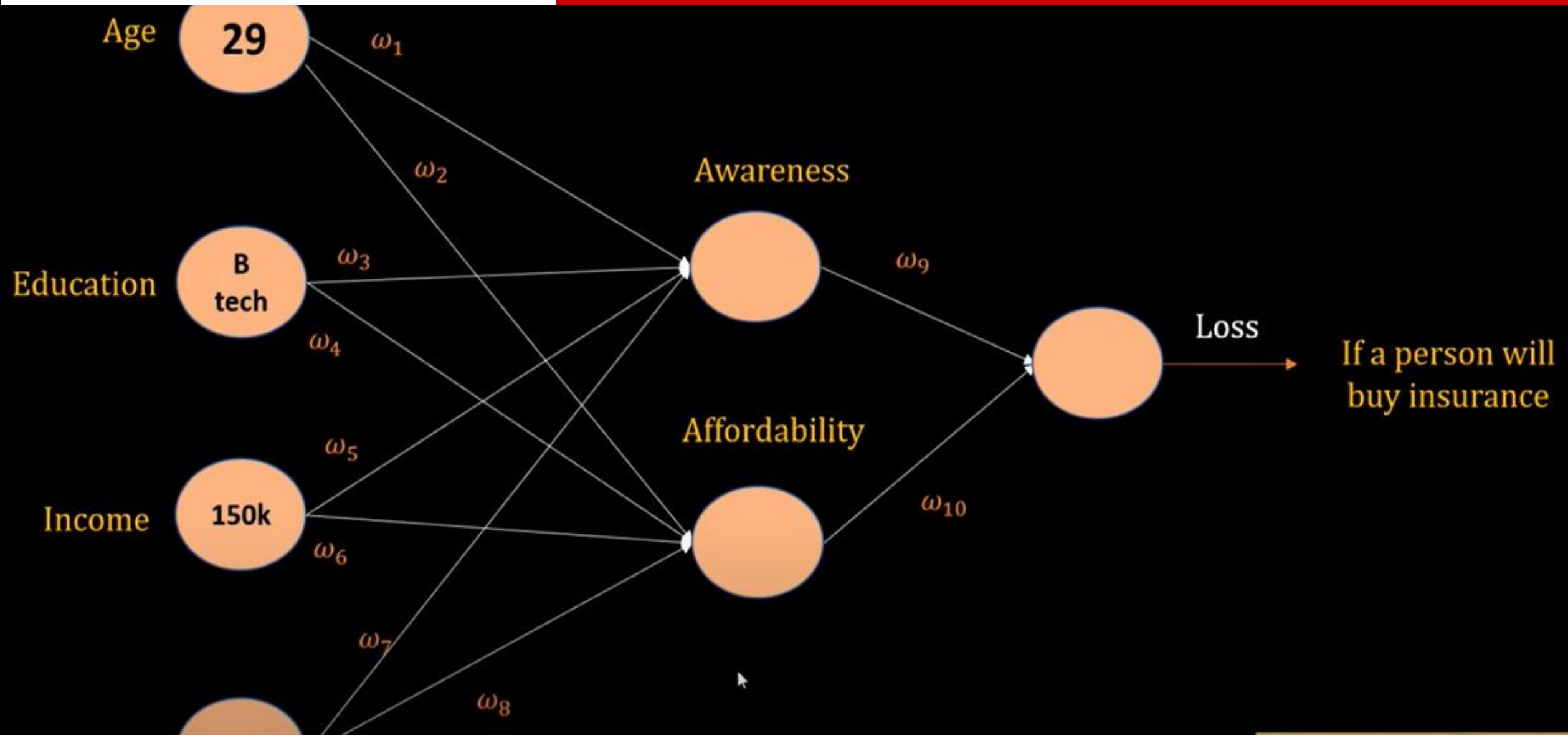
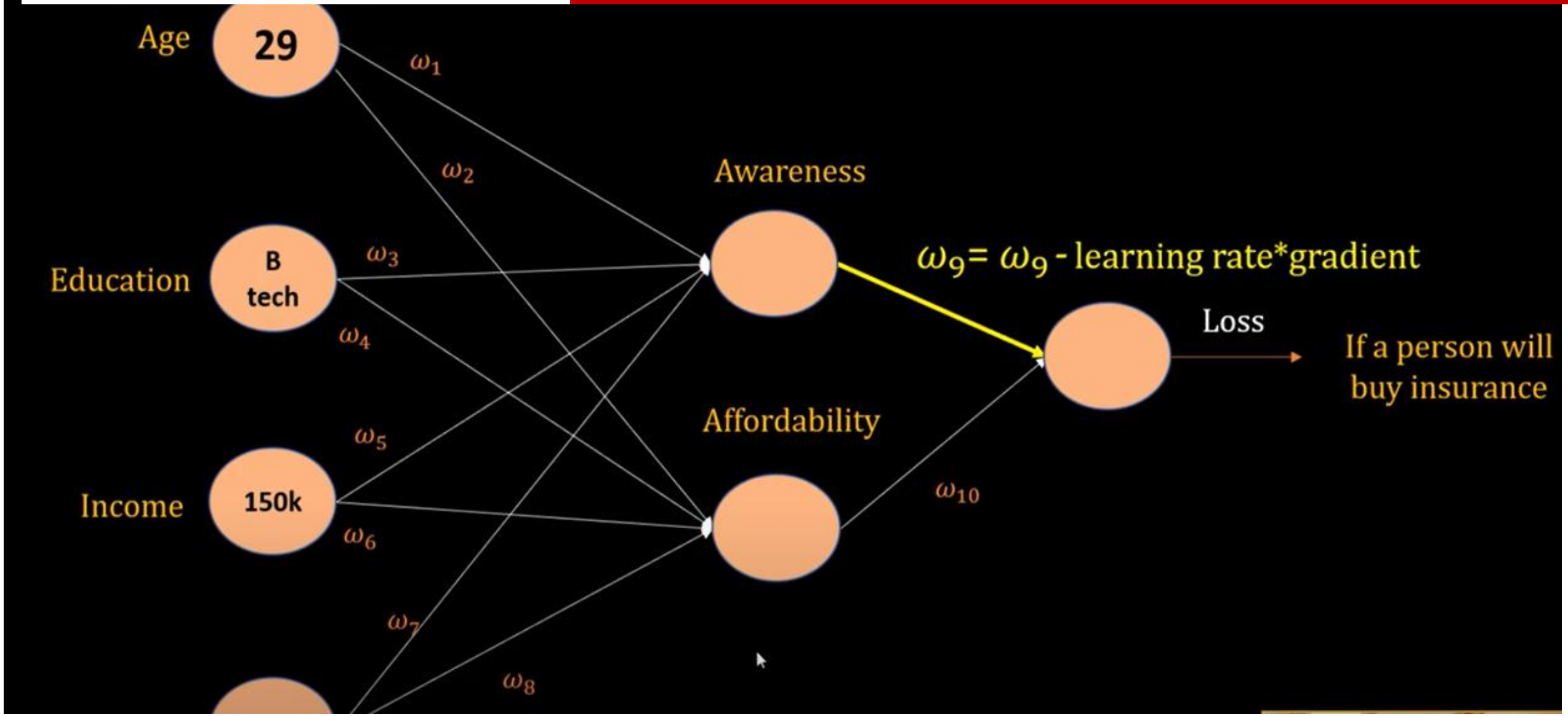
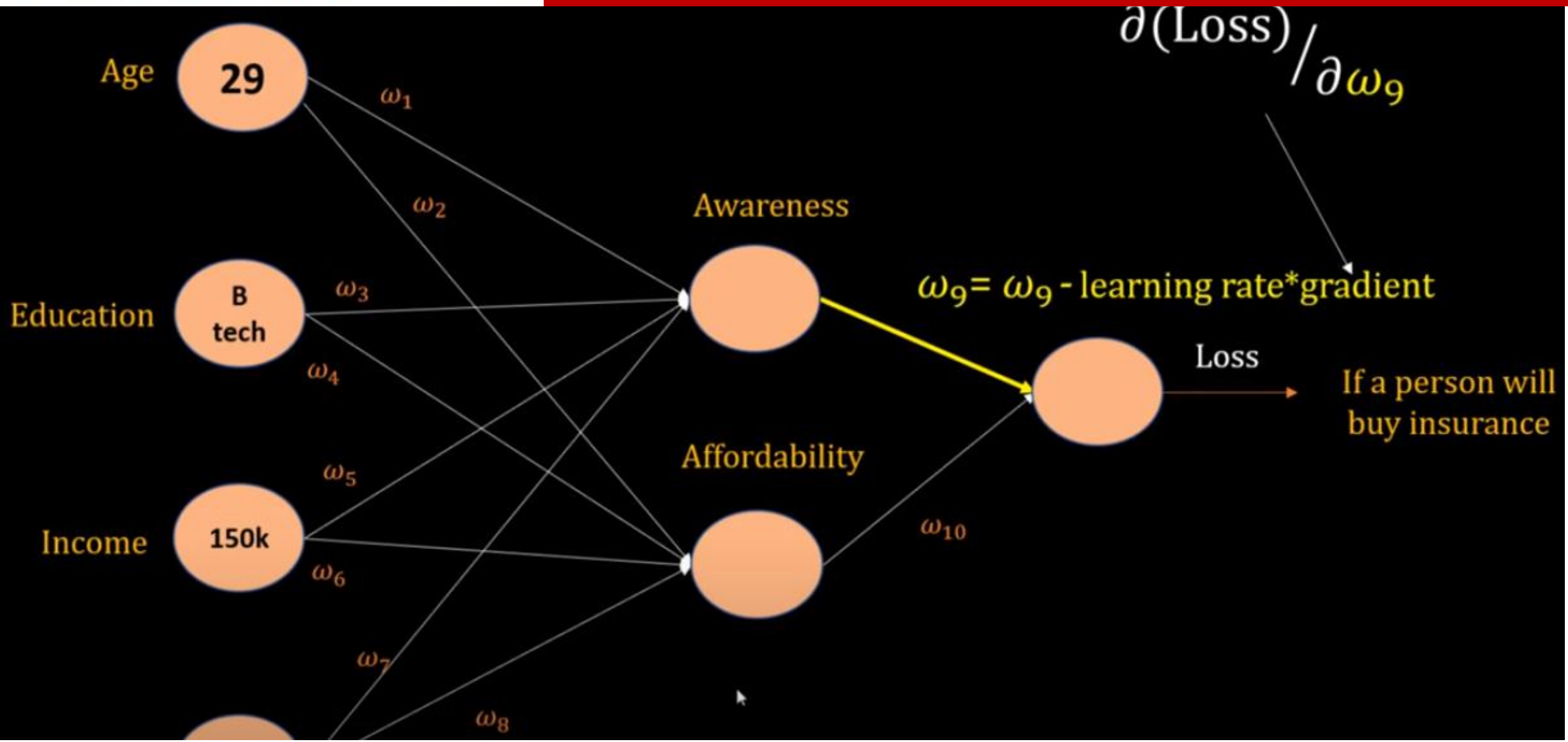
Long Short-Term Memory (LSTM)

Vanishing Gradient Issue

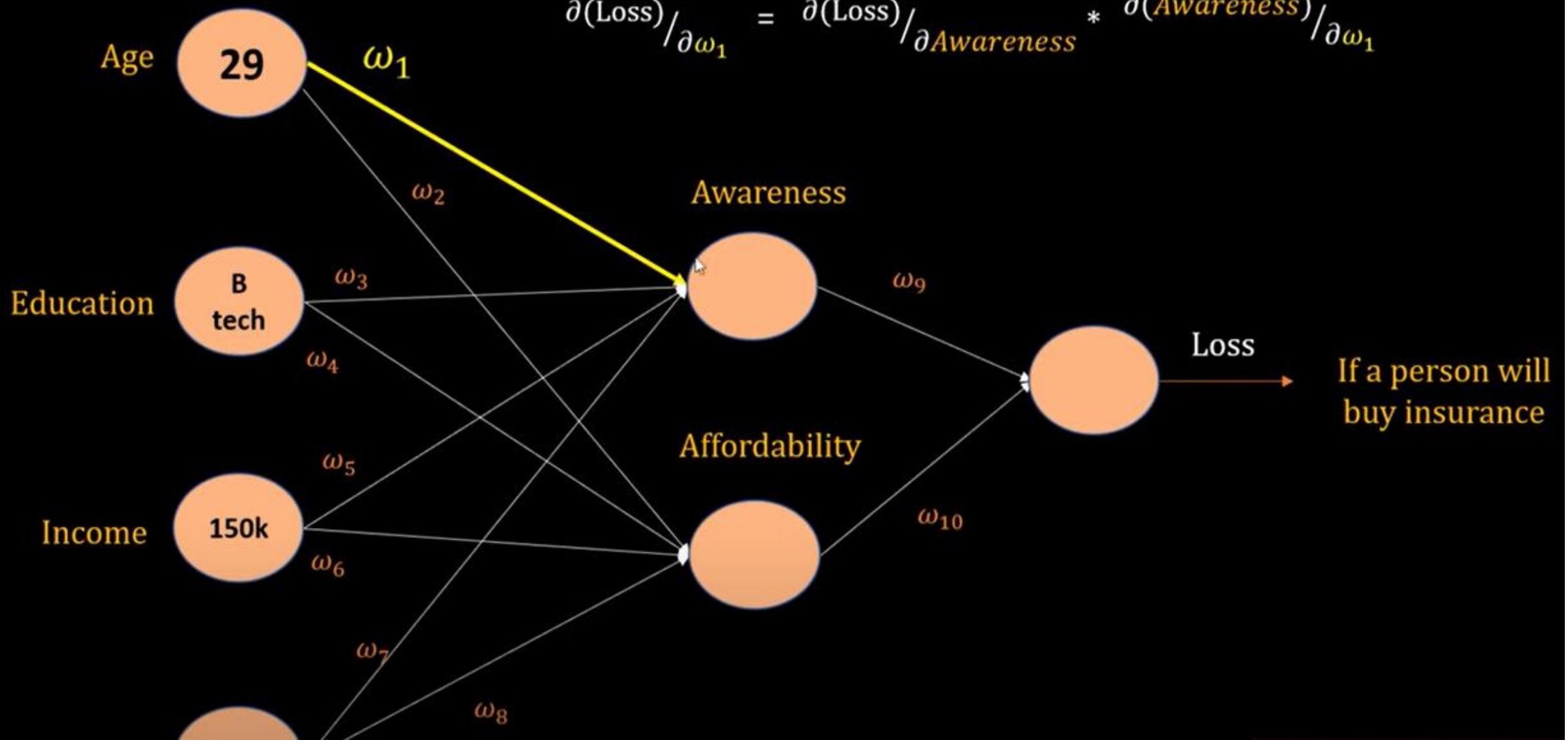

Age
Education
Income
Awareness
Affordability
ω_1
ω_2
ω_3
ω_4
ω_5
ω_6
ω_7
ω_8
ω_9
ω_{10}
If a person will buy insurance


Age
29
Education
B tech
Income
150k
ω_1
ω_2
ω_3
ω_4
ω_5
ω_6
ω_7
ω_8
Awareness
Affordability
ω_9
ω_{10}
Loss
If a person will buy insurance


Age
29
Education
B tech
Income
150k
ω_1
ω_2
ω_3
ω_4
ω_5
ω_6
ω_7
ω_8
Awareness
Affordability
$\omega_9 = \omega_9$ - learning rate*gradient
ω_{10}
Loss
If a person will buy insurance


∂(Loss)/∂ω9
Age
29
ω1
ω2
Awareness
Education
B tech
ω3
ω4
ω9= ω9 - learning rate*gradient
Loss
If a person will buy insurance
ω5
Affordability
Income
150k
ω6
ω10
ω7
ω8

$$\partial(\text{Loss})/\partial\omega_1 = \partial(\text{Loss})/\partial\text{Awareness} * \partial(\text{Awareness})/\partial\omega_1$$



$$\partial(\text{Loss})/\partial\omega_1 = \partial(\text{Loss})/\partial Awareness * \partial(Awareness)/\partial\omega_1$$

$$\partial(\text{Loss})/\partial\omega_1 = \partial(\text{Loss})/\partial Awareness * \partial(Awareness)/\partial\omega_1$$

$$gradient = d1 * d2$$

$$\partial(\text{Loss})/\partial\omega_1 = \partial(\text{Loss})/\partial Awareness * \partial(Awareness)/\partial\omega_1$$

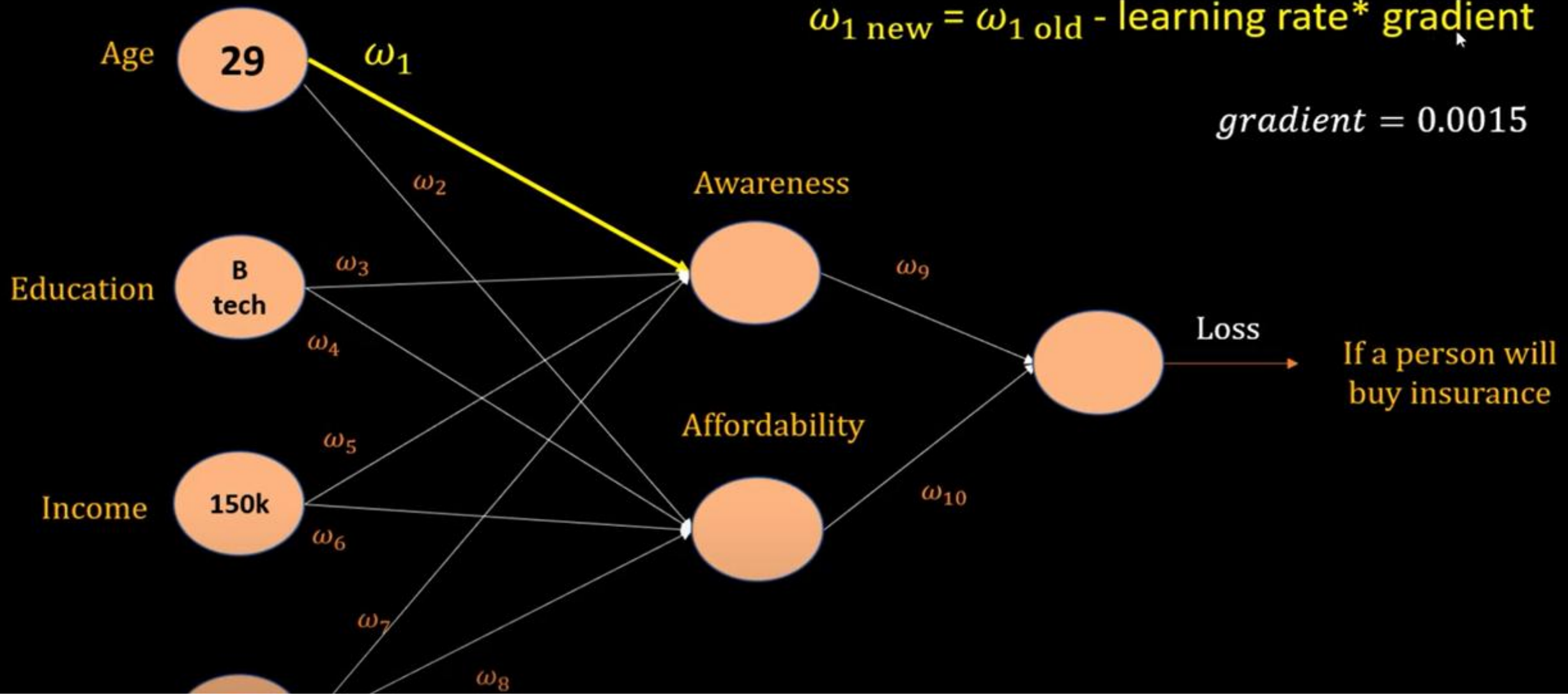
$$gradient = d1 * d2$$

$$gradient = 0.03 * 0.05$$

$$gradient = 0.0015$$

$$\omega_{1\ new} = \omega_{1\ old} - \text{learning rate} * \text{gradient}$$

$$gradient = 0.0015$$



As number of hidden layers grow, gradient becomes very small and weights will hardly change . This will hamper the learning process.

Vanishing Gradients

When individual derivatives are large, the final derivate will also become huge and weights would change drastically.

Exploding Gradients

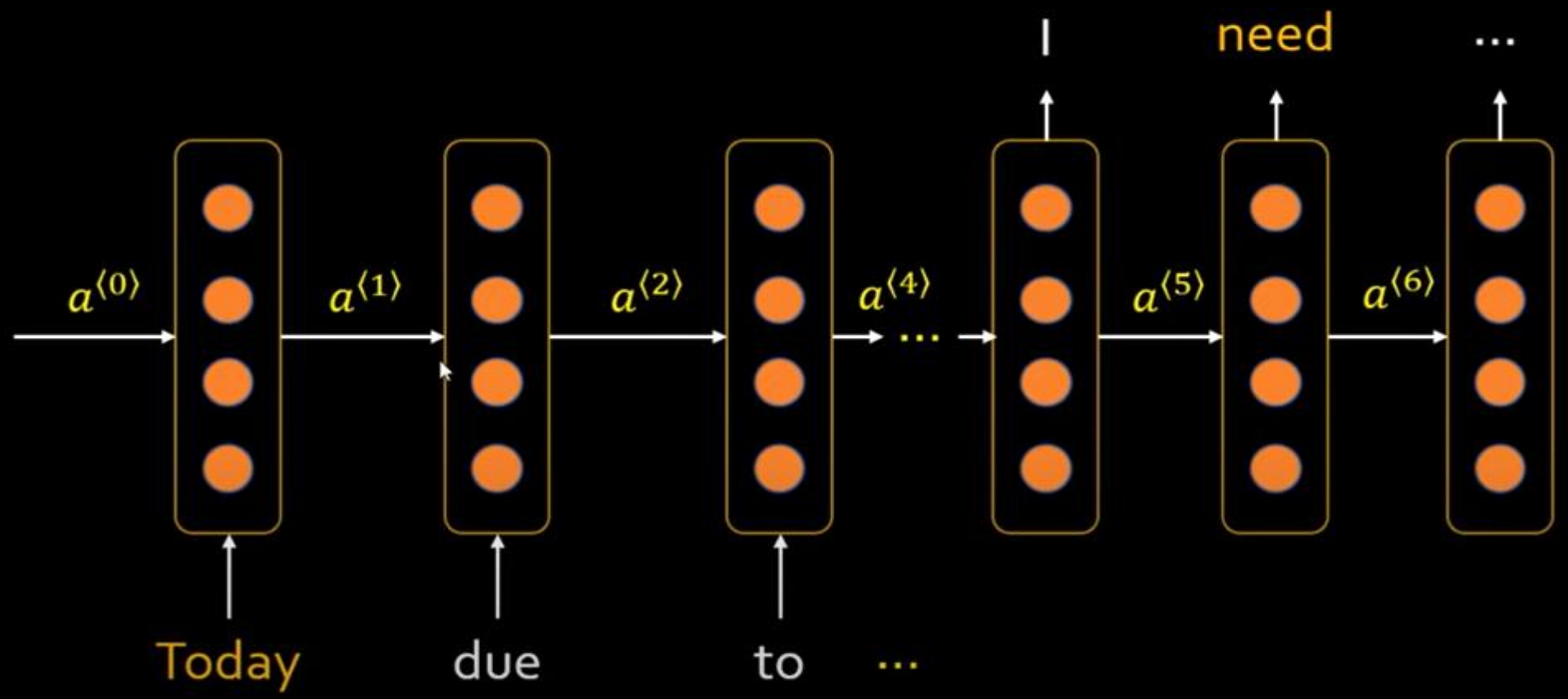
$$gradient = d1 * d2 * d3 * d4 * \ldots * dn$$

Vanishing gradient problem is more prominent in very deep neural networks.

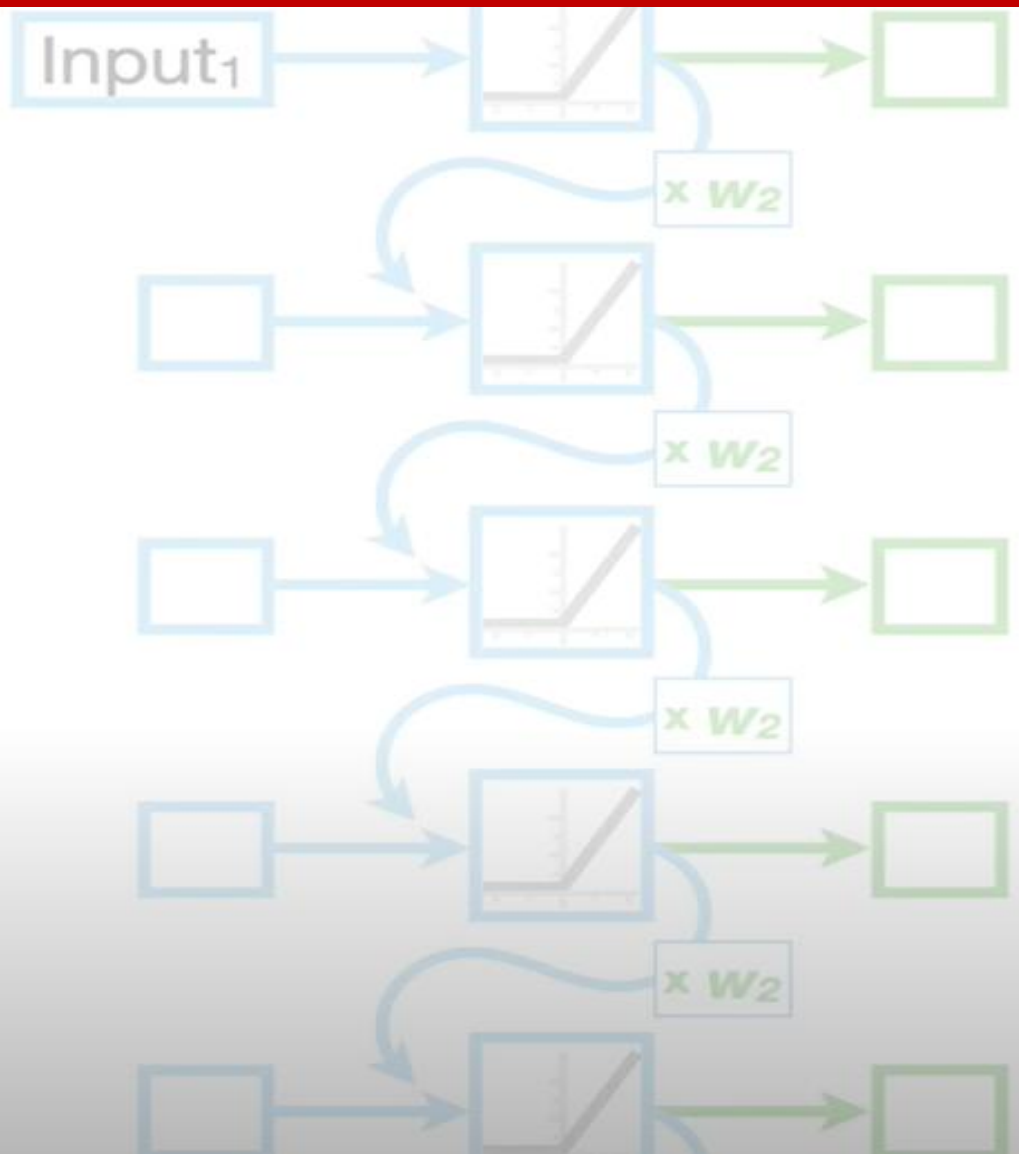
Today, due to my current job situation and family conditions, I need to take a loan.

Last year, due to my current job situation and family conditions, I had to take a loan.

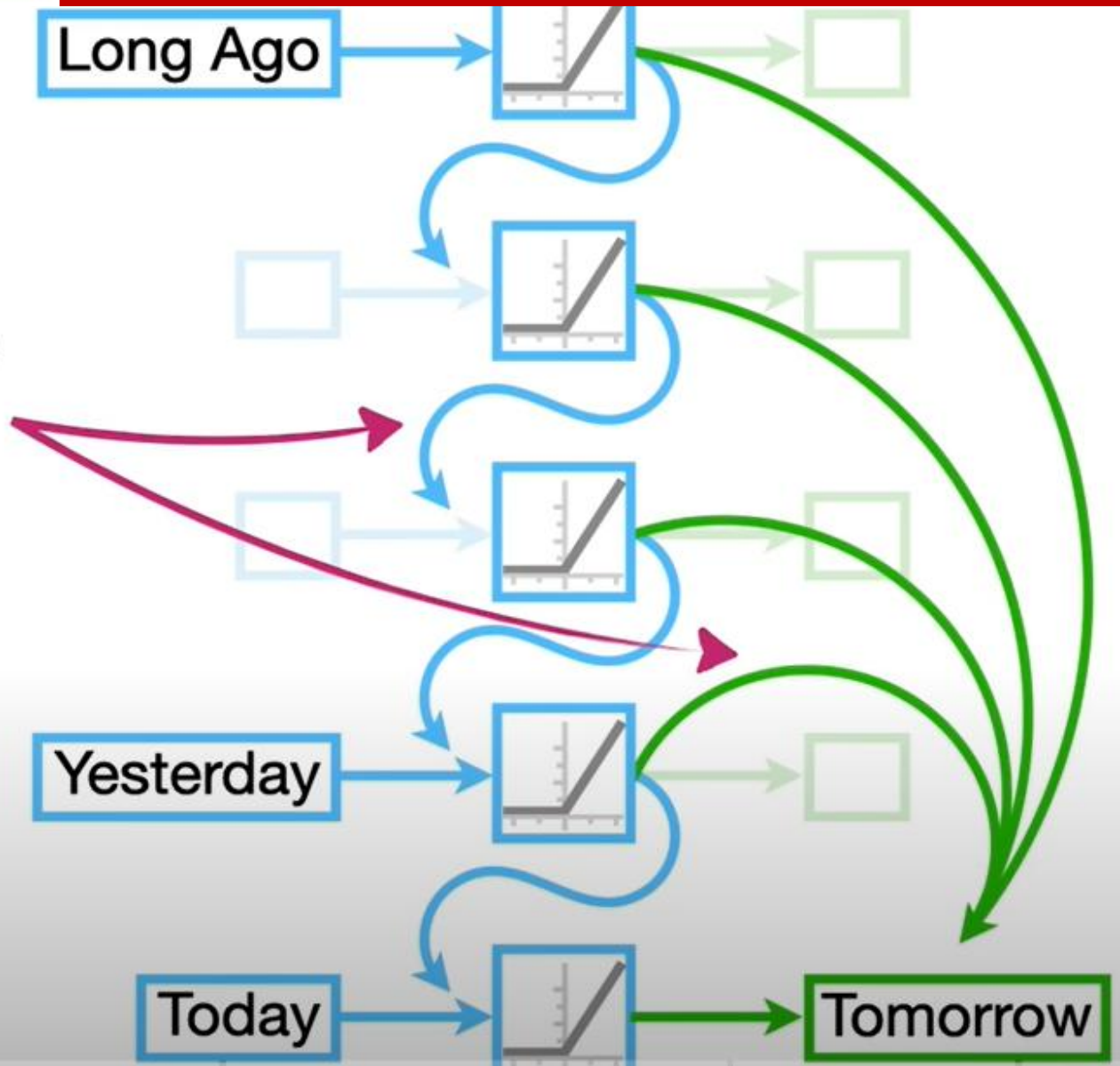
Today, due to my current job situation and family conditions, I need to take a loan.

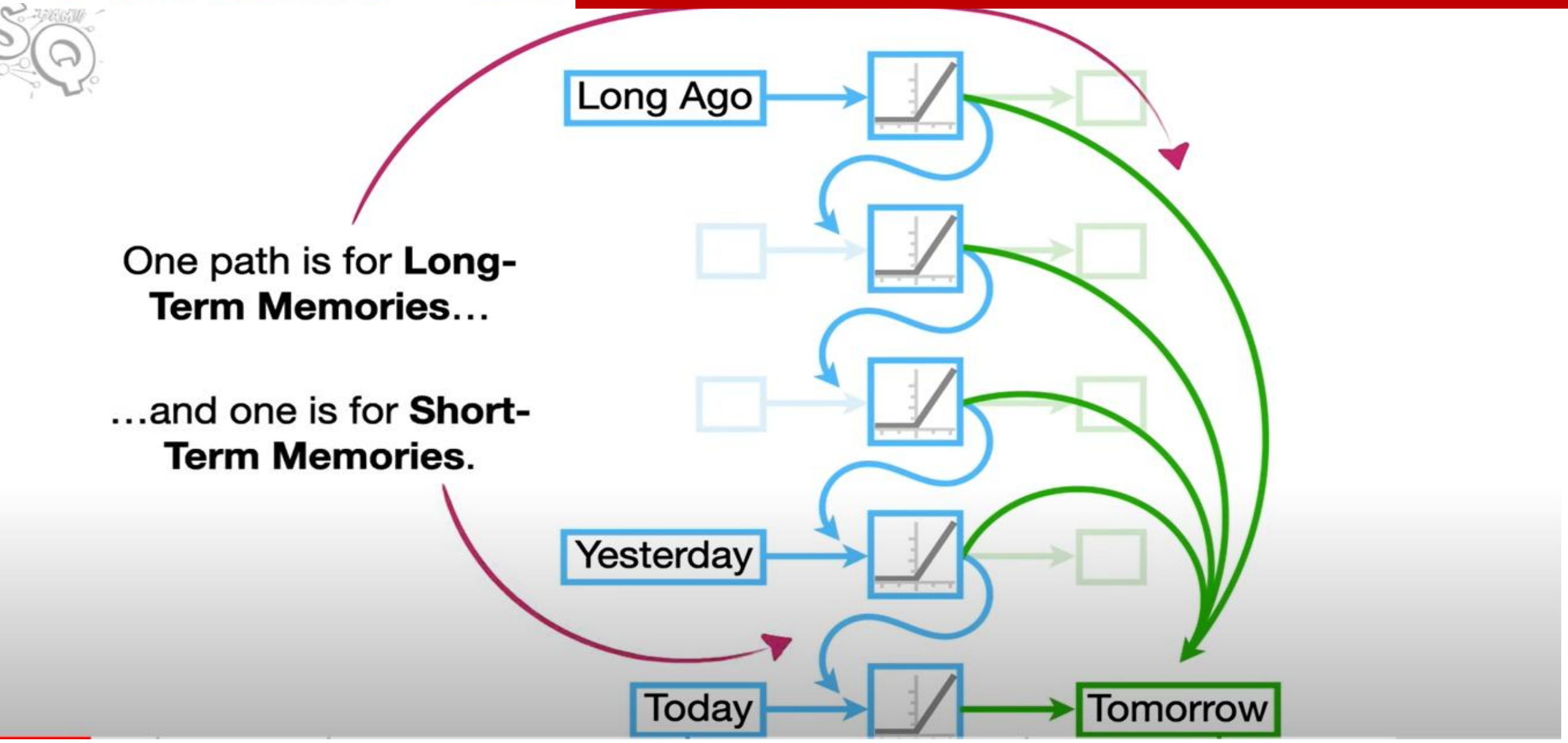


So, today, we're going to talk about **Long Short-Term Memory (LSTM)**, which is a type of **Recurrent Neural Network** that is designed to avoid the **exploding/ vanishing** gradient problem.

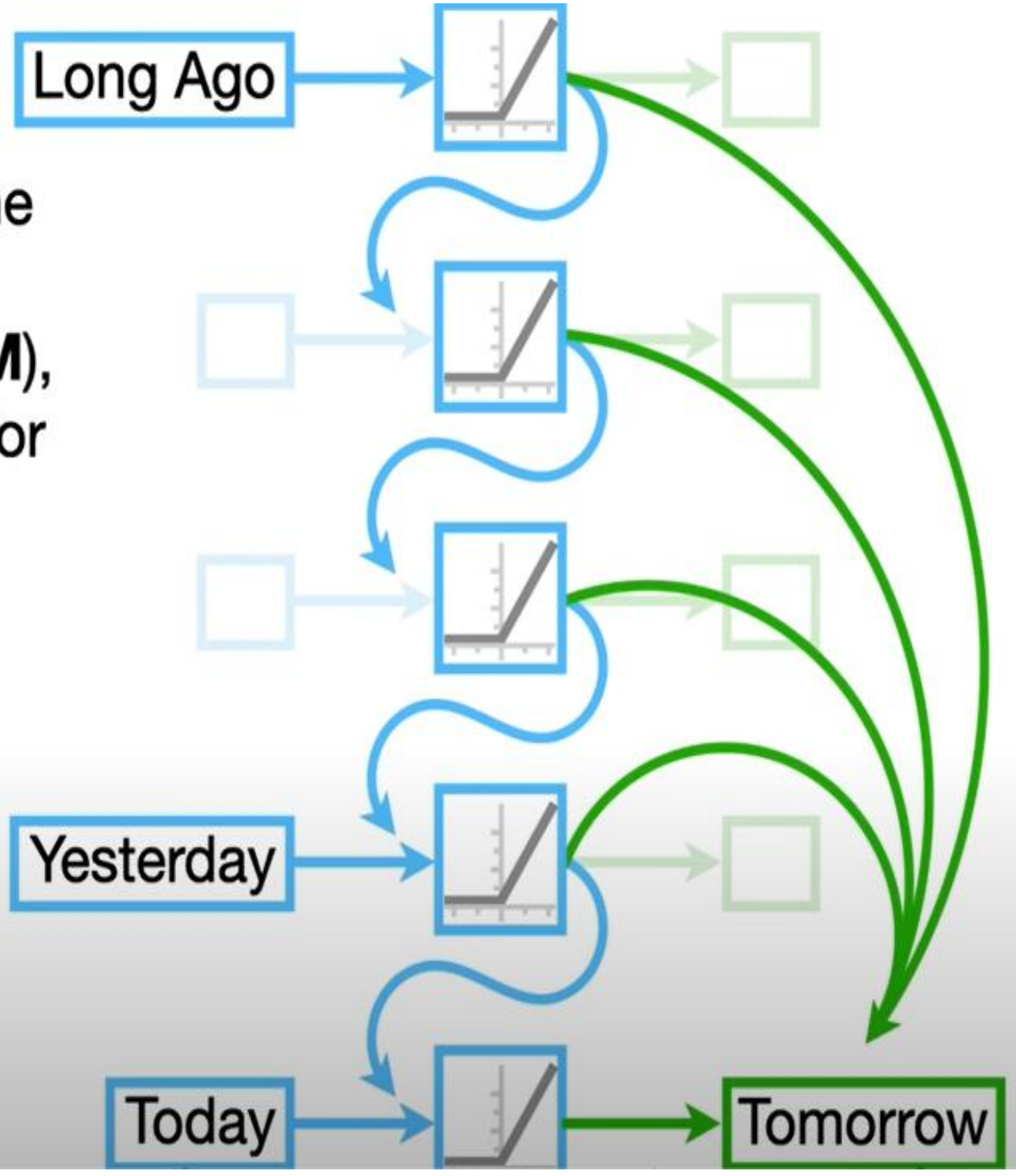


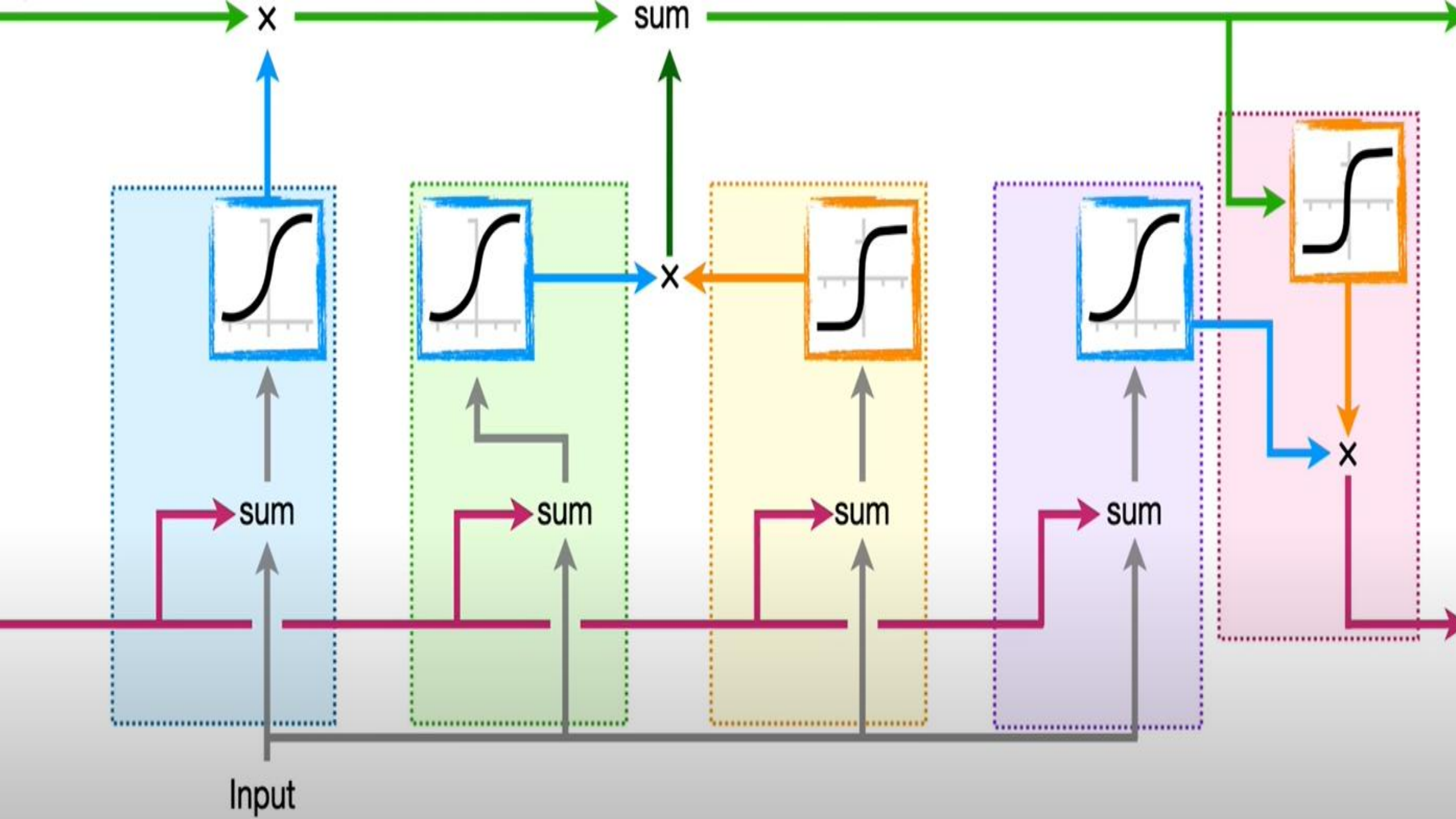
...**Long Short-Term Memory** uses two separate paths to make predictions about tomorrow.



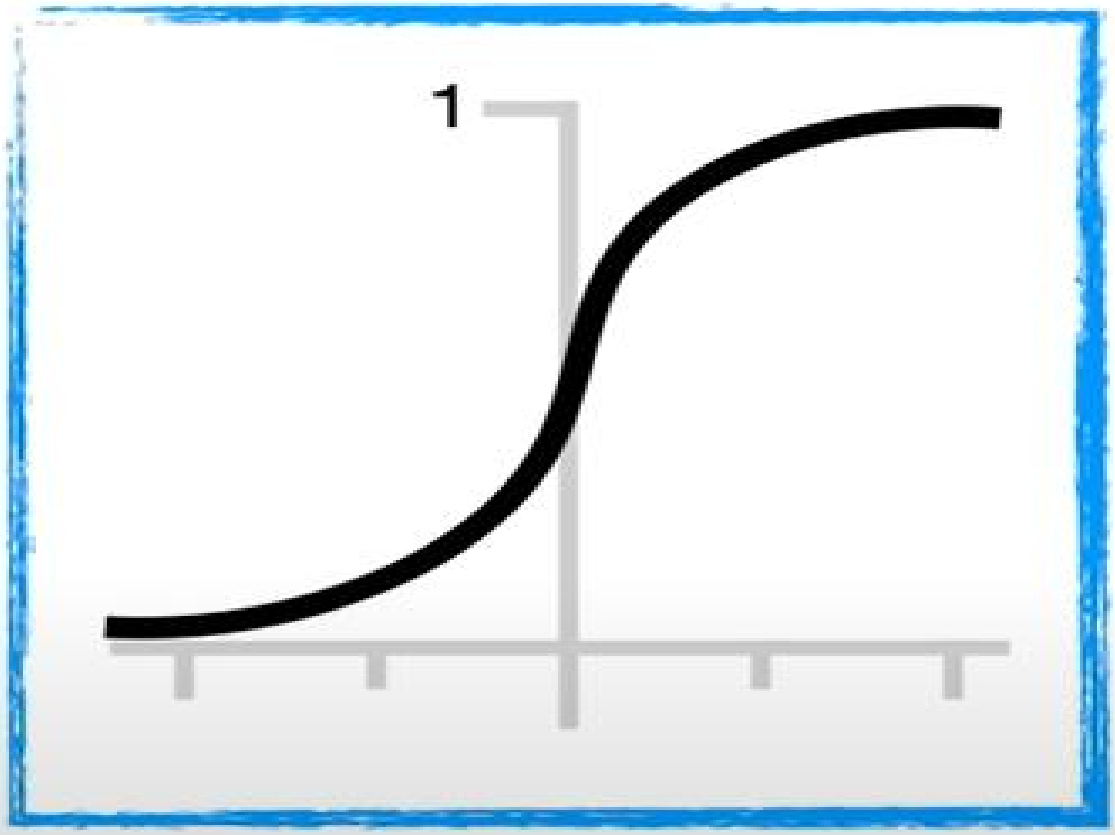

One path is for **Long-Term Memories**…
…and one is for **Short-Term Memories**.
Long Ago
Yesterday
Today
Tomorrow

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

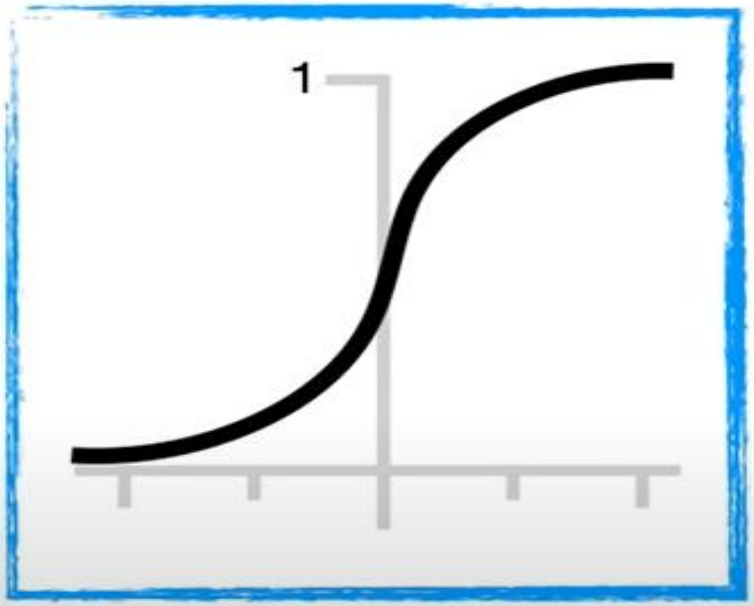



x
sum
x
sum
sum
sum
sum
x
Input

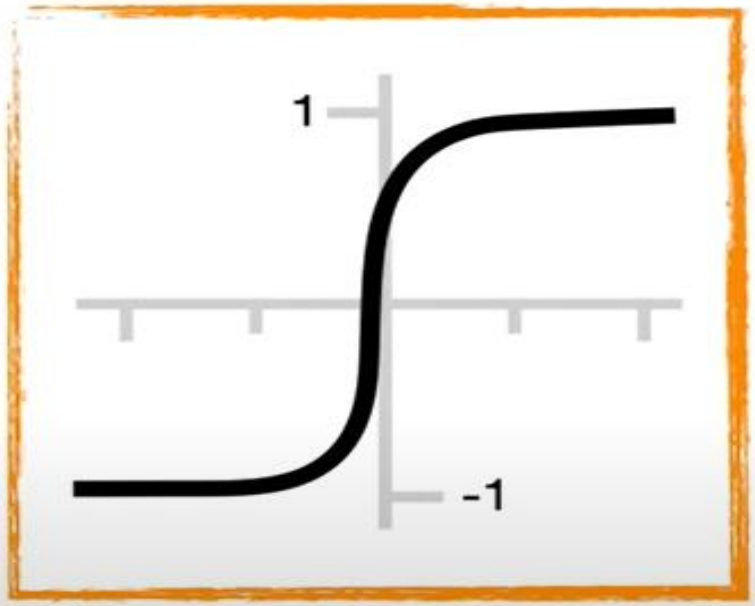
In a nutshell, the **Sigmoid Activation Function** takes any x-axis coordinate…

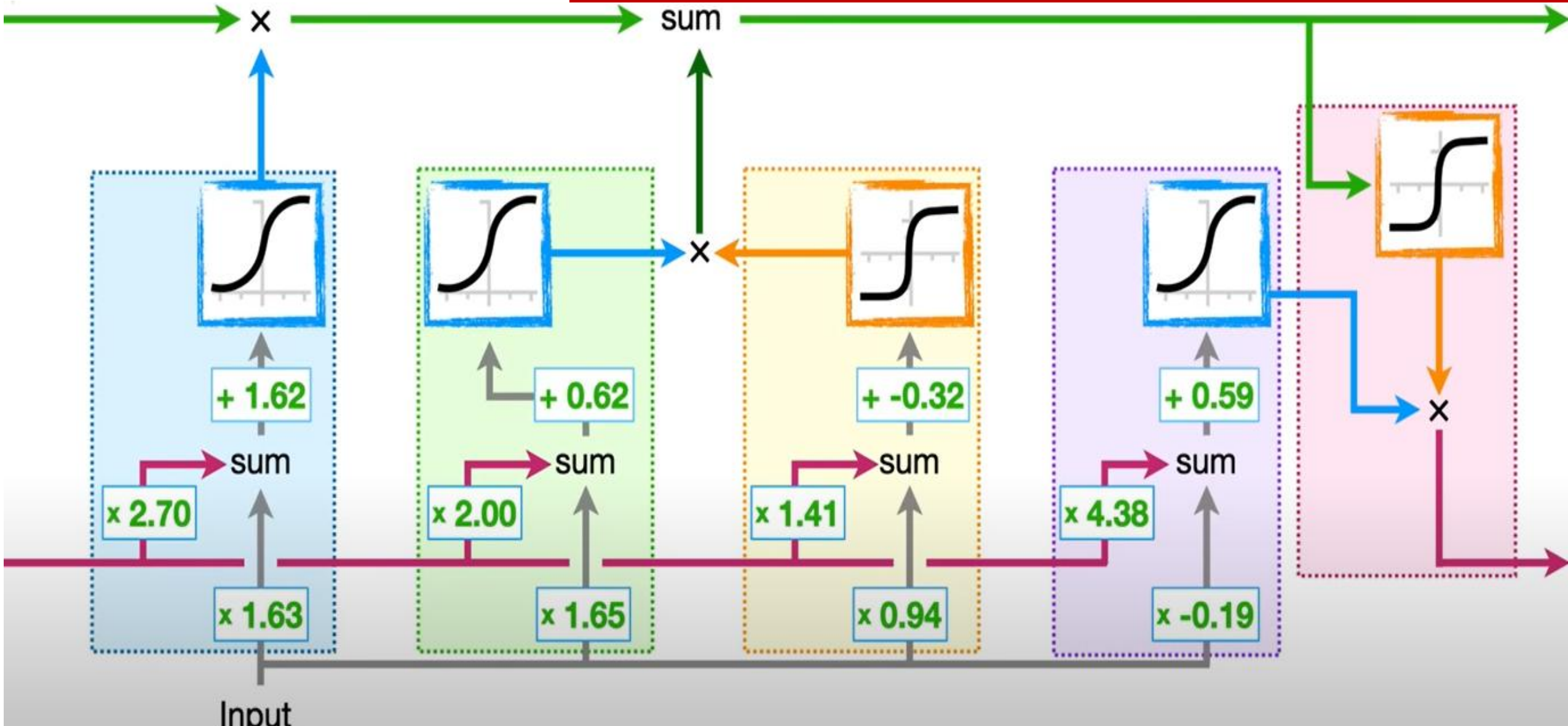


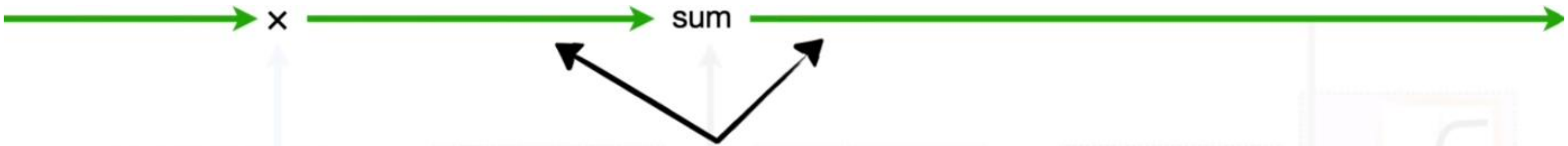
So, now that we know that the **Sigmoid Activation Function** turns any input into a number between **0** and **1**...



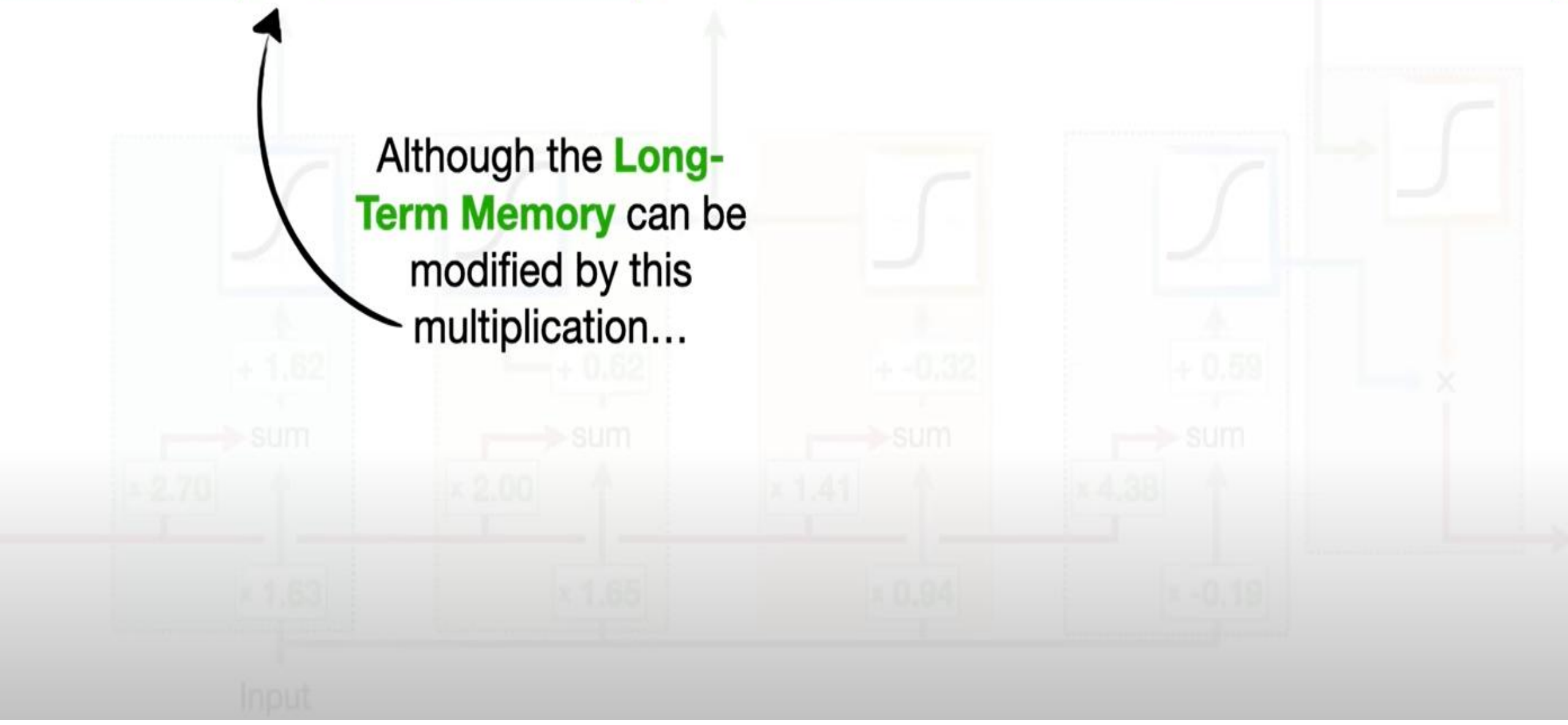
...and the **Tanh Activation Function** turns any input into a number between **-1** and **1**...

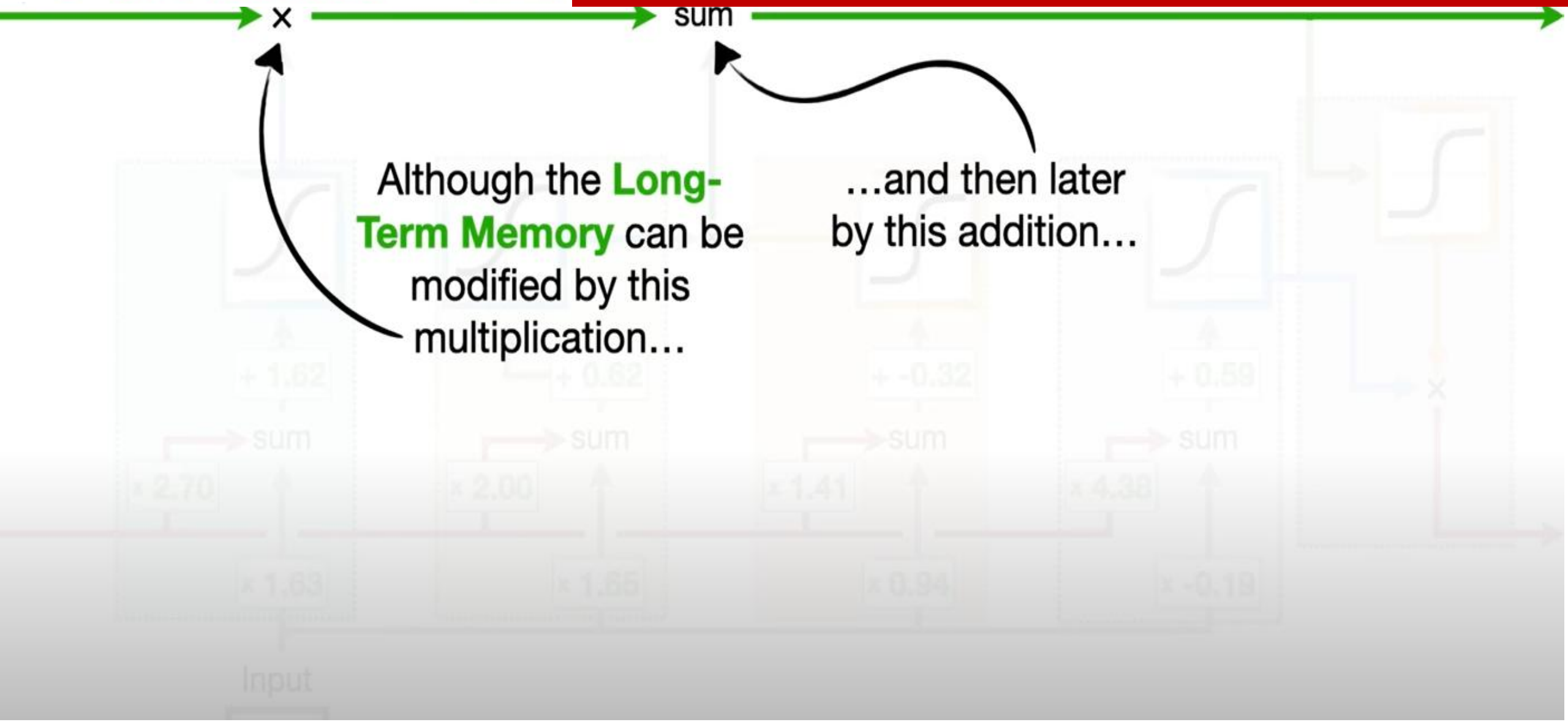


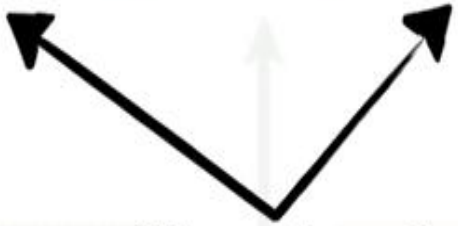

x
sum
+ 1.62
+ 0.62
+ -0.32
+ 0.59
sum
sum
sum
sum
x 2.70
x 2.00
x 1.41
x 4.38
x 1.63
x 1.65
x 0.94
x -0.19
Input



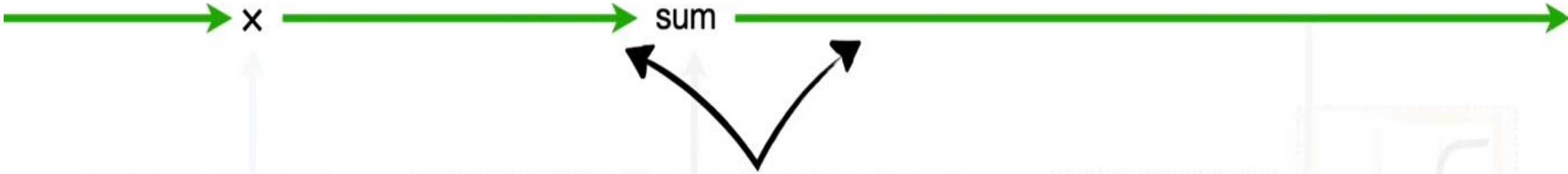
First, this **green line** that runs all the way across the top of the unit is called the **Cell State** and represents the **Long-Term Memory**.


Although the Long-Term Memory can be modified by this multiplication…
+ 1.62
+ 0.62
+ -0.32
+ 0.59
sum
sum
sum
sum
x 2.70
x 2.00
x 1.41
x 4.38
x 1.63
x 1.65
x 0.94
x -0.19
Input

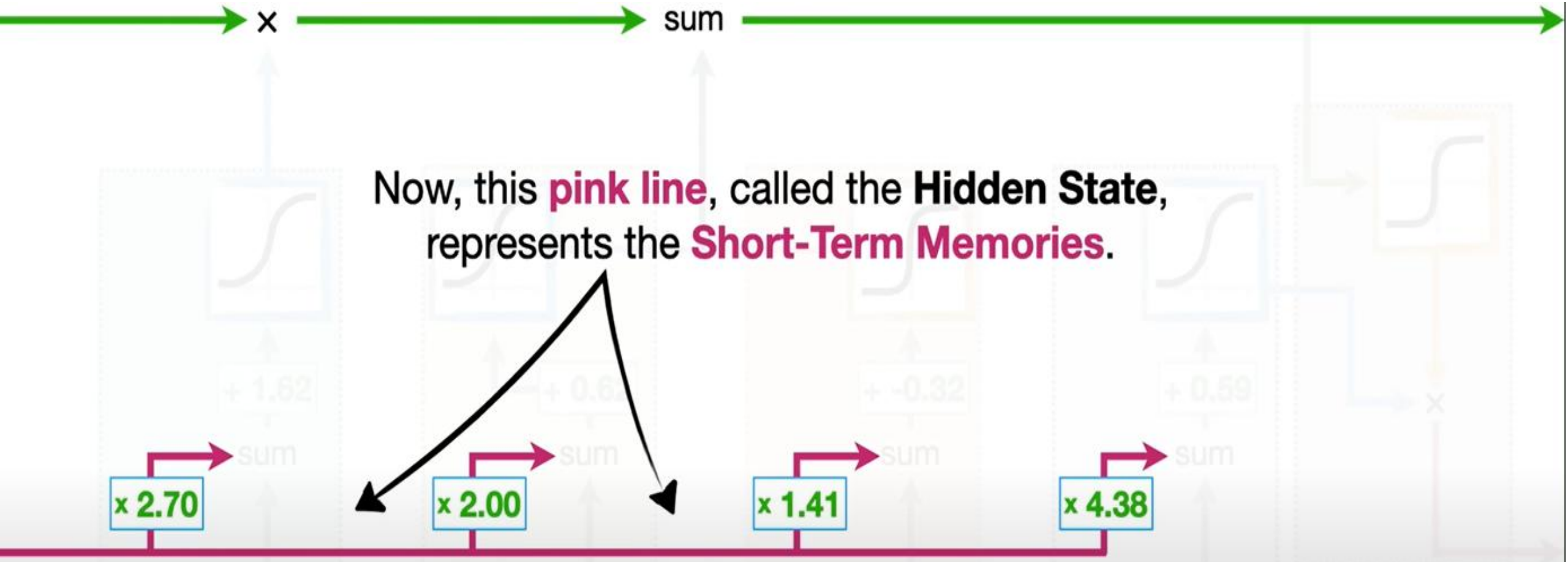

×
sum
Although the Long-Term Memory can be modified by this multiplication…
…and then later by this addition…

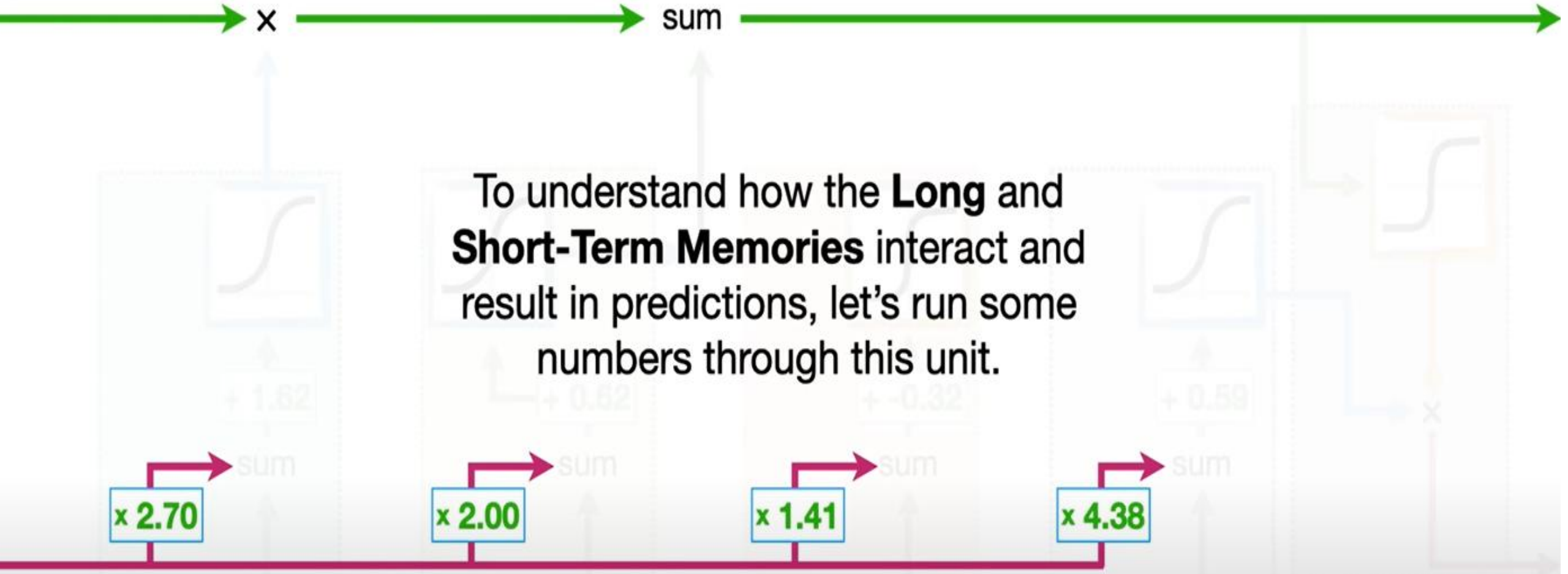


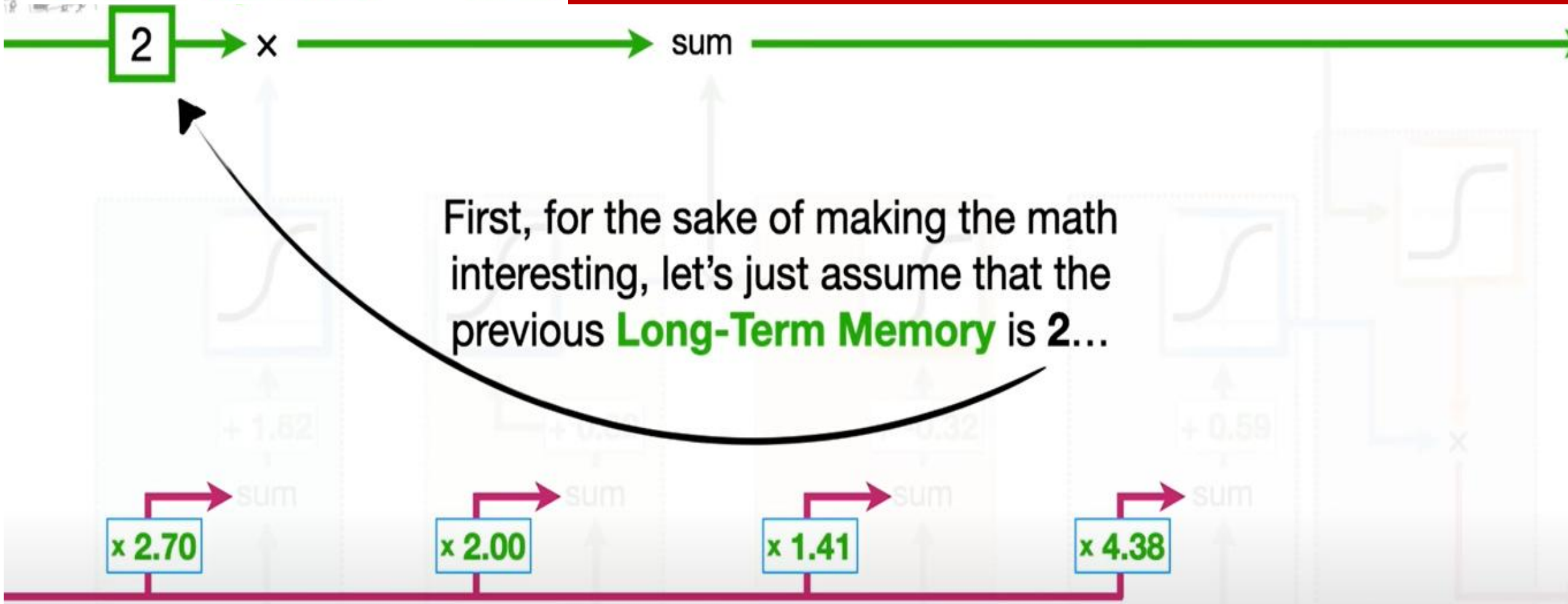
...you'll notice that there are no **Weights** and **Biases** that can modify it directly.

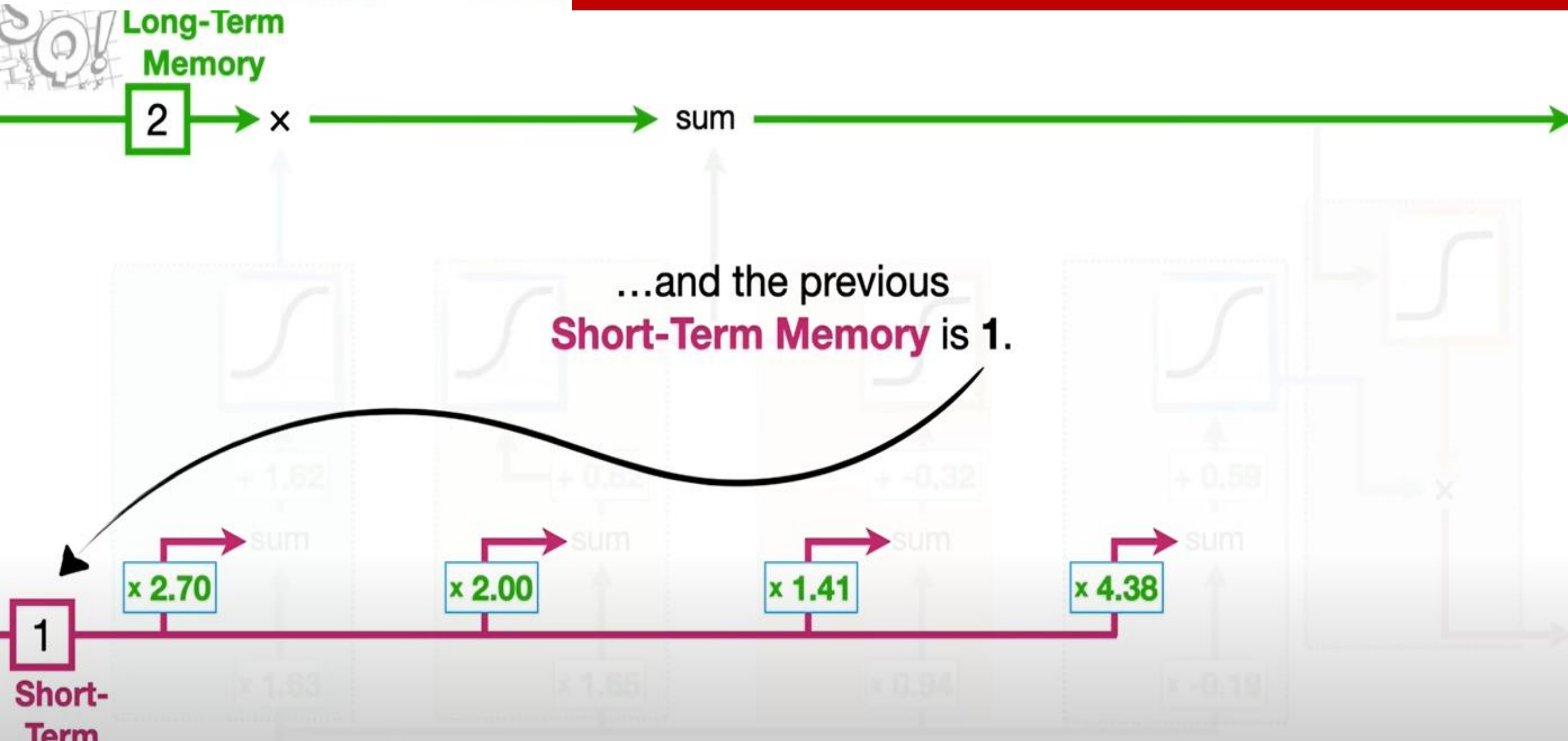


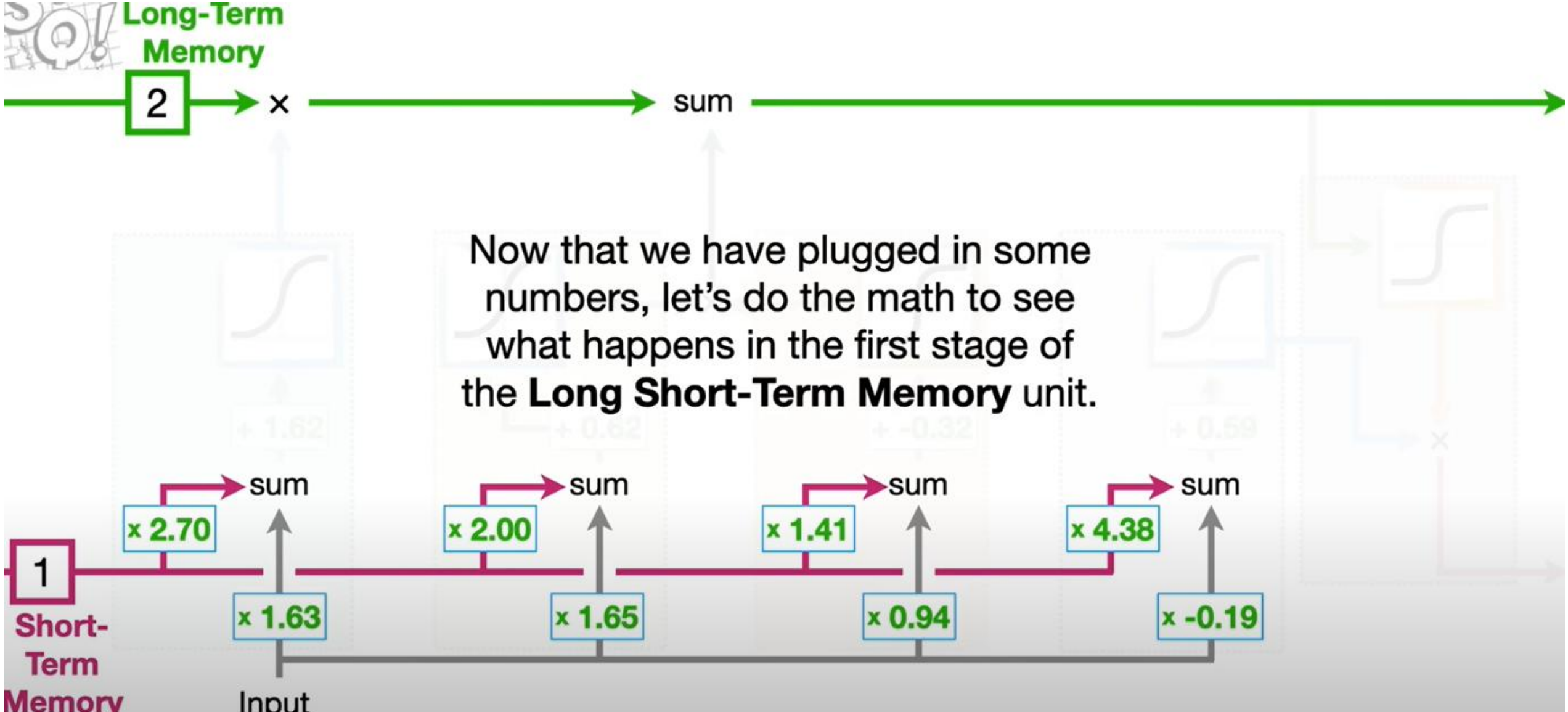
The lack of **Weights** allows the **Long-Term Memories** to flow through a series of unrolled units without causing the gradient to **explode** or **vanish**.

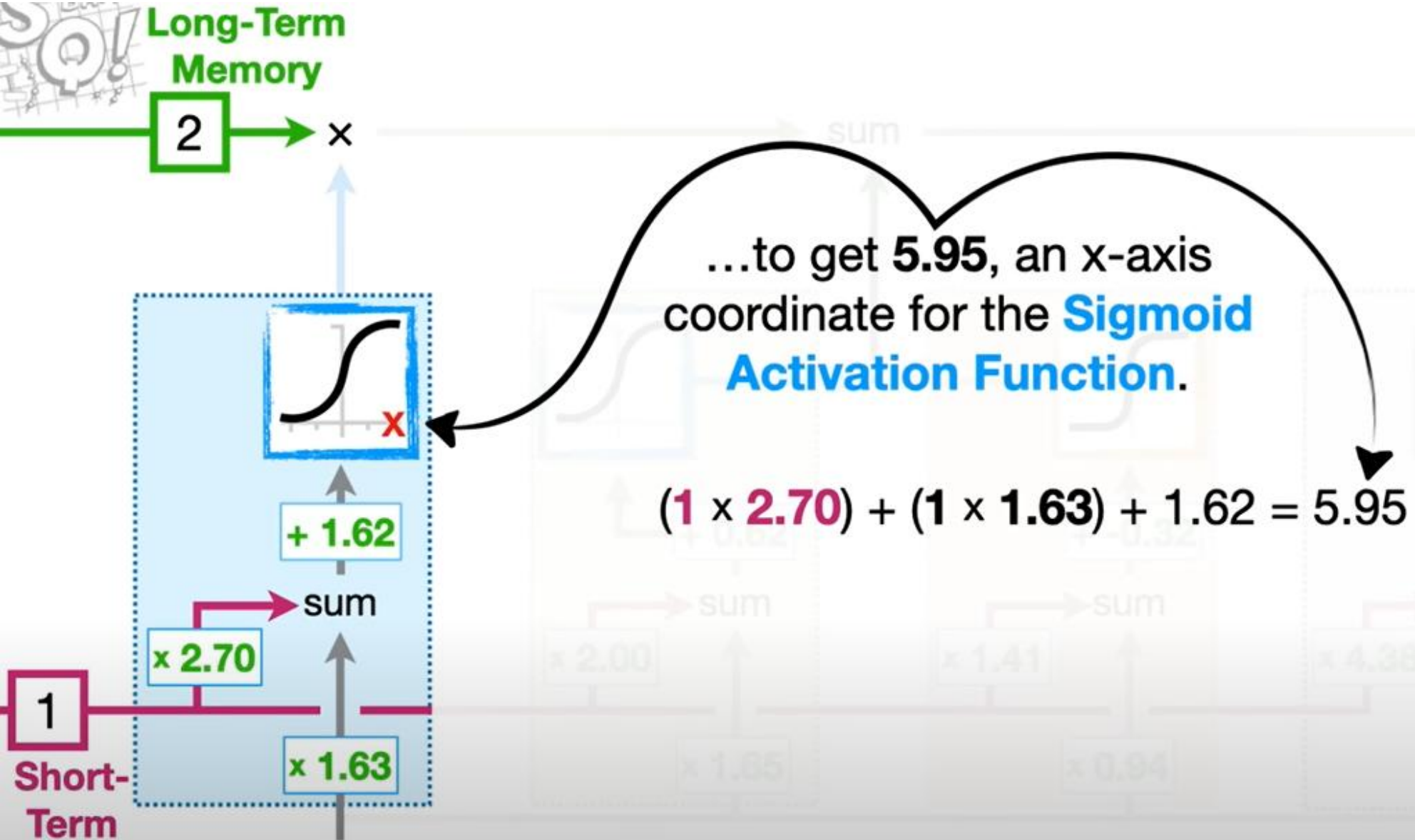

x
sum
Now, this pink line, called the Hidden State, represents the Short-Term Memories.
x 2.70
x 2.00
x 1.41
x 4.38


x
sum
To understand how the **Long** and **Short-Term Memories** interact and result in predictions, let's run some numbers through this unit.
x 2.70
x 2.00
x 1.41
x 4.38


2
x
sum
First, for the sake of making the math interesting, let's just assume that the previous Long-Term Memory is 2...
x 2.70
x 2.00
x 1.41
x 4.38

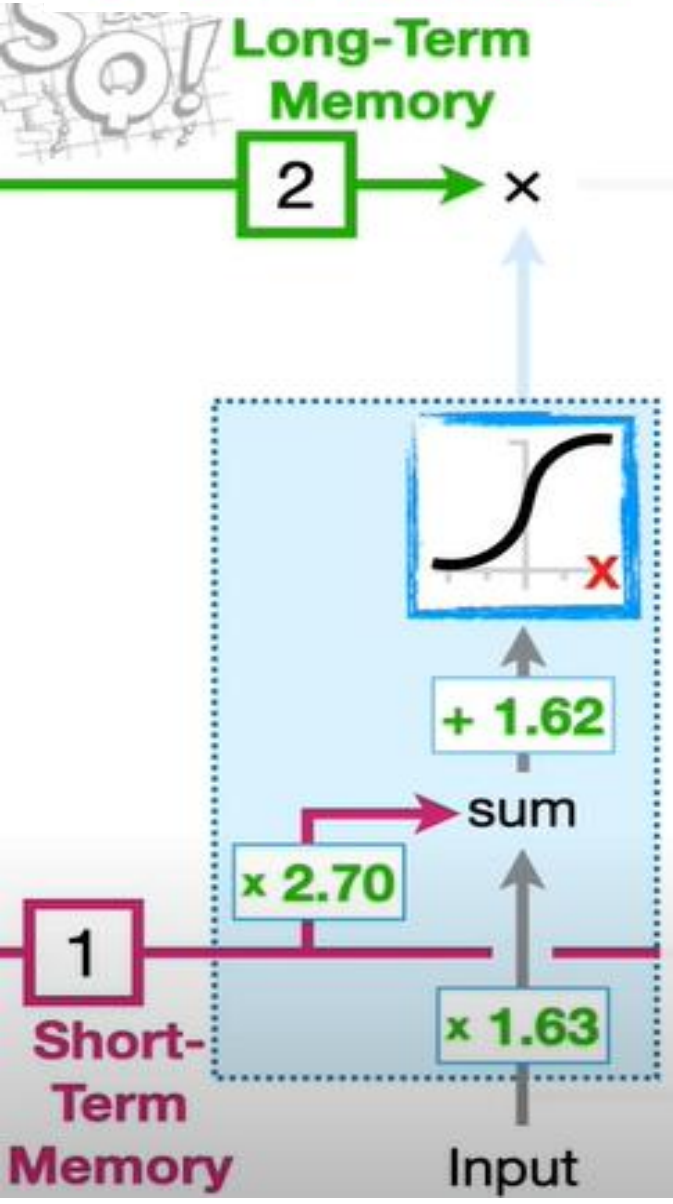

Long-Term Memory
2
x
sum
...and the previous
Short-Term Memory is 1.
x 2.70
x 2.00
x 1.41
x 4.38
1
Short-Term


Long-Term Memory
2
x
sum
Now that we have plugged in some numbers, let's do the math to see what happens in the first stage of the **Long Short-Term Memory** unit.
sum
sum
sum
sum
x 2.70
x 2.00
x 1.41
x 4.38
1
x 1.63
x 1.65
x 0.94
x -0.19
Short-Term Memory
Input



…to get **5.95**, an x-axis coordinate for the **Sigmoid Activation Function**.

$$(1 \times 2.70) + (1 \times 1.63) + 1.62 = 5.95$$

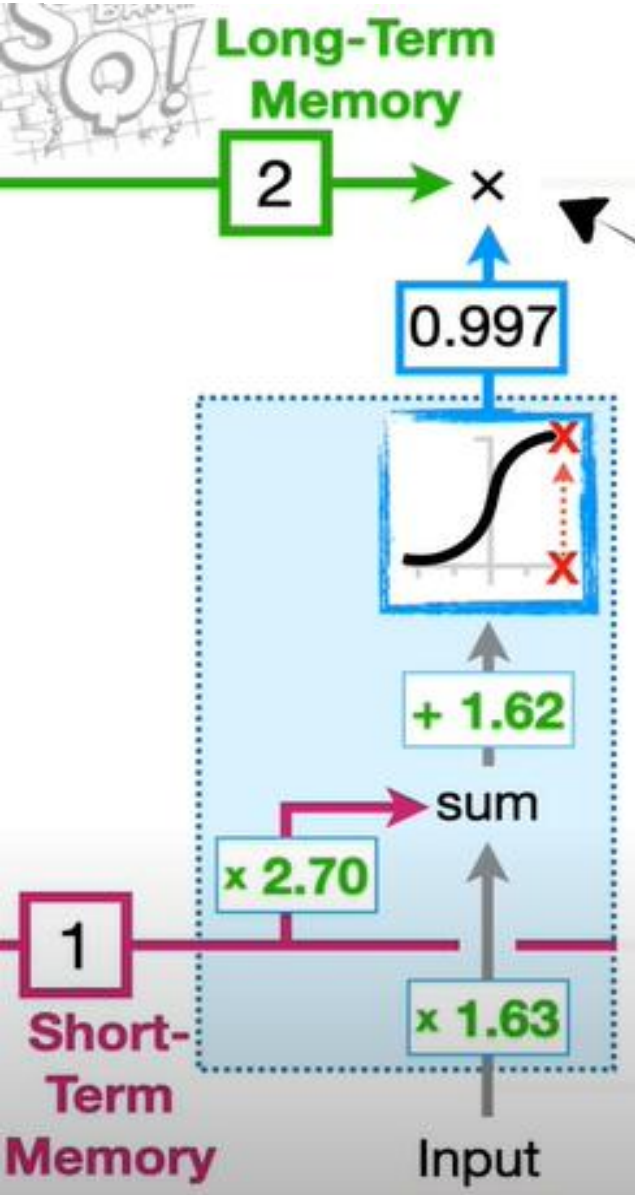


Now we plug the x-axis coordinate into the equation for the **Sigmoid Activation Function**...

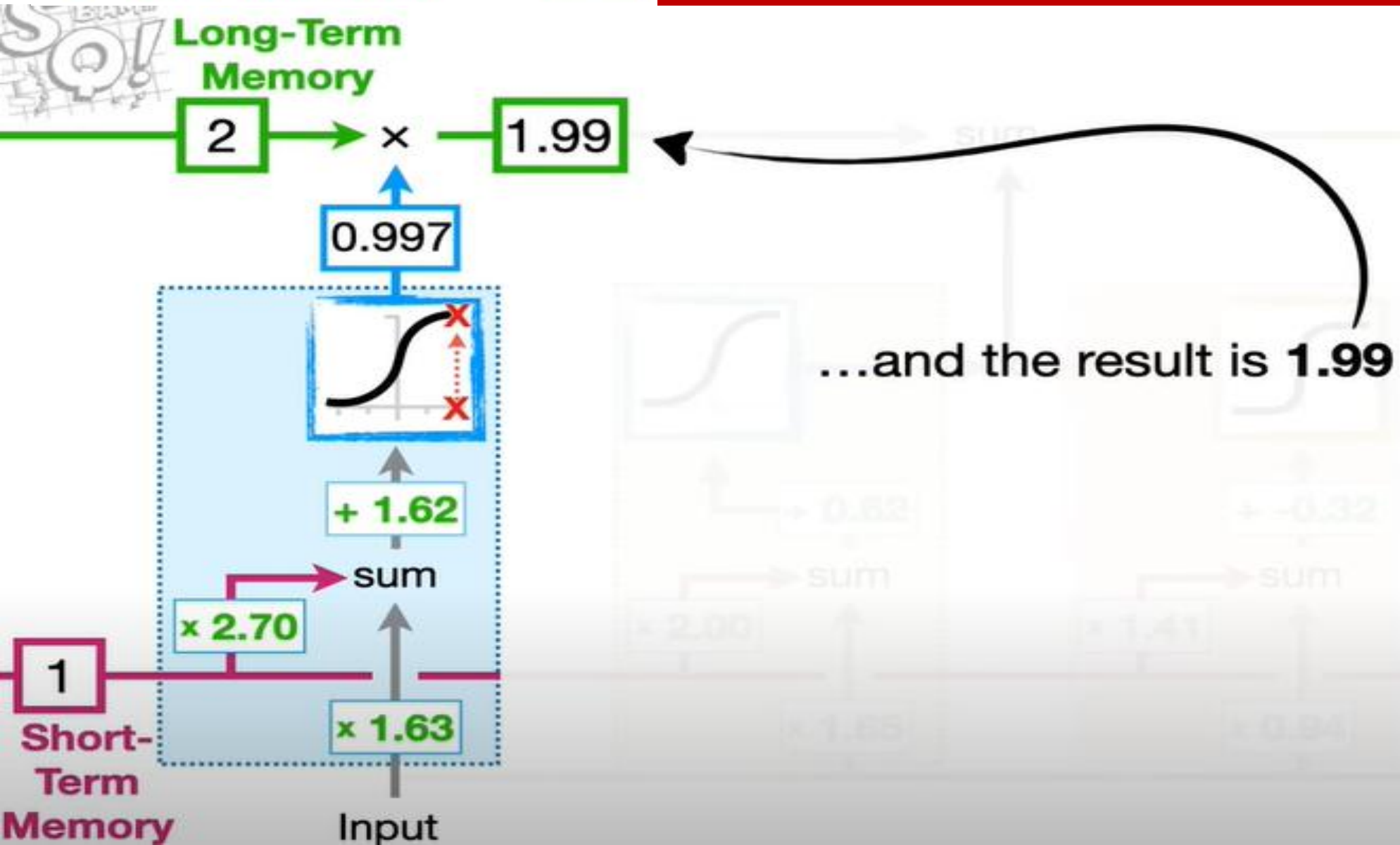
$$(1 \times 2.70) + (1 \times 1.63) + 1.62 = 5.95$$

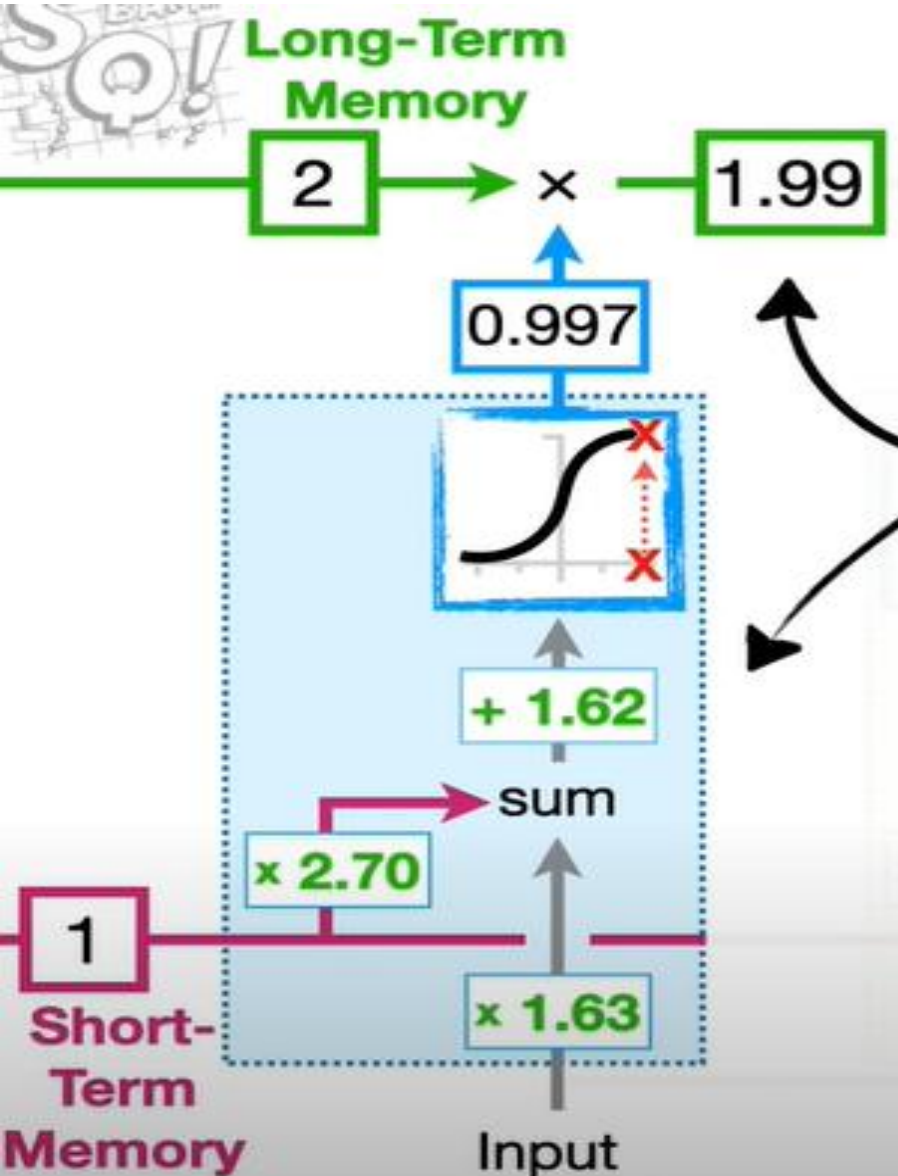
$$f(x) = \frac{e^x}{e^x + 1}$$

$$f(5.95) = \frac{e^{5.95}}{e^{5.95} + 1}$$

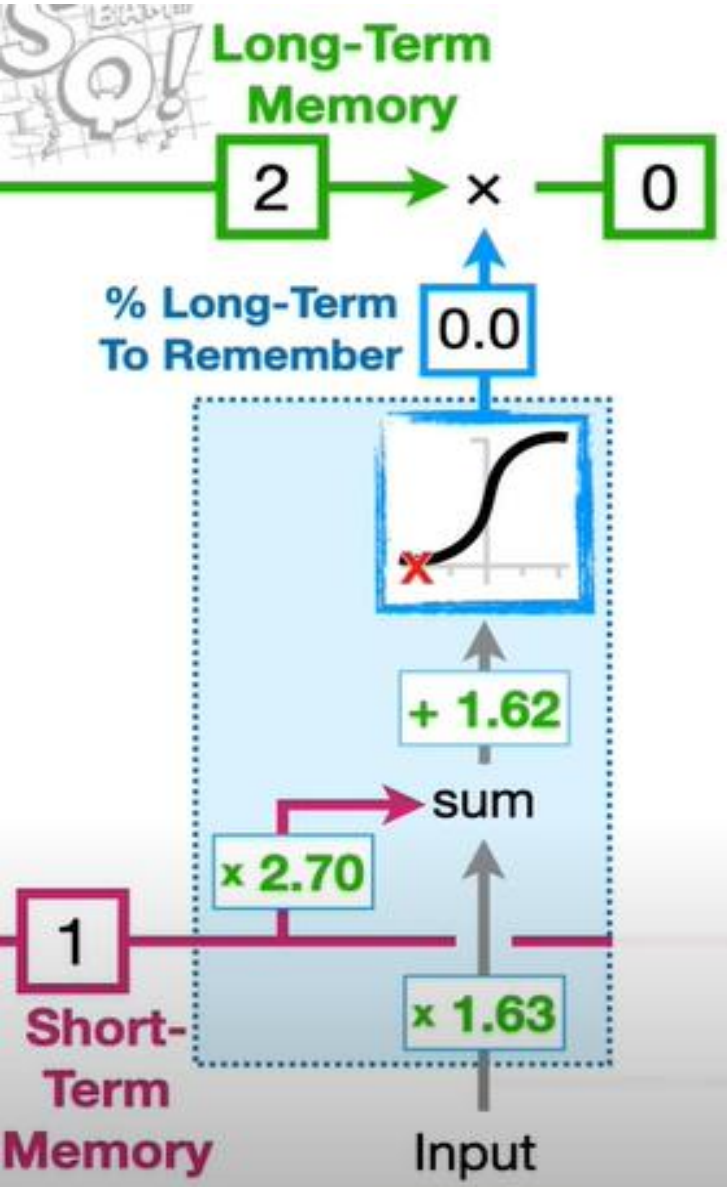


Lastly, we multiply the **Long-Term Memory**, **2**, by the y-axis coordinate, **0.997**…


Long-Term Memory
2
×
1.99
0.997
+ 1.62
sum
× 2.70
1
Short-Term Memory
× 1.63
Input
...and the result is 1.99



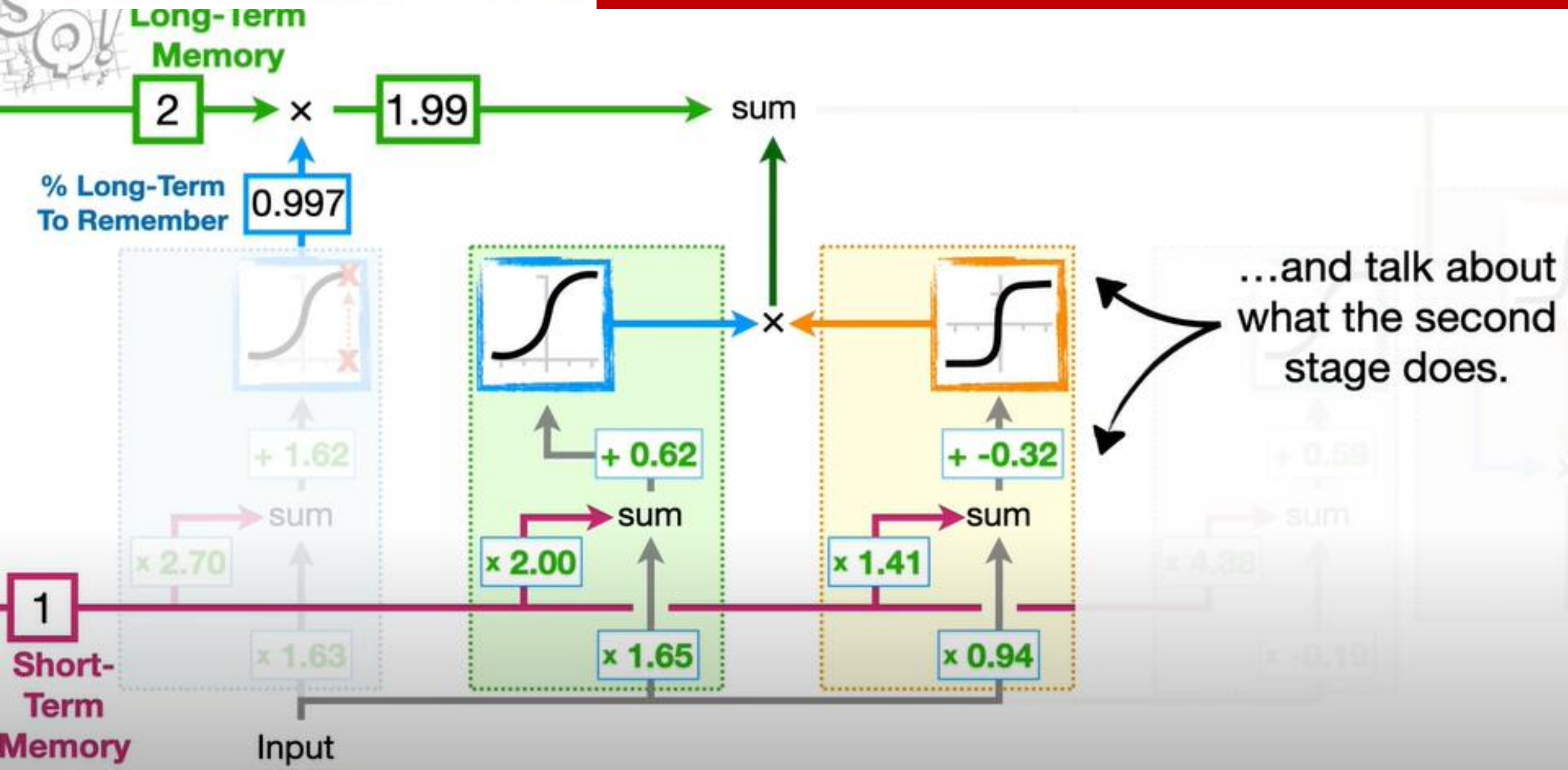
So this first stage of the **Long Short-Term Memory (LSTM)** unit reduced the **Long-Term Memory** by a little bit.

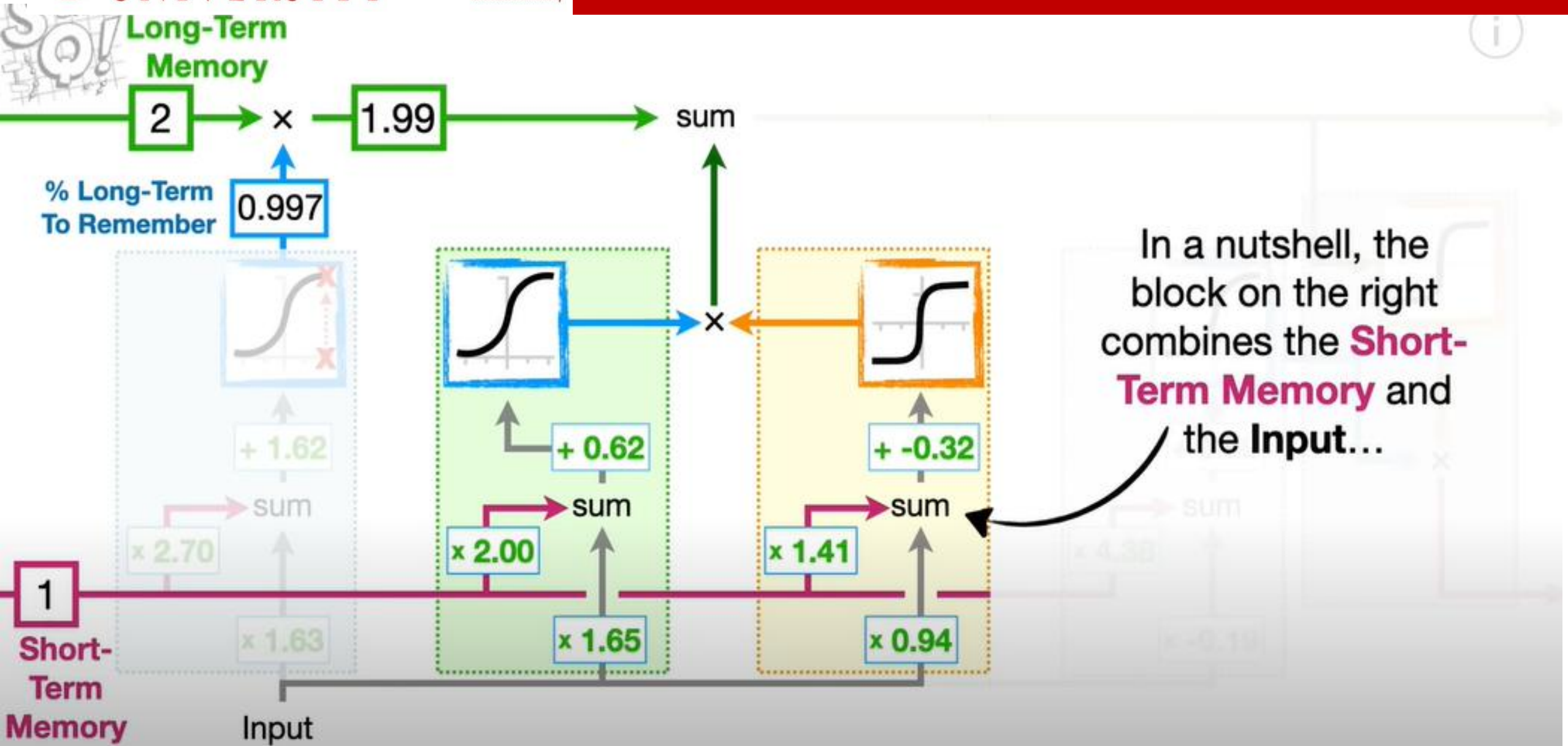


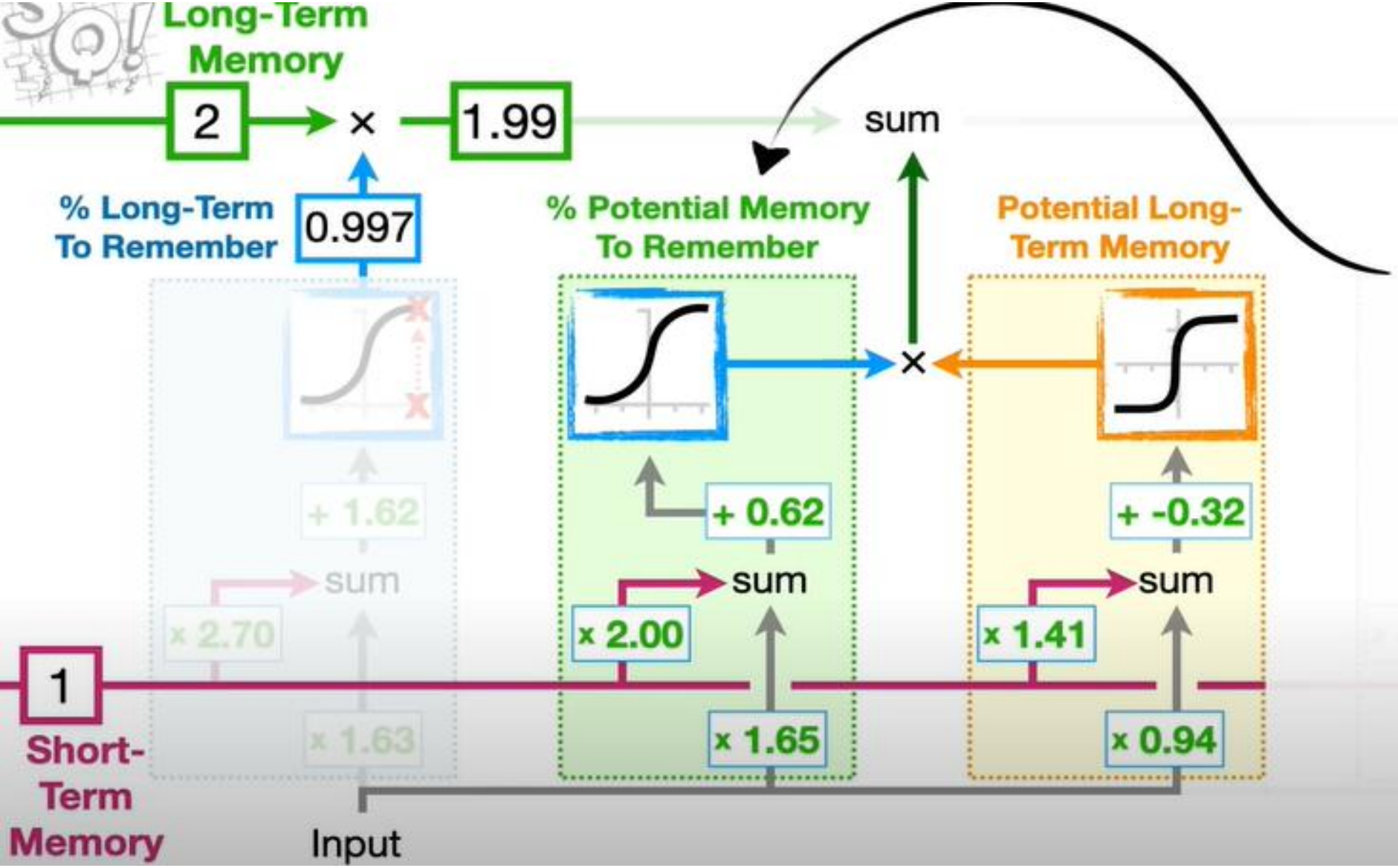
TERMINOLOGY ALERT!!!

Even though this part of the **Long Short-Term Memory** unit determines what percentage of the **Long-Term Memory** will be remembered…

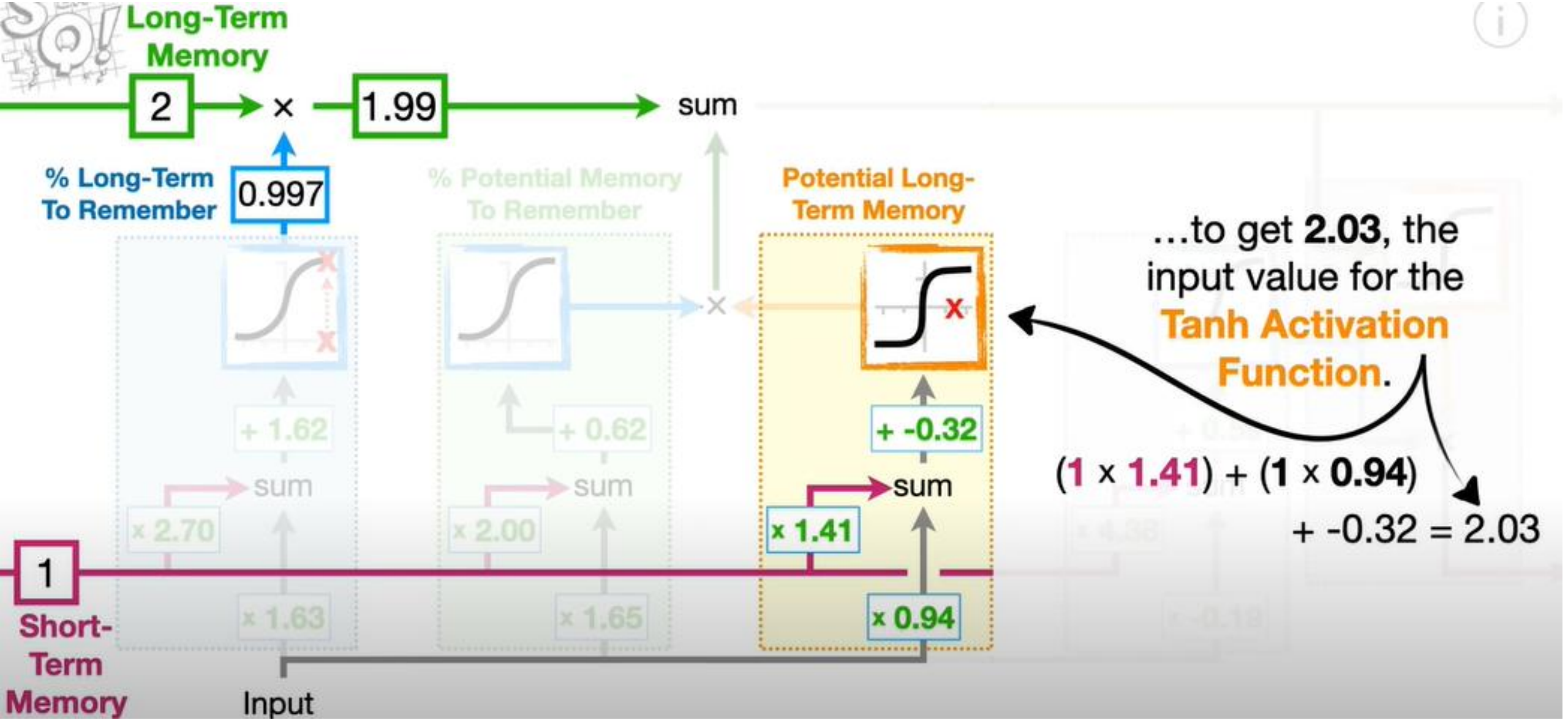
…it is usually called the **Forget Gate**.

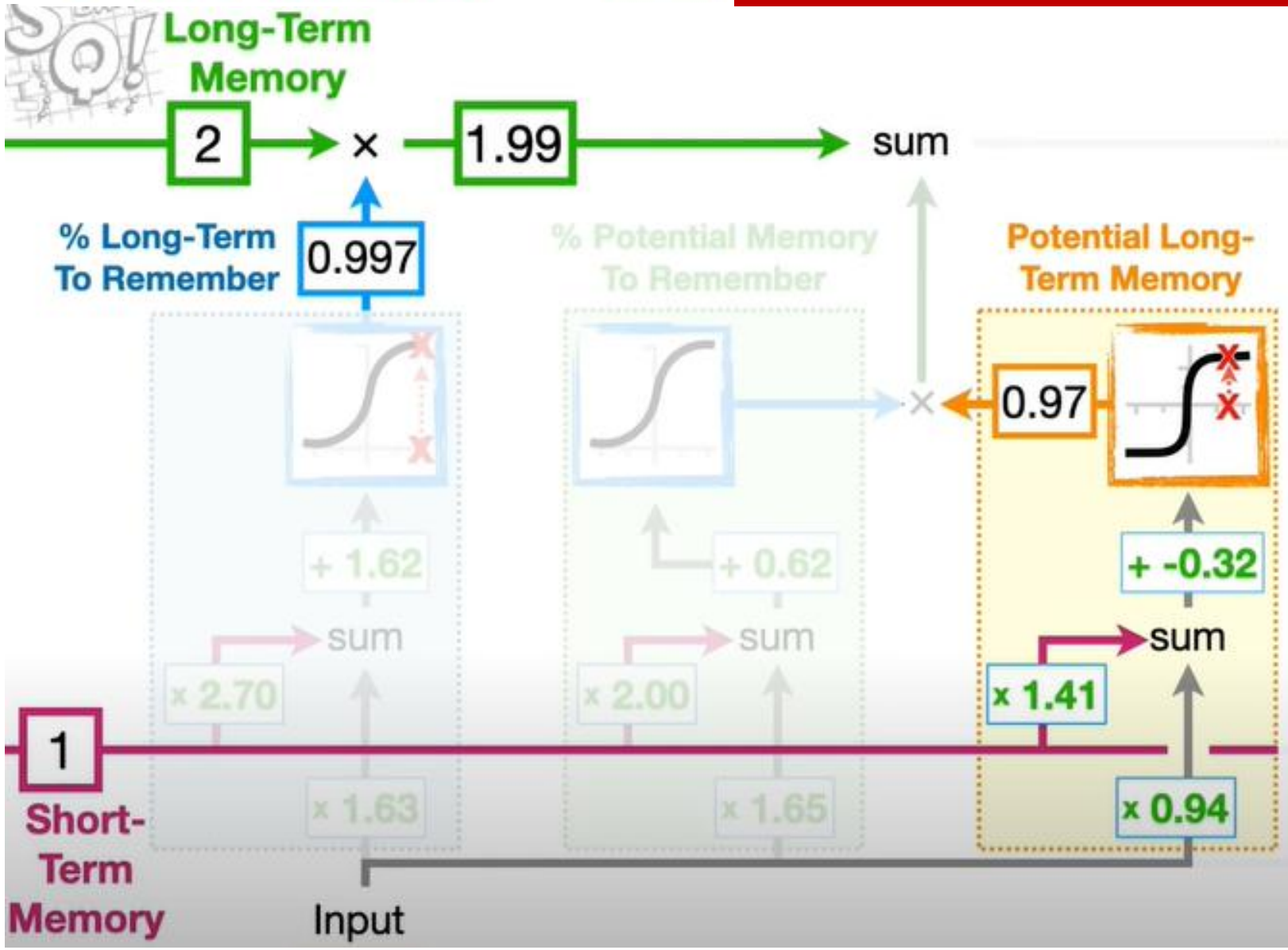

Long-Term Memory
2
×
1.99
sum
% Long-Term To Remember
0.997
+ 0.62
+ -0.32
sum
sum
× 2.00
× 1.41
× 1.65
× 0.94
1
Short-Term Memory
Input
...and talk about what the second stage does.


Long-Term Memory
2
×
1.99
sum
% Long-Term To Remember
0.997
+ 1.62
sum
× 2.70
× 1.63
+ 0.62
sum
× 2.00
× 1.65
×
+ -0.32
sum
× 1.41
× 0.94
1
Short-Term Memory
Input
In a nutshell, the block on the right combines the Short-Term Memory and the Input…

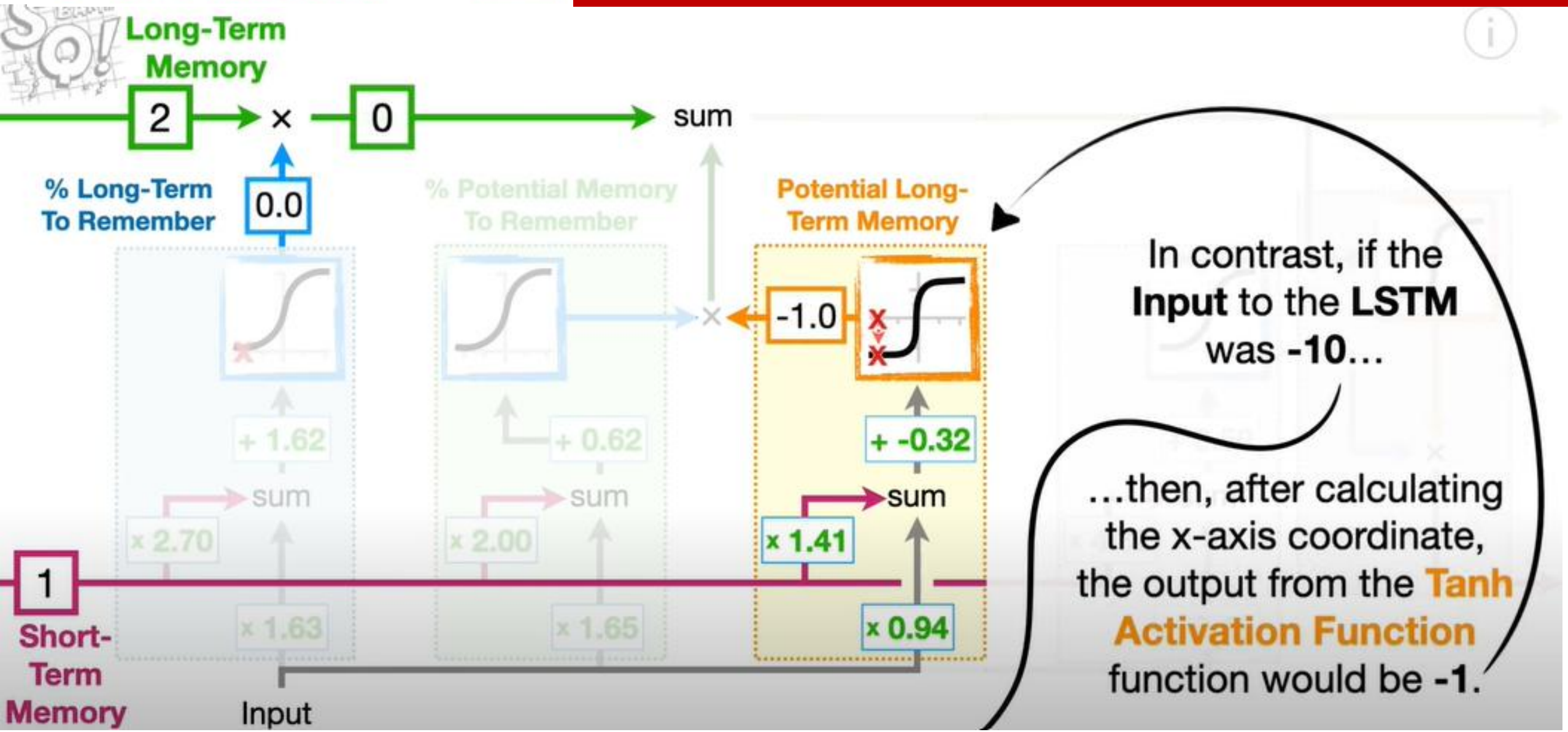


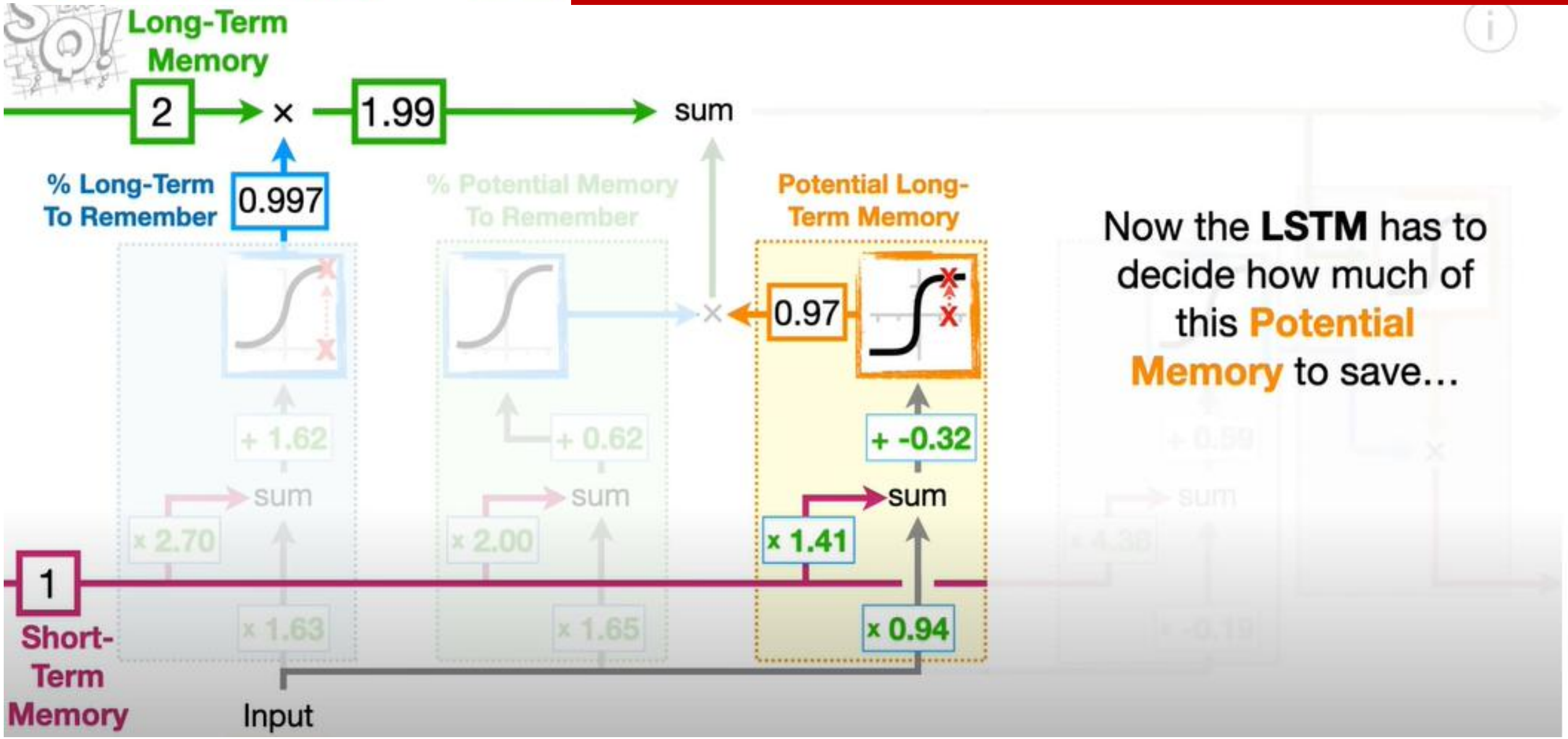
...and the block on the left determines what percentage of that **Potential Memory** to add to the **Long-Term Memory**.

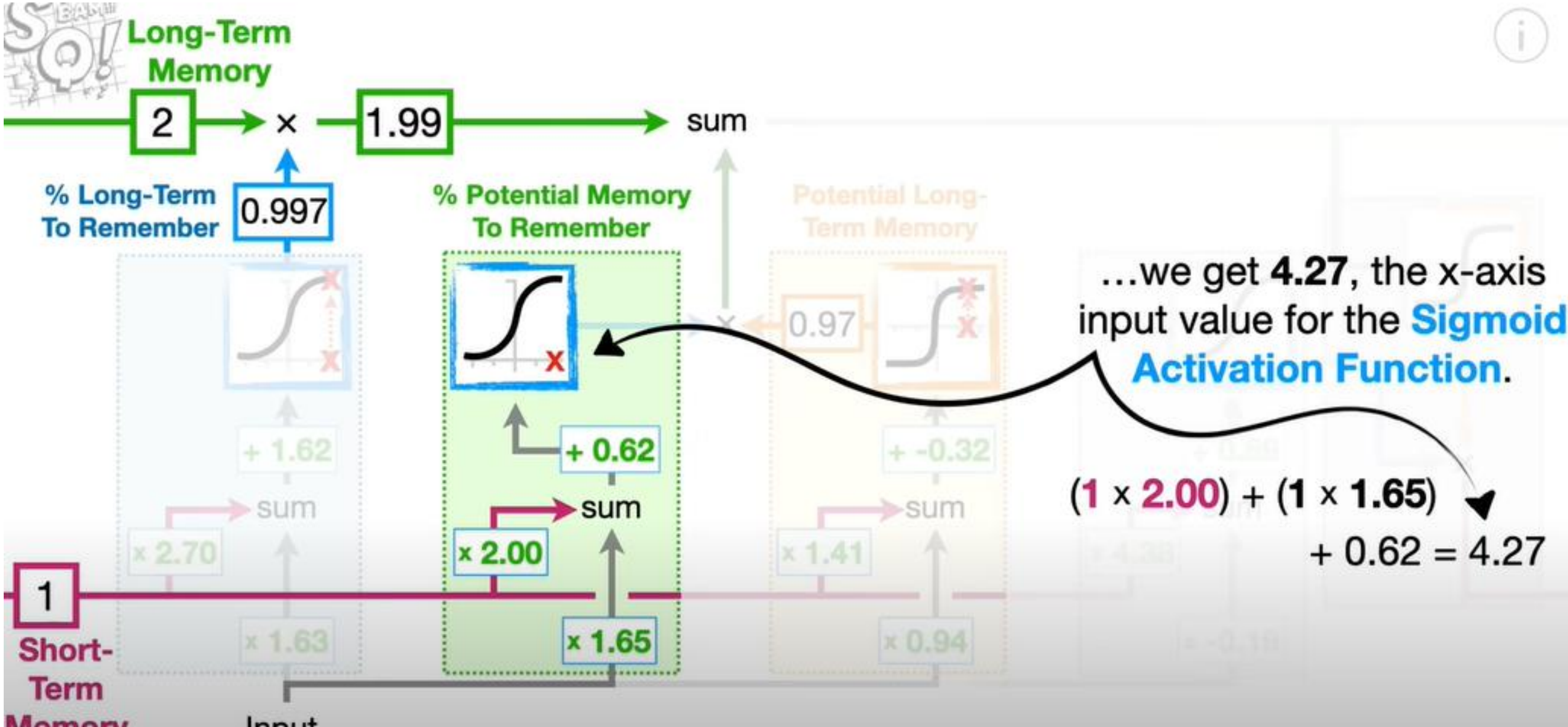

Long-Term Memory
2
×
1.99
sum
% Long-Term To Remember
0.997
% Potential Memory To Remember
Potential Long-Term Memory
+ 1.62
+ 0.62
+ -0.32
sum
sum
sum
× 2.70
× 2.00
× 1.41
1
× 1.63
× 1.65
× 0.94
Short-Term Memory
Input
…to get 2.03, the input value for the Tanh Activation Function.
(1 × 1.41) + (1 × 0.94) + -0.32 = 2.03

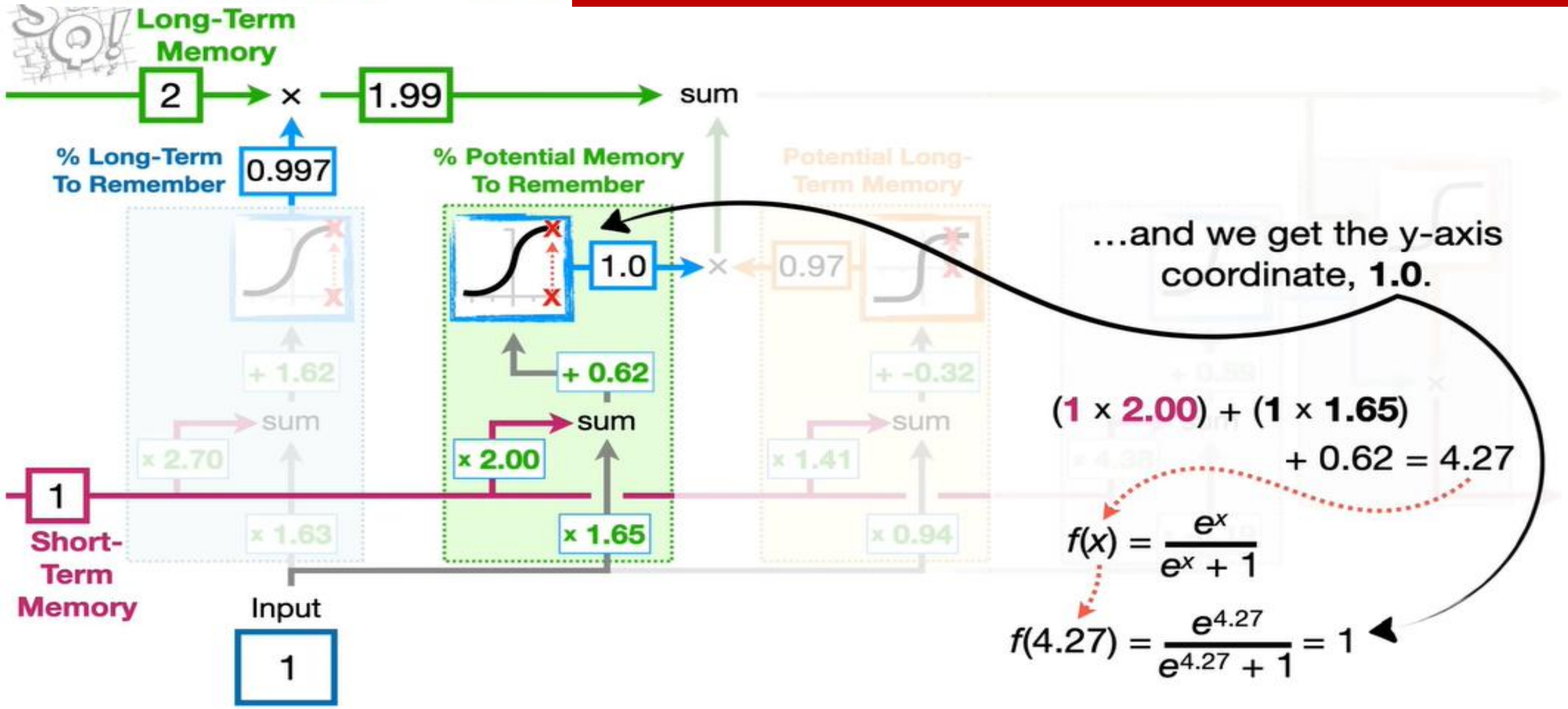


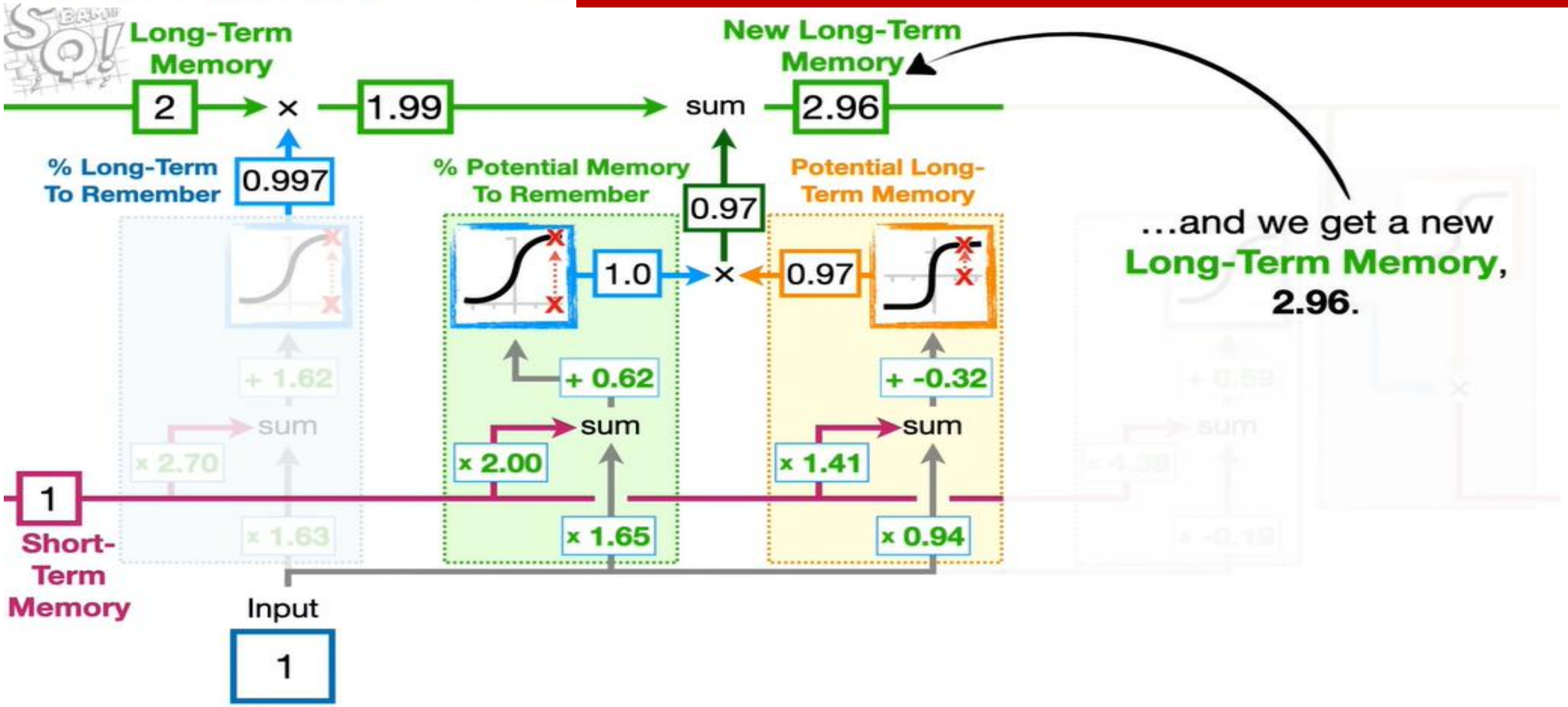
Remember, the **Tanh Activation Function** turns any input into a number between **-1** and **1**...

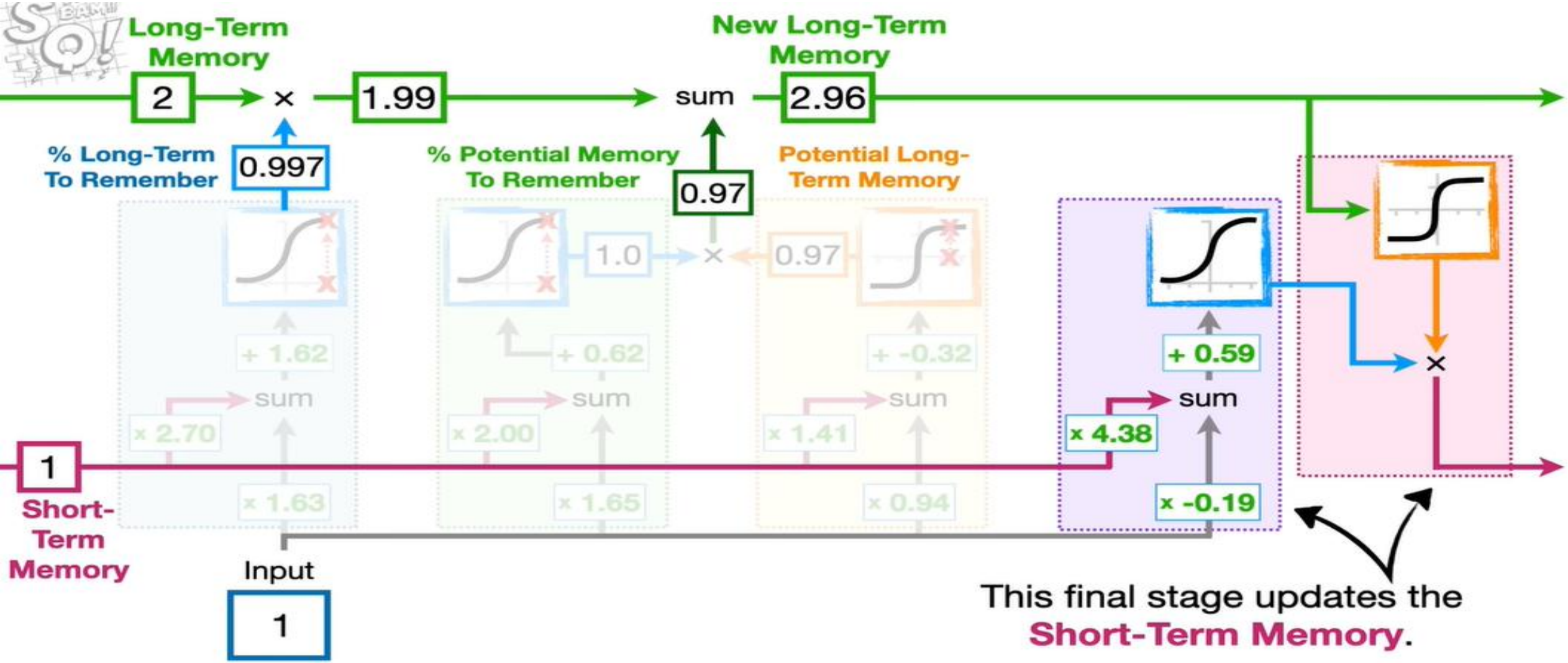

Long-Term Memory
2
×
0
sum
% Long-Term To Remember
0.0
% Potential Memory To Remember
Potential Long-Term Memory
-1.0
+ 1.62
+ 0.62
+ -0.32
sum
sum
sum
× 2.70
× 2.00
× 1.41
1
× 1.63
× 1.65
× 0.94
Short-Term Memory
Input
In contrast, if the Input to the LSTM was -10…
…then, after calculating the x-axis coordinate, the output from the Tanh Activation Function function would be -1.

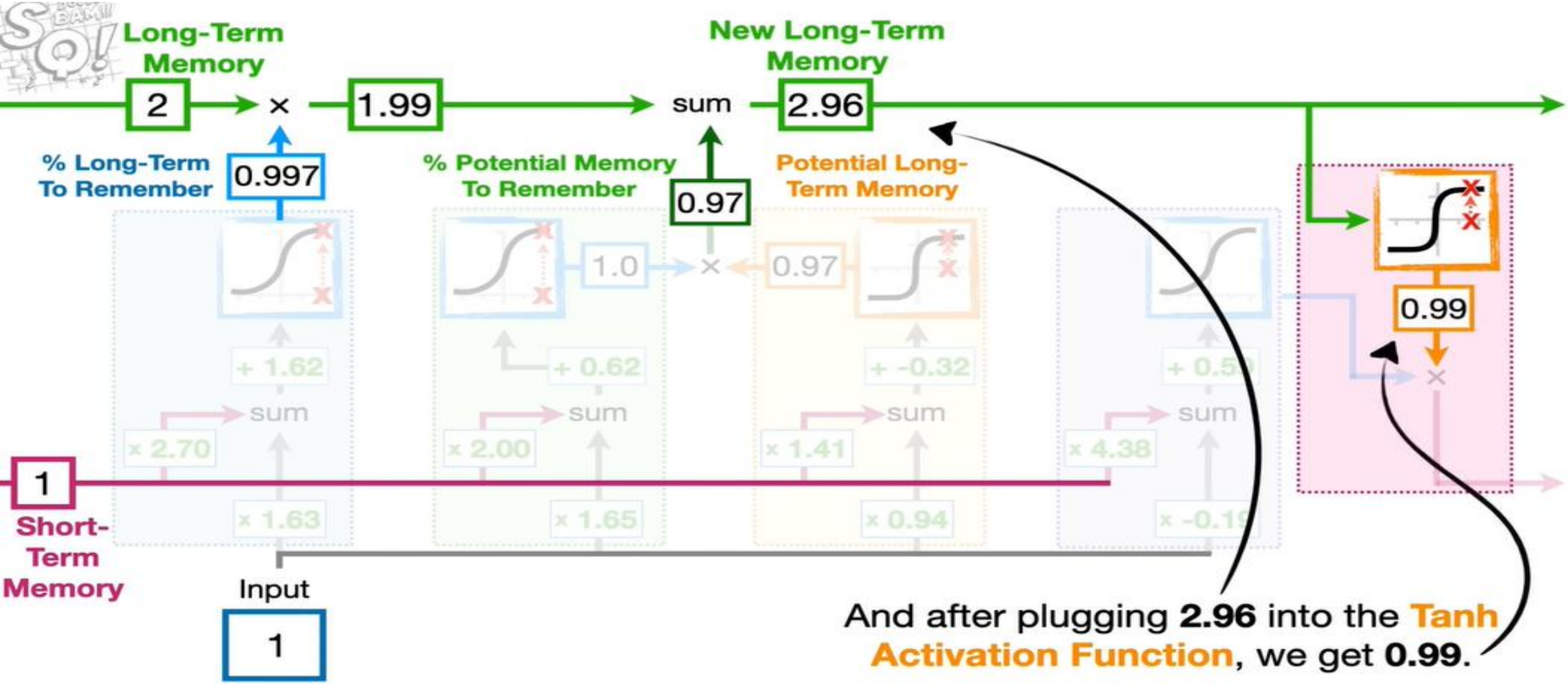

Long-Term Memory
2
×
1.99
sum
% Long-Term To Remember
0.997
% Potential Memory To Remember
Potential Long-Term Memory
0.97
+ 1.62
+ 0.62
+ -0.32
sum
sum
sum
× 2.70
× 2.00
× 1.41
1
× 1.63
× 1.65
× 0.94
Short-Term Memory
Input
Now the LSTM has to decide how much of this Potential Memory to save…

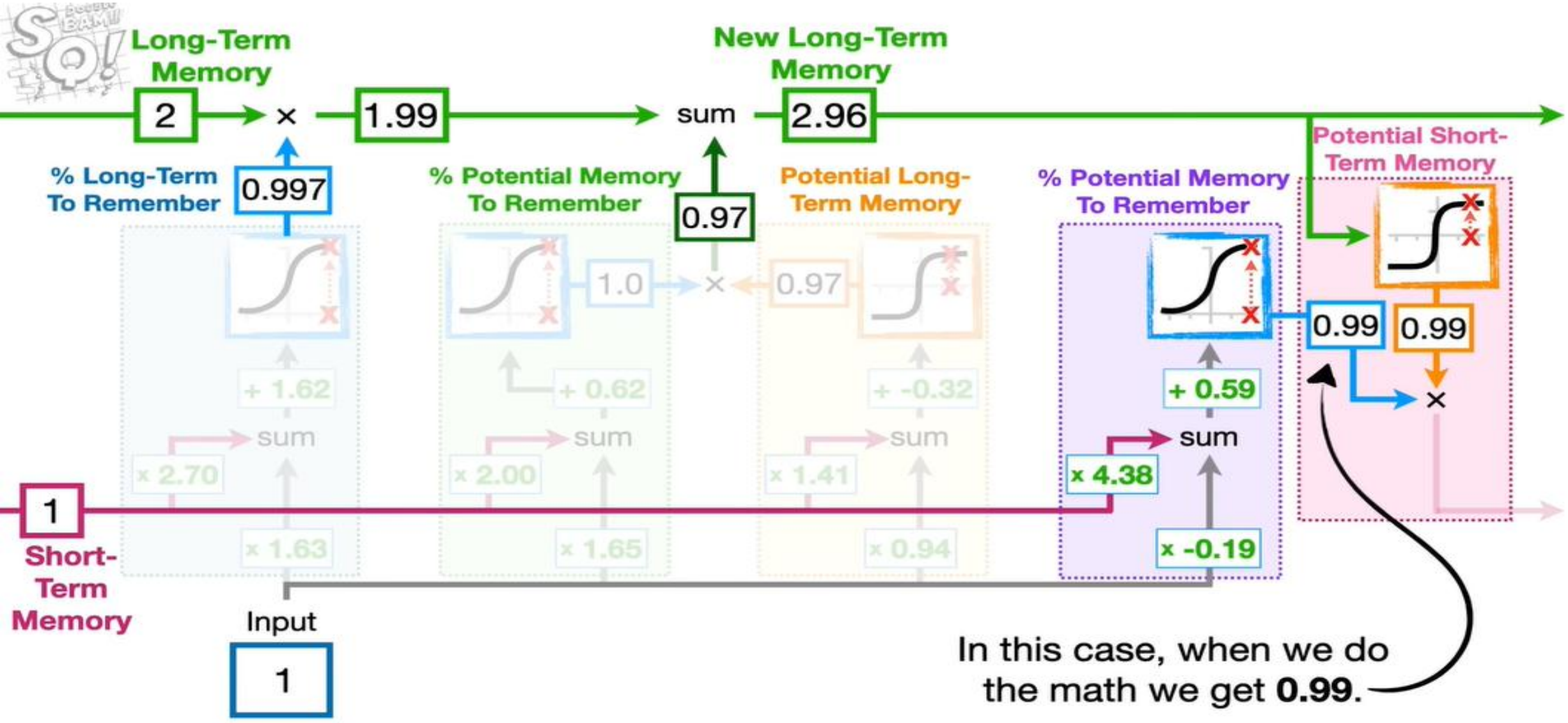

Long-Term Memory
2
×
1.99
sum
% Long-Term To Remember
0.997
% Potential Memory To Remember
Potential Long-Term Memory
0.97
+ 1.62
+ 0.62
+ -0.32
sum
sum
sum
× 2.70
× 2.00
× 1.41
1
× 1.63
× 1.65
× 0.94
Short-Term Memory
Input
...we get 4.27, the x-axis input value for the Sigmoid Activation Function.
(1 × 2.00) + (1 × 1.65) + 0.62 = 4.27

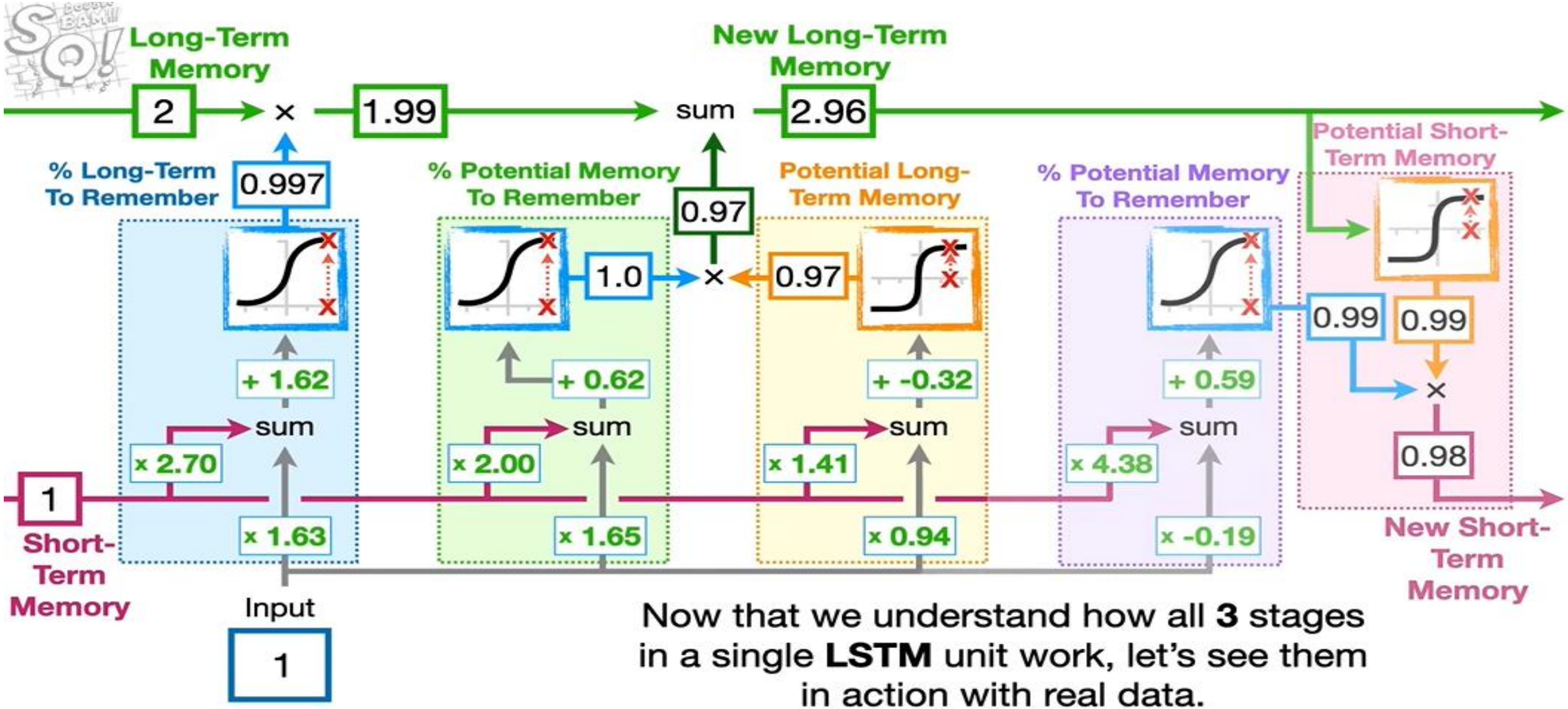

Long-Term Memory
2
×
1.99
sum
% Long-Term To Remember
0.997
% Potential Memory To Remember
Potential Long-Term Memory
1.0
×
0.97
+ 1.62
sum
× 2.70
+ 0.62
sum
× 2.00
+ -0.32
sum
× 1.41
1
Short-Term Memory
× 1.63
× 1.65
× 0.94
Input
1
...and we get the y-axis coordinate, 1.0.
(1 × 2.00) + (1 × 1.65) + 0.62 = 4.27
$f(x) = \frac{e^x}{e^x + 1}$
$f(4.27) = \frac{e^{4.27}}{e^{4.27} + 1} = 1$


Long-Term Memory
2
×
1.99
sum
New Long-Term Memory
2.96
% Long-Term To Remember
0.997
% Potential Memory To Remember
0.97
Potential Long-Term Memory
1.0
×
0.97
+ 0.62
+ -0.32
sum
sum
× 2.00
× 1.41
× 1.65
× 0.94
1
Short-Term Memory
Input
1
...and we get a new Long-Term Memory, 2.96.


Long-Term Memory
2
×
1.99
sum
New Long-Term Memory
2.96
% Long-Term To Remember
0.997
% Potential Memory To Remember
0.97
Potential Long-Term Memory
1.0
×
0.97
+ 1.62
sum
× 2.70
× 1.63
+ 0.62
sum
× 2.00
× 1.65
+ -0.32
sum
× 1.41
× 0.94
+ 0.59
sum
× 4.38
× -0.19
×
1
Short-Term Memory
Input
1
This final stage updates the Short-Term Memory.


Long-Term Memory
2
×
1.99
sum
New Long-Term Memory
2.96
% Long-Term To Remember
0.997
% Potential Memory To Remember
0.97
Potential Long-Term Memory
1.0
×
0.97
+ 1.62
sum
× 2.70
× 1.63
+ 0.62
sum
× 2.00
× 1.65
+ -0.32
sum
× 1.41
× 0.94
+ 0.5
sum
× 4.38
× -0.19
0.99
1
Short-Term Memory
Input
1
And after plugging 2.96 into the Tanh Activation Function, we get 0.99.


Long-Term Memory
2
×
1.99
sum
New Long-Term Memory
2.96
% Long-Term To Remember
0.997
% Potential Memory To Remember
0.97
Potential Long-Term Memory
% Potential Memory To Remember
Potential Short-Term Memory
1.0
×
0.97
+ 1.62
sum
× 2.70
× 1.63
+ 0.62
sum
× 2.00
× 1.65
+ -0.32
sum
× 1.41
× 0.94
+ 0.59
sum
× 4.38
× -0.19
0.99
0.99
×
1
Short-Term Memory
Input
1
In this case, when we do the math we get 0.99.



Now that we understand how all **3** stages in a single **LSTM** unit work, let's see them in action with real data.

Thank You