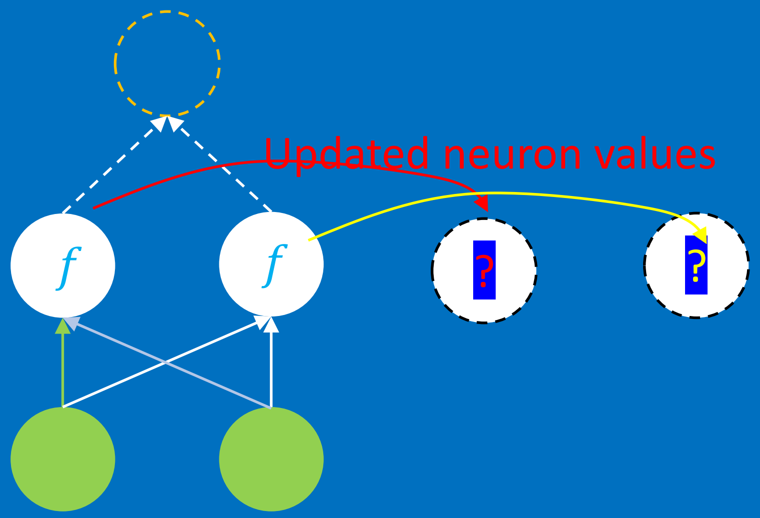
RNN: LSTM (Long short-term memory)

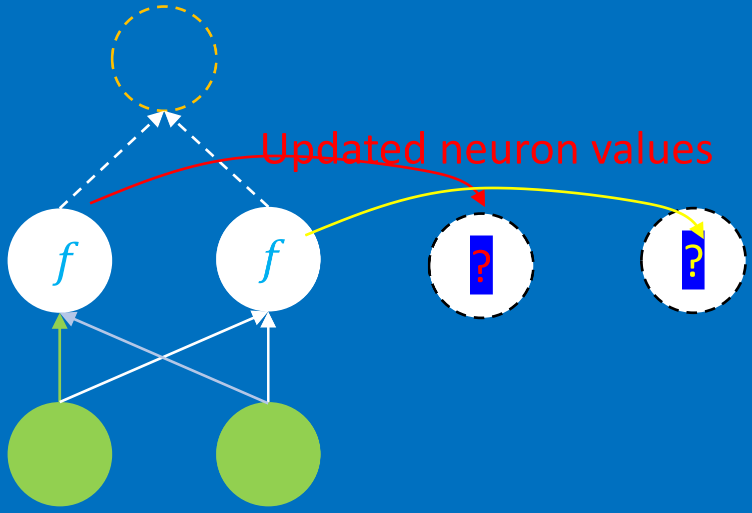
The difference between Simple RNN and LSTM

For simple RNN, all the updated neuron values are written into the memory and can be used by subsequent time step

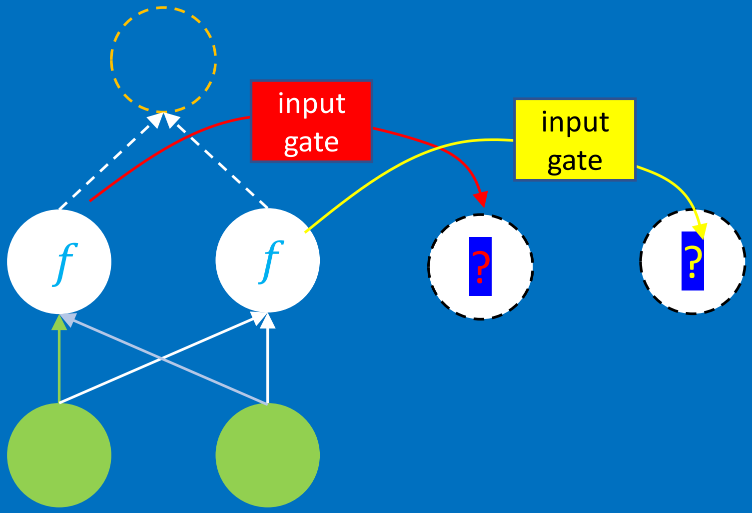


The difference between Simple RNN and LSTM

For simple RNN, all the updated neuron values are written into the memory and can be used by subsequent time step



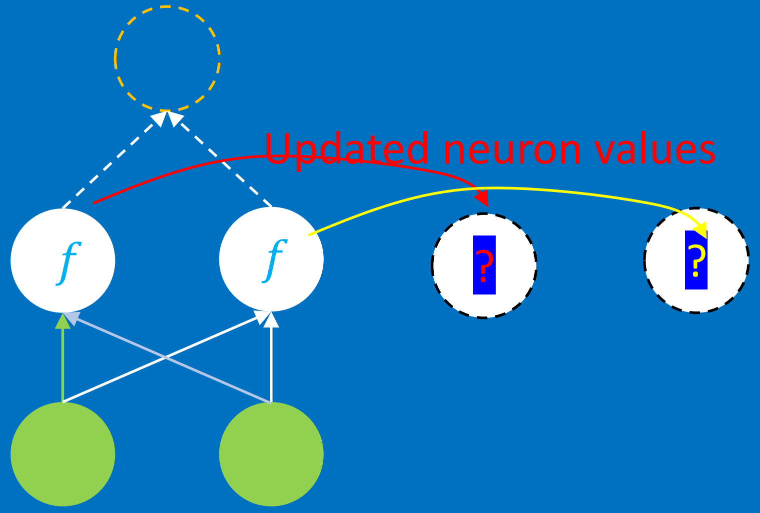
For LSTM, First there is an “input gate” to control whether we write the updated neuron value into memory



whether the status of “input gate” is “open” or “close” is learnt by the model during training

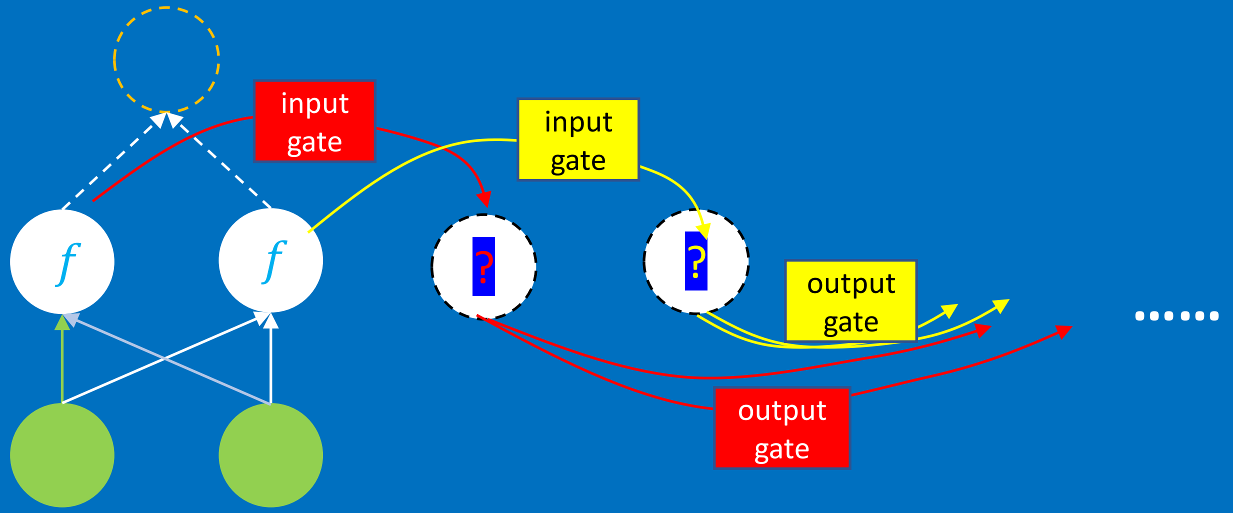
The difference between Simple RNN and LSTM

For simple RNN, all the updated neuron values are written into the memory and can be used by subsequent time step



For LSTM, <u>First</u> there is an "input gate" to control whether we write the updated neuron value into memory

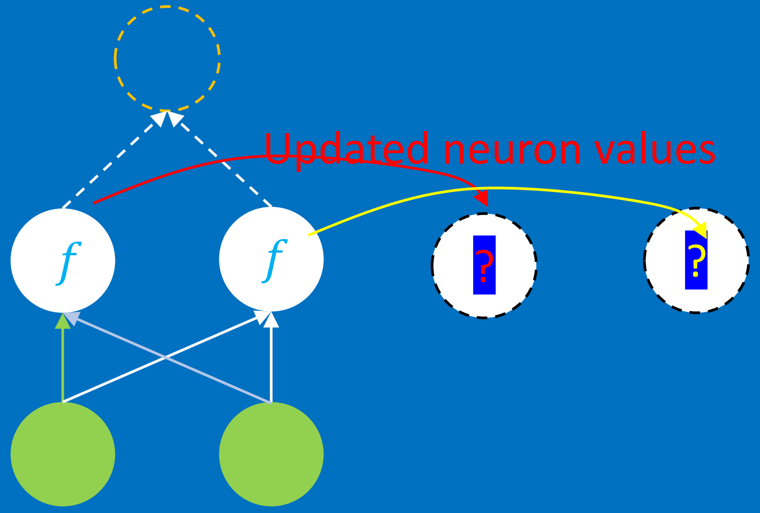
<u>Second</u> there is an "output gate" to control whether the next timestep can use these updated neuron values



whether the status of "input gate" is "open" or "close" is learnt by the model during training

The difference between Simple RNN and LSTM

For simple RNN, all the updated neuron values are written into the memory and can be used by subsequent time step

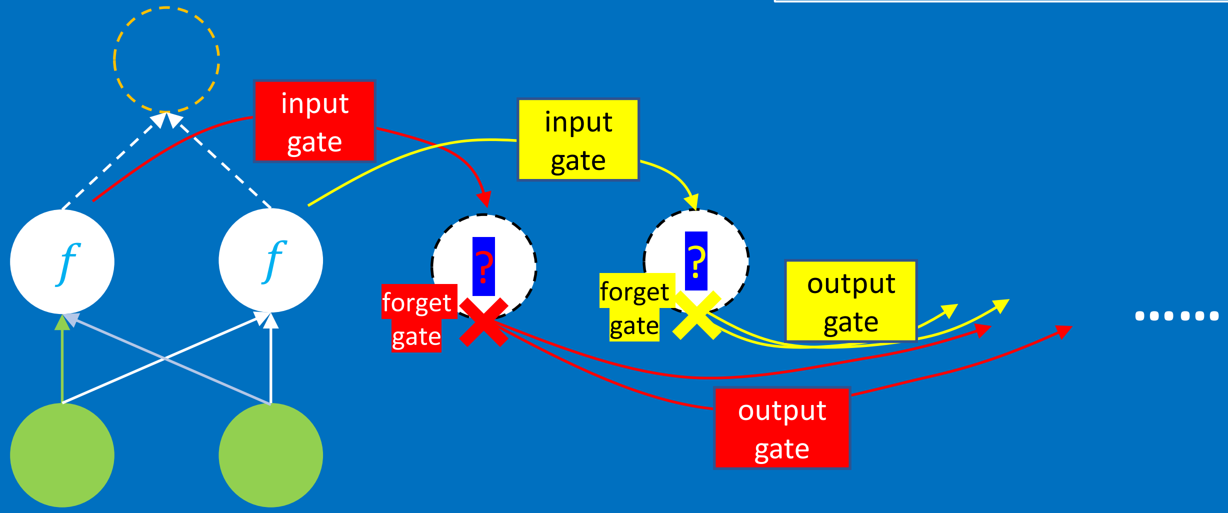


For LSTM, First there is an “input gate” to control whether we write the updated neuron value into memory

Second there is an “output gate” to control whether the next timestep can use these updated neuron values

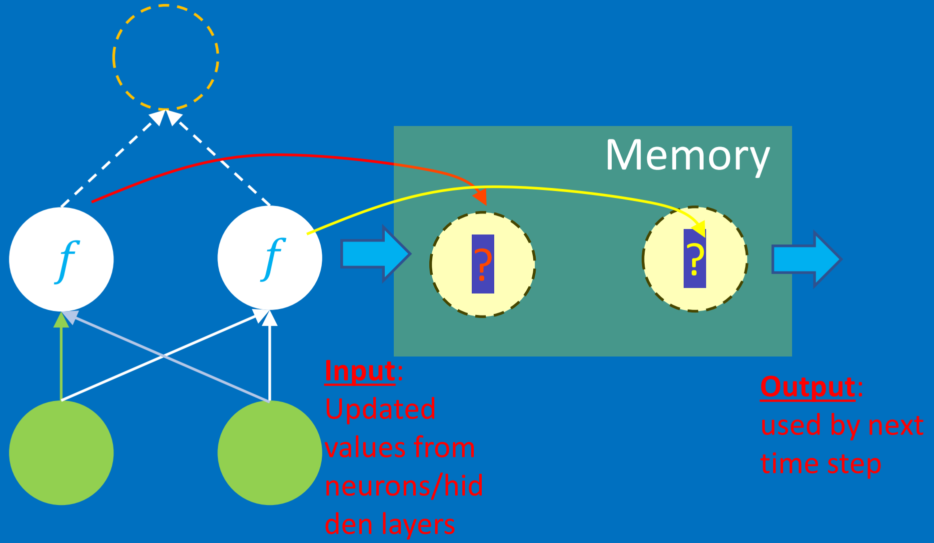
Third there is an “forget gate” to decide whether to forget ”all” the memorized neuron values

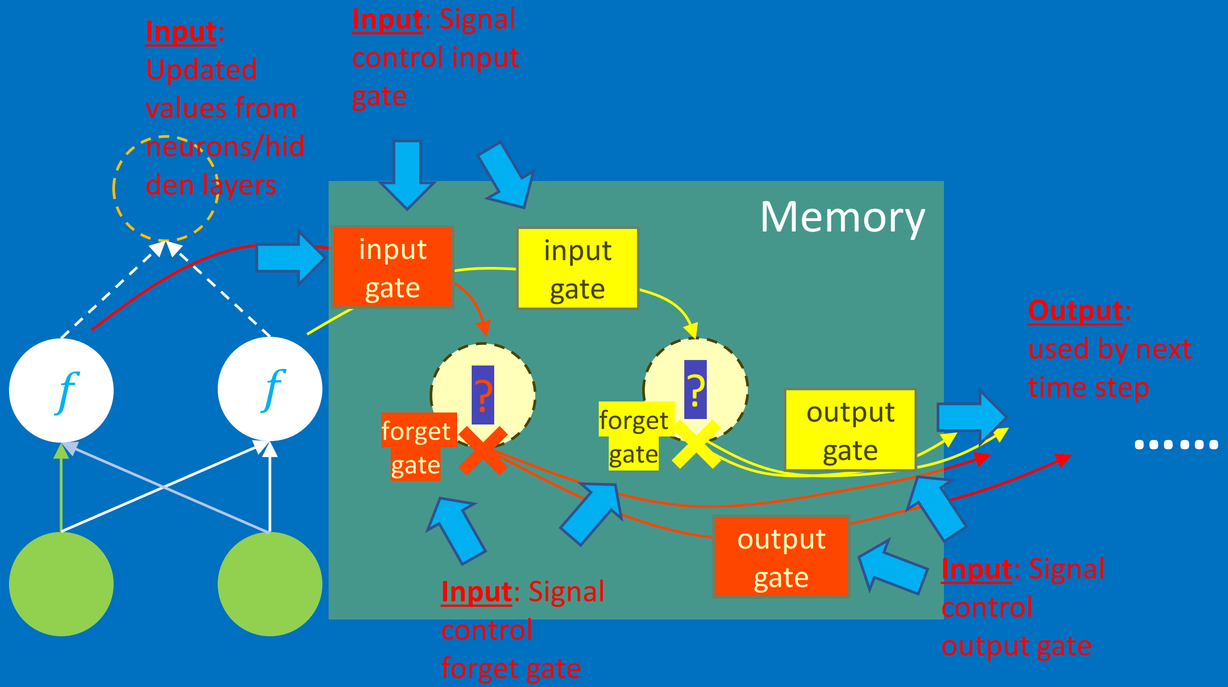
Note that for the input or output gates, we only controls whether the updated neurons for this timestep will be remembered (or used), however, for the forget gate, if it is “on”, then any previous learnt memory will be erased



Note the status of the gates are learnt by the model during training

The difference between Simple RNN and LSTM





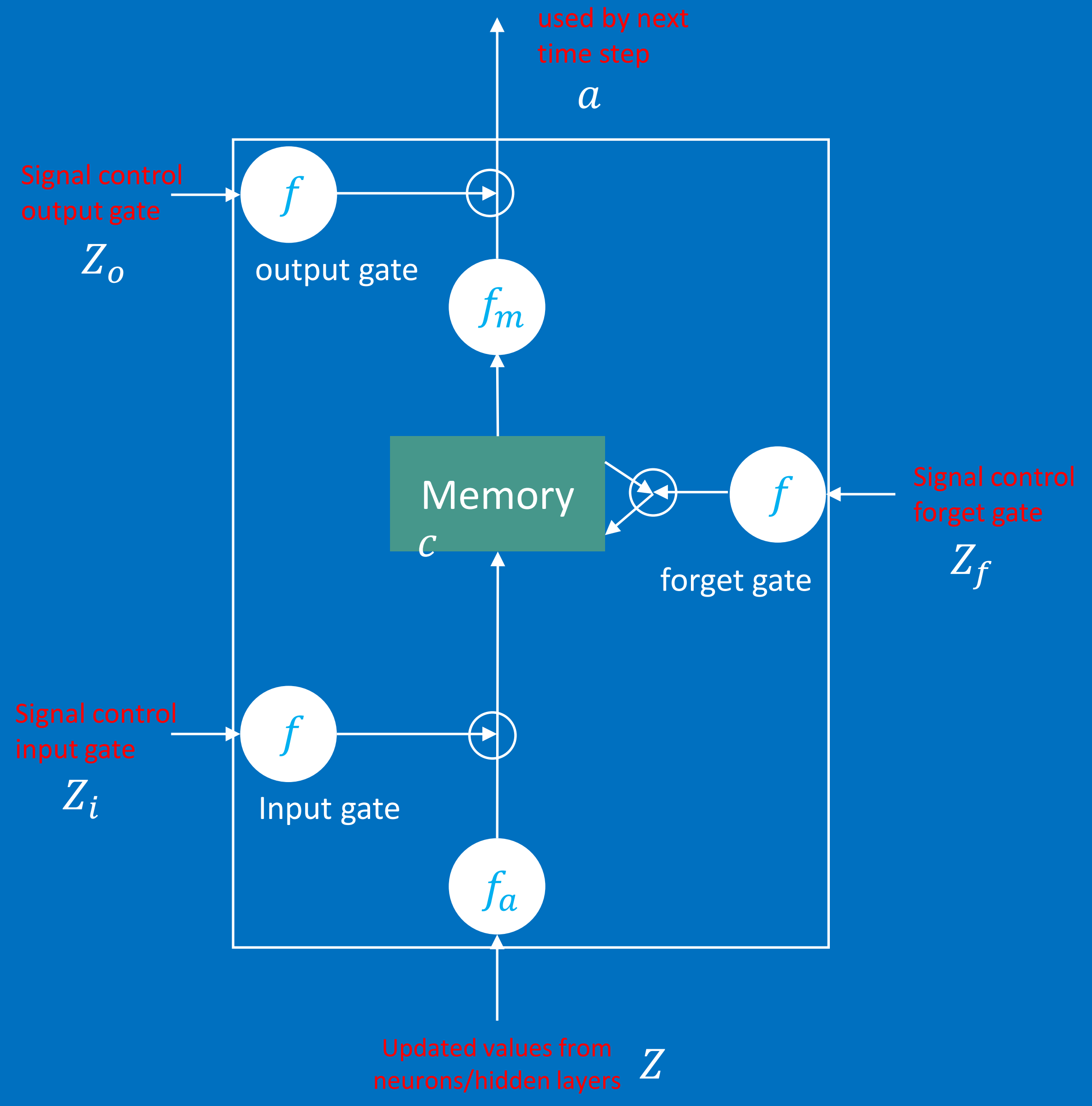
Simple RNN has:

- "1" input to be stored in the memory
- "1" output to be used by next time step of RNN (or other parts of network)

LSTM has:

- "4" input to be stored in the memory
- "1" output to be used by next time step of LSTM (or other parts of network)

How LSTM works



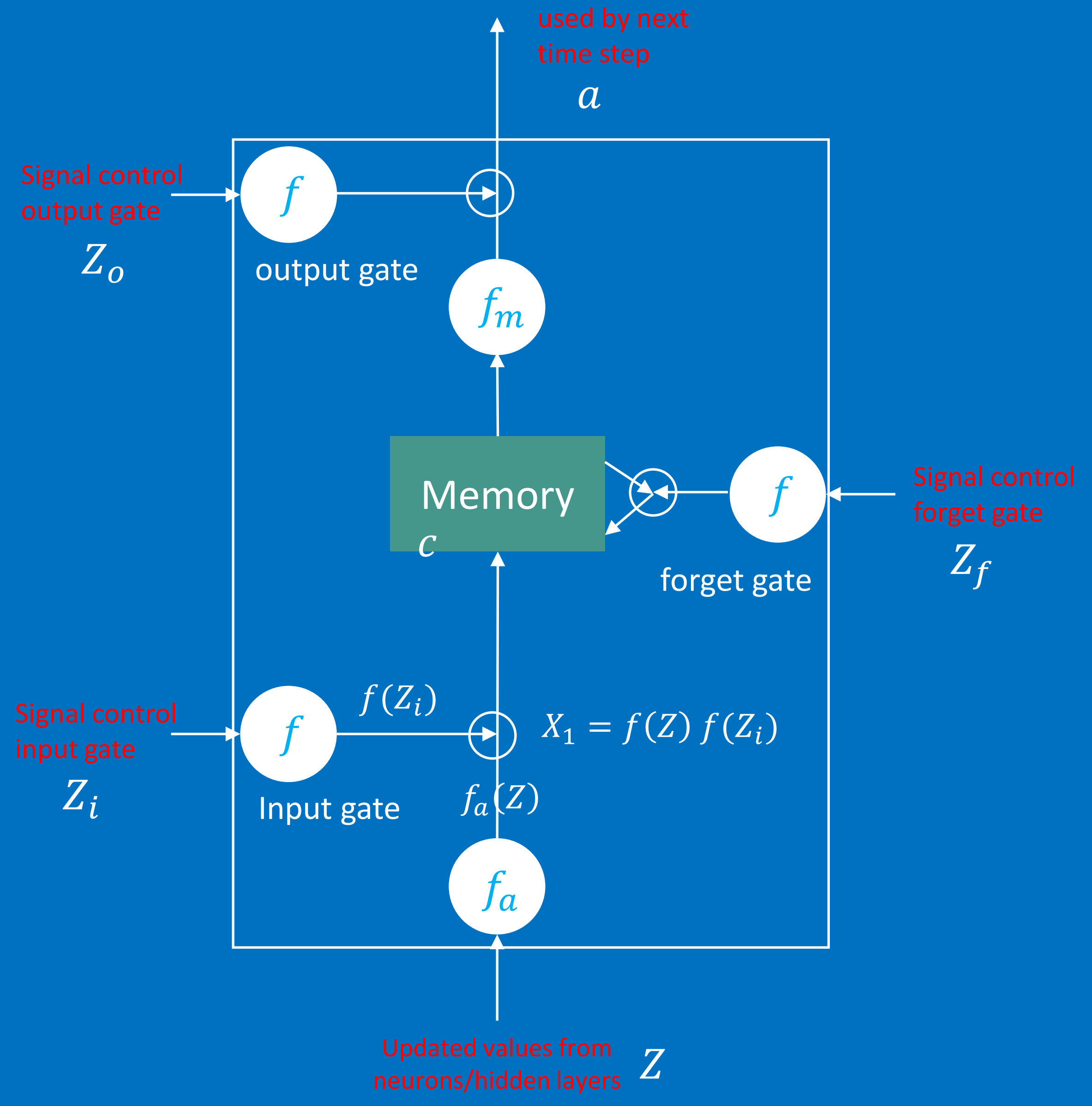
Step 1

Assuming that

- the input from external of this cell is called Z
- the signal controls input gate is called Z_i
- the signal controls forget gate is called Z_f
- the signal controls output gate is called Z_o
- the output is a
- f represents the activation function for gates
- f_a or f_m represent the activation functions used for non-gate related activities

Also, before this cell, the memory already has a value c

How LSTM works



Step 1

Assuming that

- the input from external of this cell is called Z
- the signal controls input gate is called Z_i
- the signal controls forget gate is called Z_f
- the signal controls output gate is called Z_o
- the output is a
- f represents the activation function for gates
- f_a or f_m represent the activation functions used for non-gate related activities

Also, before this cell, the memory already has a value c

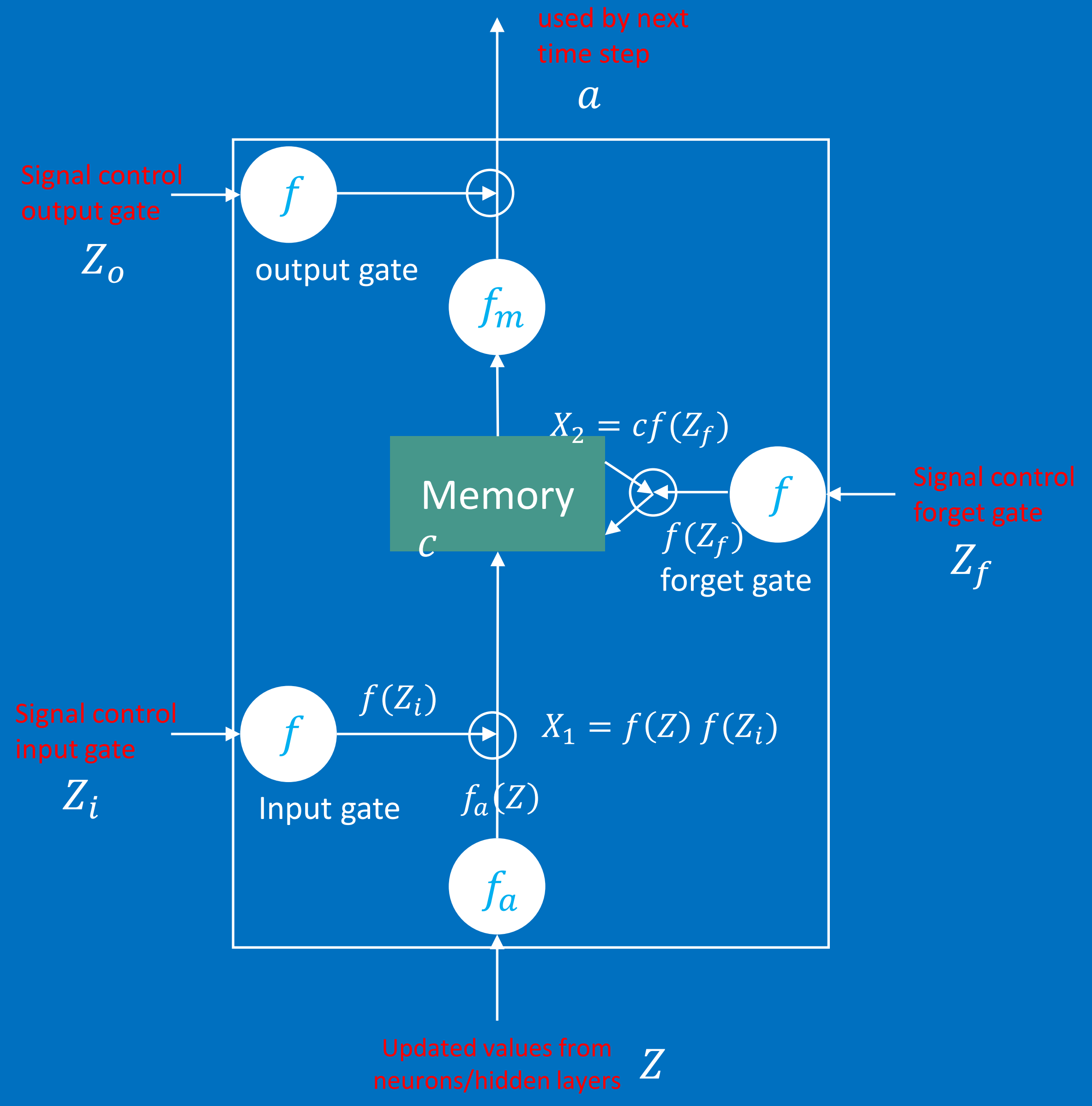
Step 2

Apply the f to Z and Z_i

Since f is a sigmoid function, so $f_a(Z)$ and $f(Z_i)$ are between 0 and 1

Creating a multiplication product: $X_1 = f_a(Z) f(Z_i)$

How LSTM works



Step 2

Apply the f to Z and Z_i

Since f is a sigmoid function, so$f_a(Z)$and $f(Z_i)$ are between 0 and 1

Creating a multiplication product: $X_1 = f_a(Z)f(Z_i)$

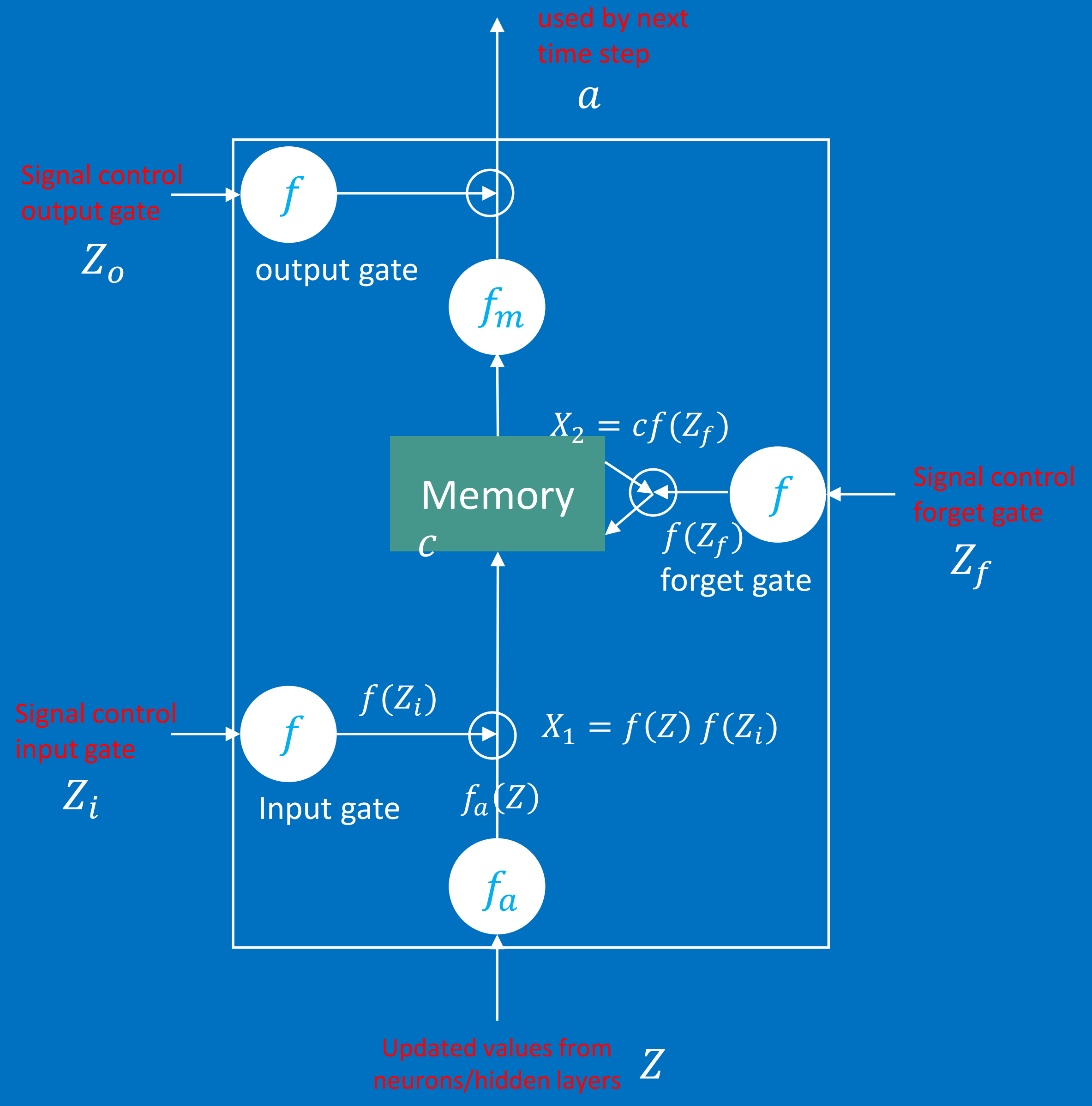
Step 3

Apply the f to Z_f: $f(Z_f)$

Creating a multiplication production between the existing memory value c and $f(Z_f)$:

$$X_2 = cf(Z_f)$$

How LSTM works



Step 2

Apply the f to Z and Z_i

Since f is a sigmoid function, so$f_a(Z)$and $f(Z_i)$ are between 0 and 1

Creating a multiplication product: $X_1 = f_a(Z)f(Z_i)$

Step 3

Apply the f to Z_f: $f(Z_f)$

Creating a multiplication production between the existing memory value c and $f(Z_f)$:

$$X_2 = \mathrm{c}f(Z_f)$$

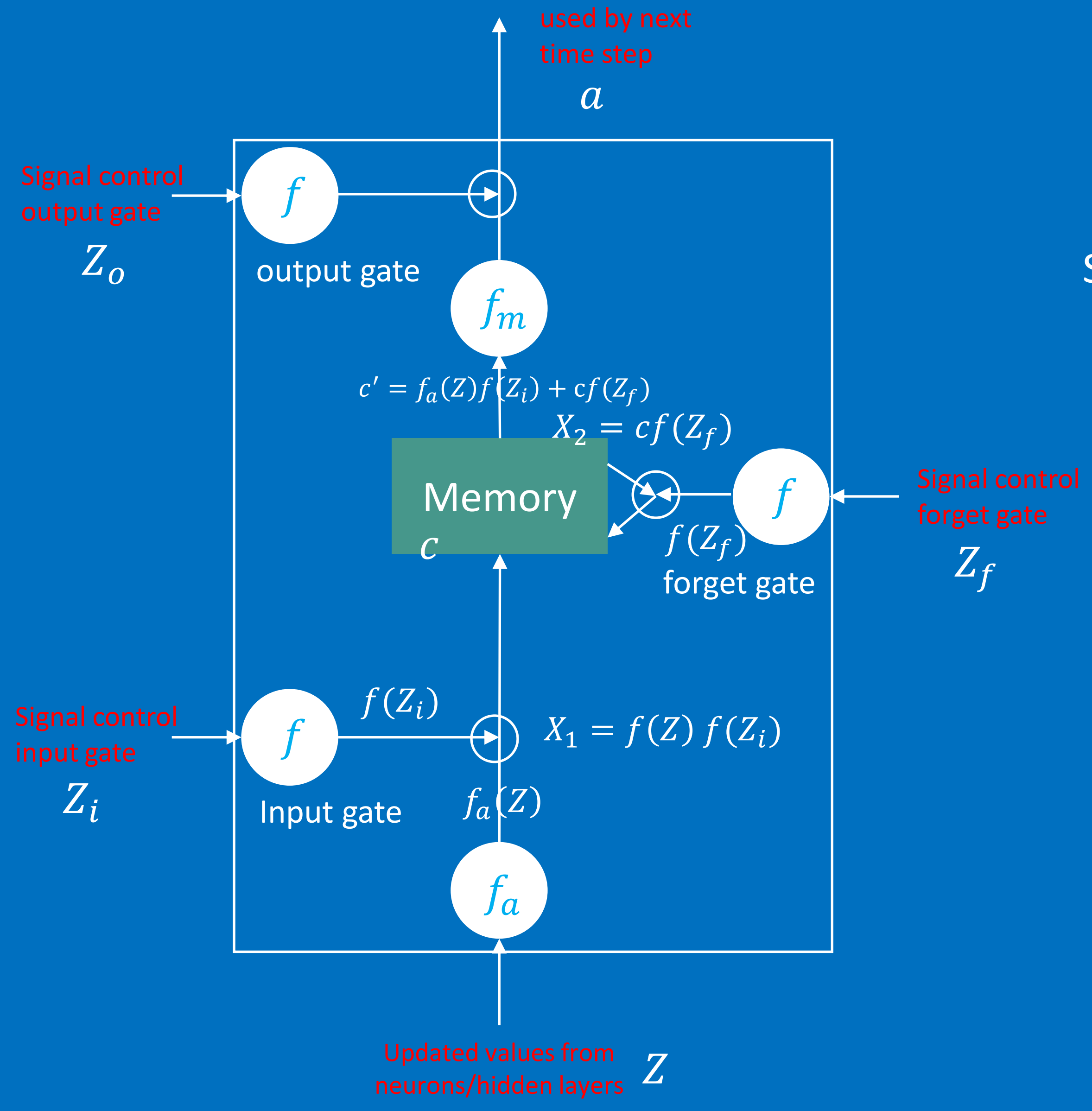
Step 4

Combining X_1 and X_2 as:

$$X_{1,2} = f_a(Z)f(Z_i) + \mathrm{c}f(Z_f)$$

So $X_{1,2}$ is the new value to be saved in the memory

How LSTM works



Step 3

Apply the f to Z_f: $f(Z_f)$

Creating a multiplication production between the existing memory value c and $f(Z_f)$:

$$X_2 = cf(Z_f)$$

Step 4

Combining X_1 and X_2 as:

$$c' = f_a(Z)f(Z_i) + cf(Z_f)$$

So c' is the new value to be saved in the memory

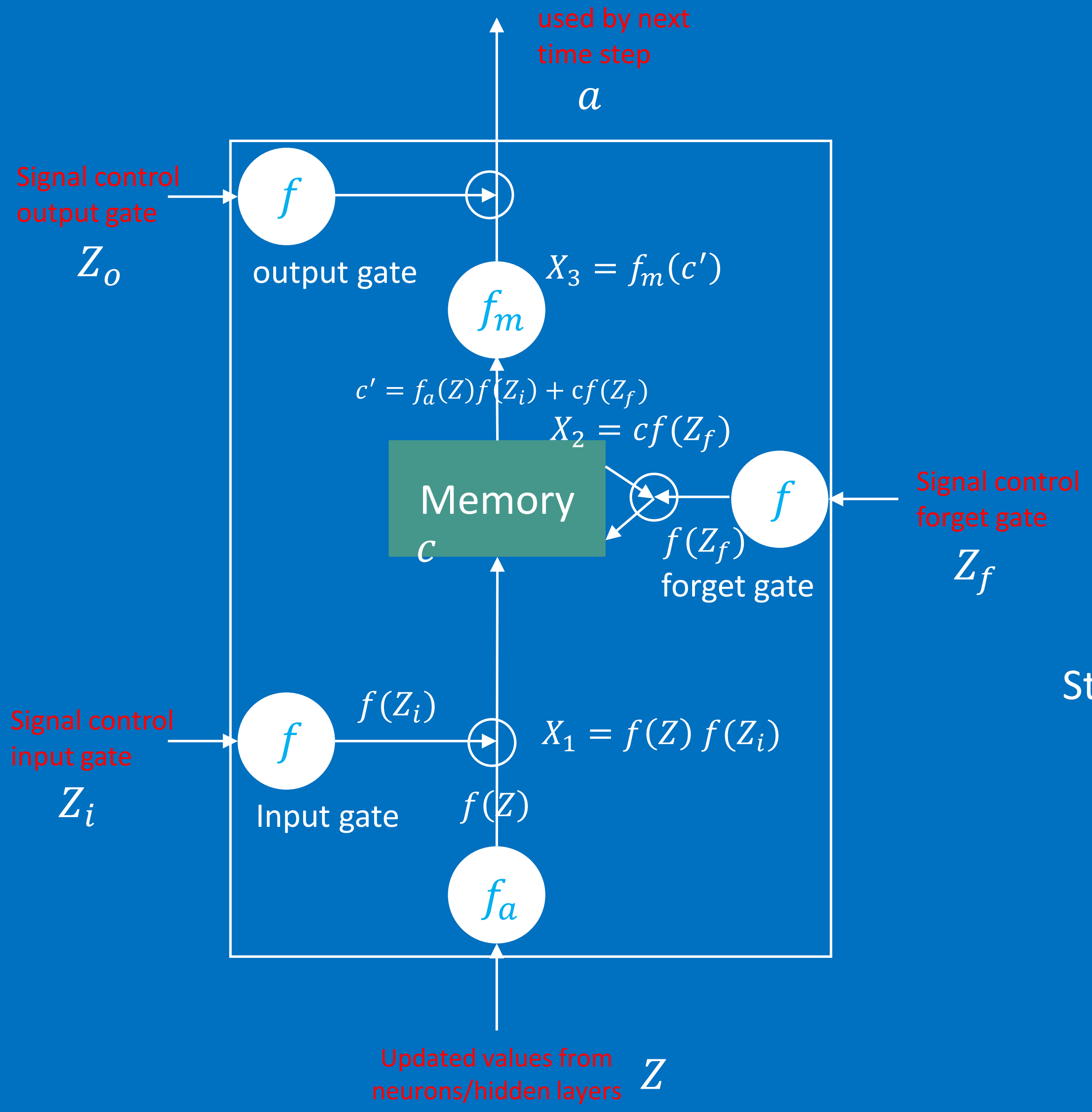
Usually f is a sigmoid function, so $f(Z_i)$ and $f(Z_f)$ is between 0 and 1

$f(x) = 0$: this means that the gate is closed (no data pass)

$f(x) = 1$: this means that the gate is open (no data pass)

For example, if $f(Z_f) = 1$, c will be kept in the memory, and if it is zero, c will be erased

How LSTM works



Step 4

Combining X_1 and X_2 as:

$$c' = f_a(Z)f(Z_i) + cf(Z_f)$$

So c' is the new value to be saved in the memory

Usually f is a sigmoid function, so $f(Z_i)$ and $f\left(Z_f\right)$ is between 0 and 1

$f(x) = 0$: this means that the gate is closed (no data pass)

$f(x) = 1$: this means that the gate is open (no data pass)

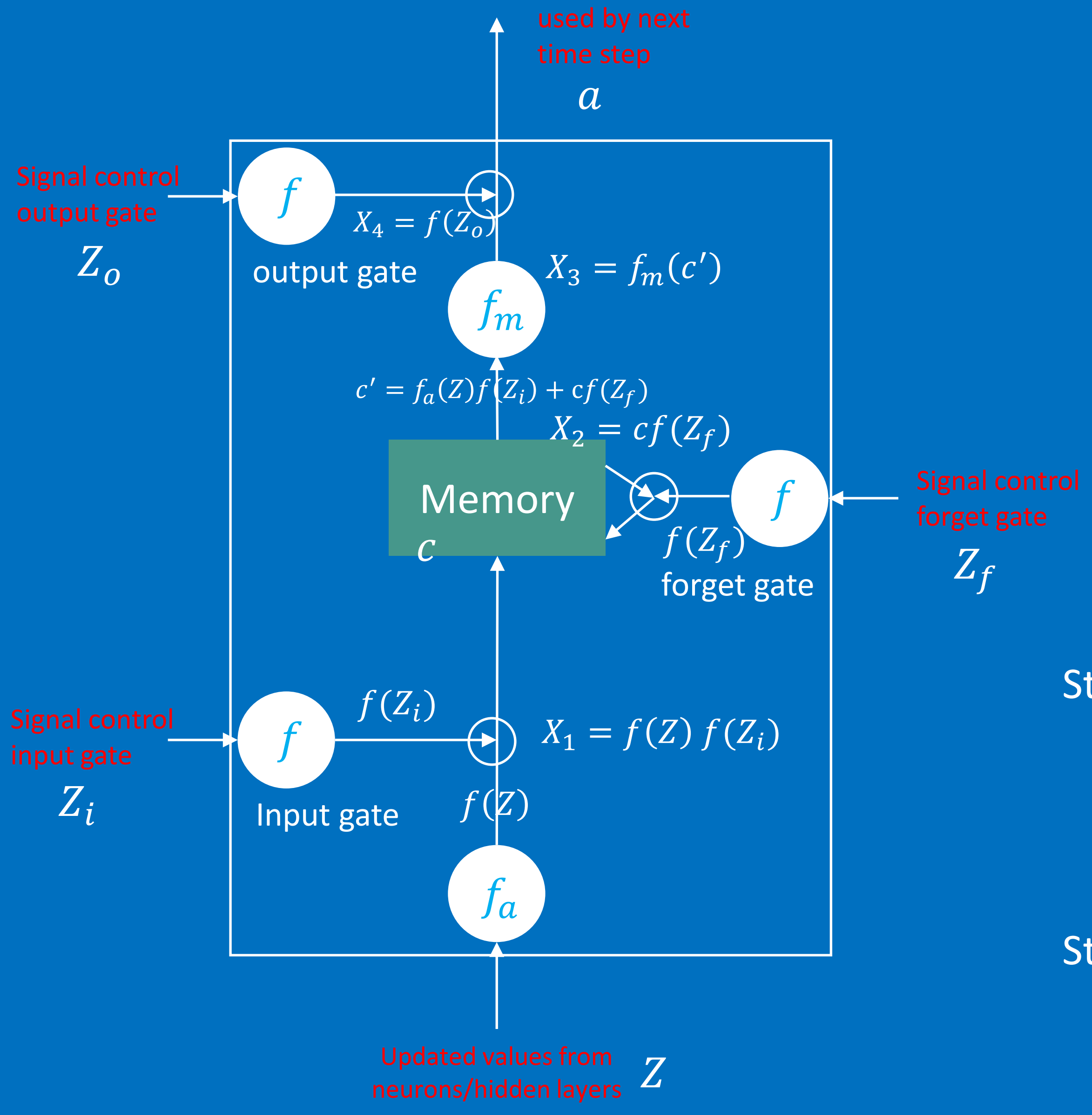
For example, if $f\left(Z_f\right) = 1$, c will be kept in the memory, and if it is zero, c will be erased

Step 5

Taking the c' and apply it to the activation function f_m

$$X_3 = f_m(c')$$

How LSTM works



Step 4

Combining X_1 and X_2 as:

$$c' = f_a(Z)f(Z_i) + cf(Z_f)$$

So c' is the new value to be saved in the memory

Usually f is a sigmoid function, so $f(Z_i)$ and $f(Z_f)$ is between 0 and 1

$f(x) = 0$: this means that the gate is closed (no data pass)

$f(x) = 1$: this means that the gate is open (no data pass)

For example, if $f(Z_f) = 1$, c will be kept in the memory, and if it is zero, c will be erased

Step 5

Taking the c' and apply it to the activation function f_m

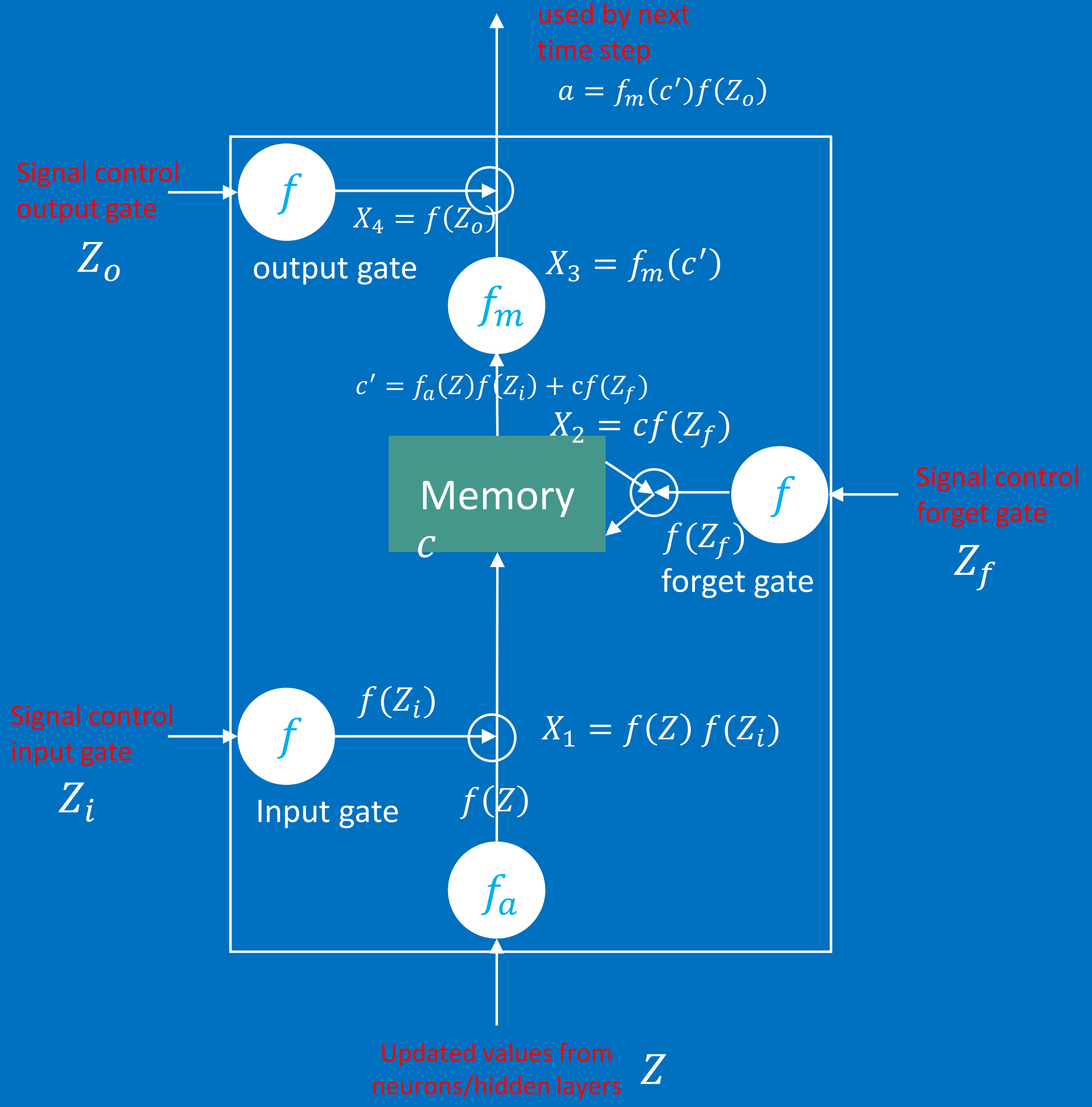
$$X_3 = f_m(c')$$

Step 6

The output gate can be represented as:

$$X_4 = f(Z_o)$$

How LSTM works



Step 5

Taking the c' and apply it to the activation function f_m

$$X_3 = f_m(c')$$

Step 6

The output gate can be represented as:

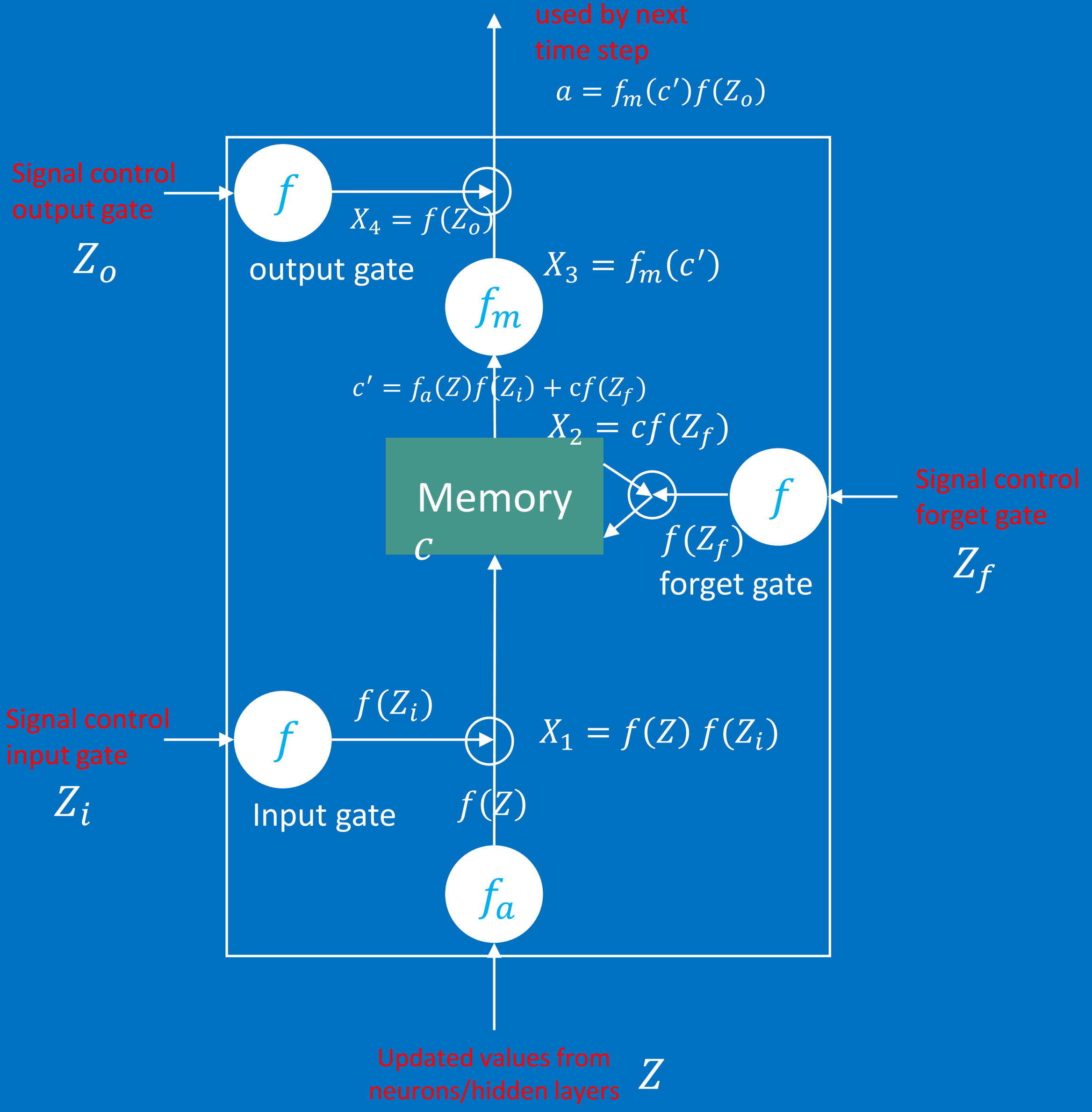
$$X_4 = f(Z_o)$$

Step 7

Finally we can get the output as

$$a = f_m(c')f(Z_o)$$

How LSTM works



In a summary, the output from a LSTM is produced by

output gate

input gate forget gate

$$a = f_m\big(f_a(Z)f(Z_i) + cf(Z_f)\big)f(Z_o)$$

contribution from input

contribution from previous stored value in memory

Contribution from the current input + memory

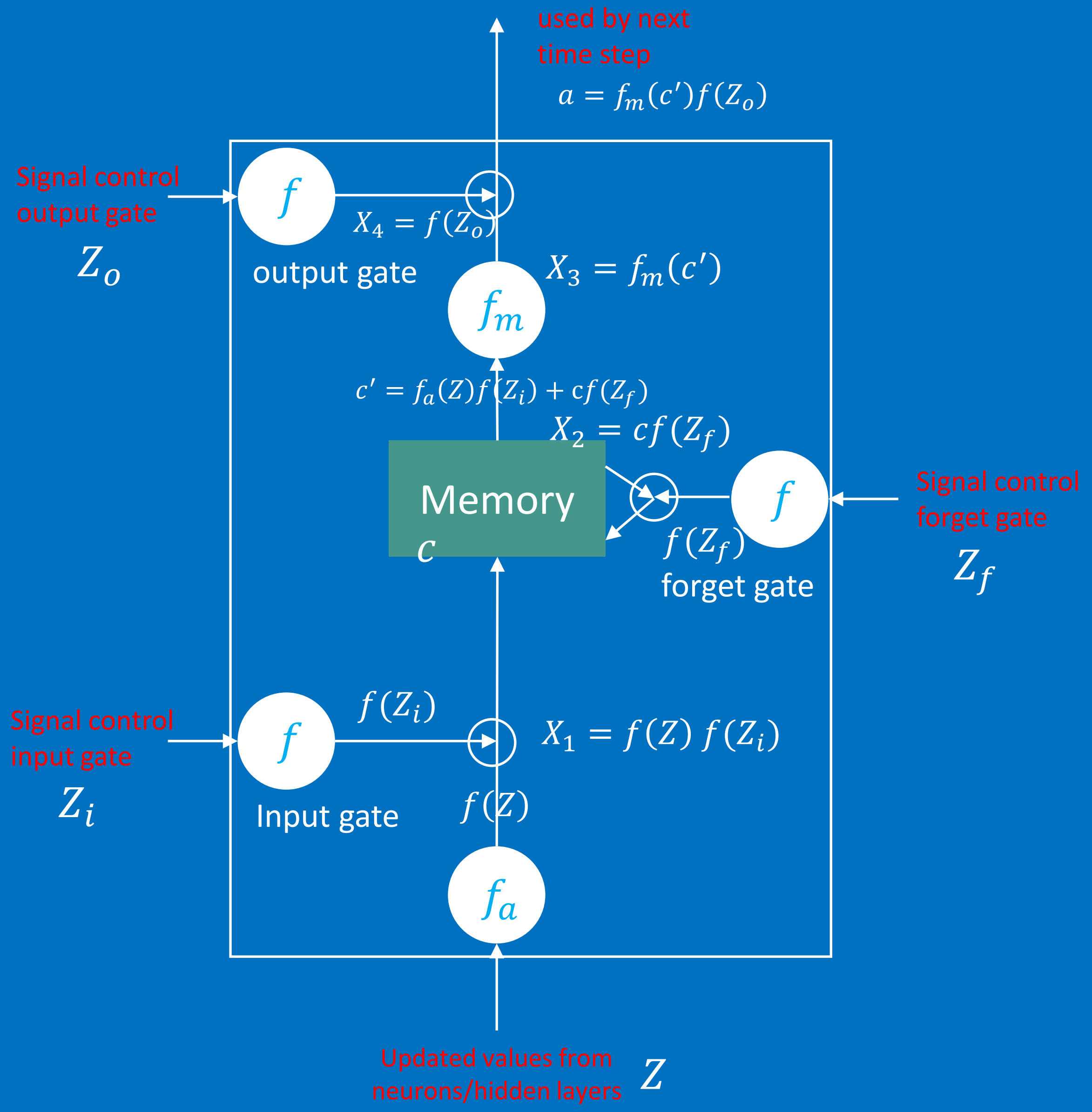
output

Decide whether：

- erase the old neuron value in the memory
- using “input” to update (or initialize) the neuron value in the memory

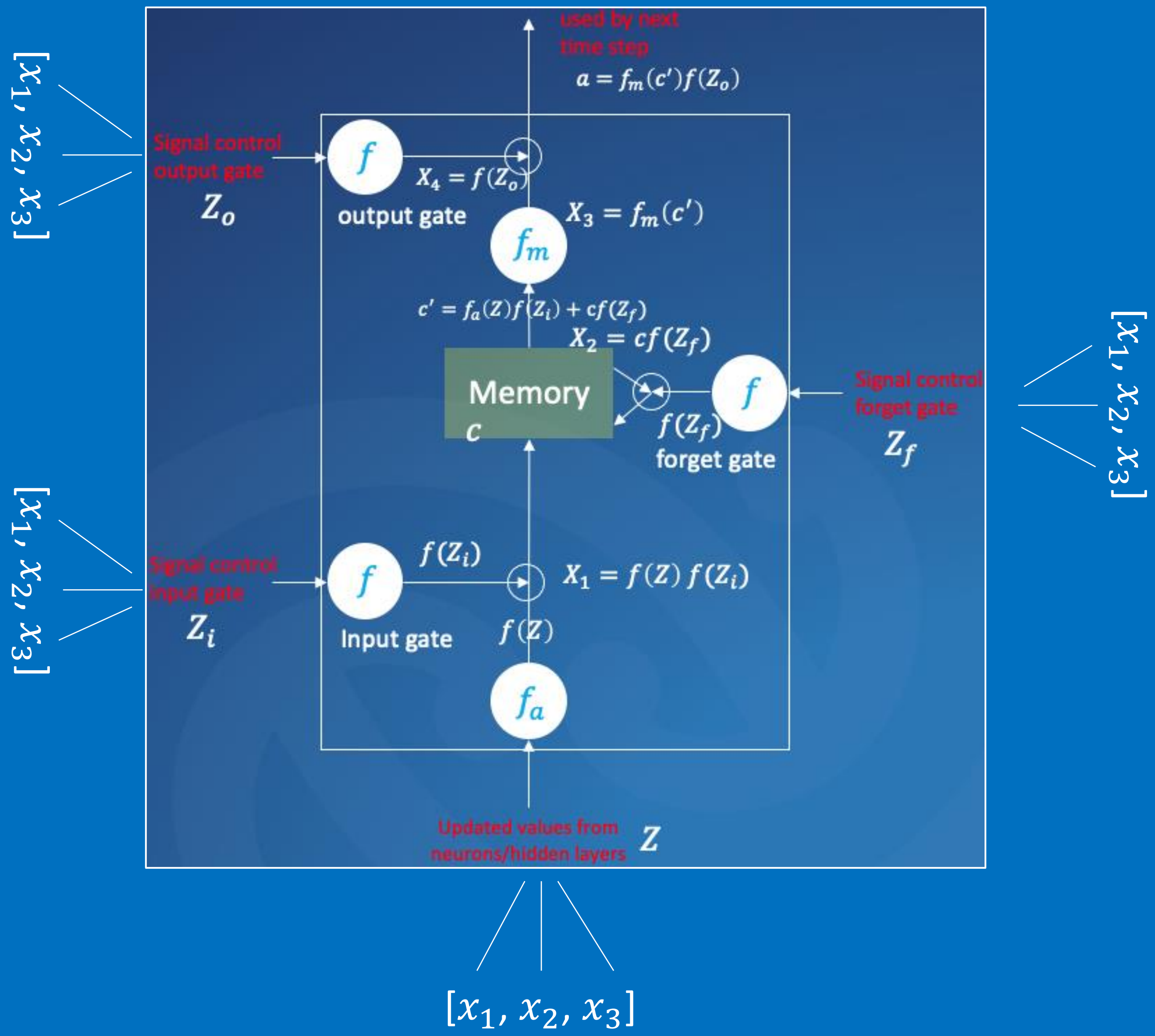
How LSTM works

The next question is that where Z, Z_i, Z_f and Z_o comes from ?



They are actually all determined by the input data

How LSTM works The next question is that where Z, Z_i, Z_f and Z_o comes from ?

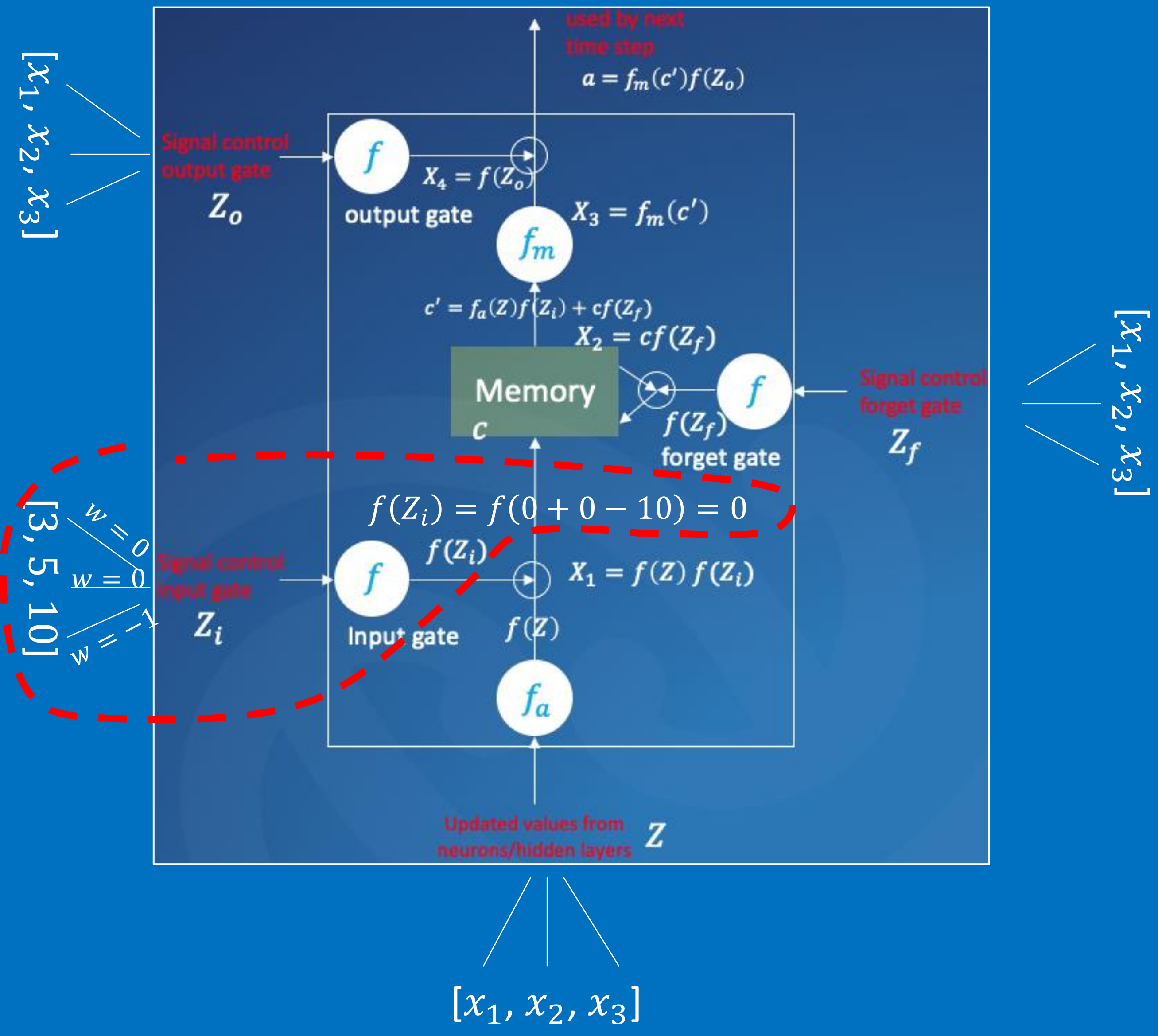


For example, assuming that at a particular time step, we have inputs: $[x_1, x_2, x_3]$

So apparently, the signals control all the gates are the input

How LSTM works

The next question is that where Z, Z_i, Z_f and Z_o comes from ?



For example, assuming that at a particular time step, we have inputs: $[x_1, x_2, x_3]$

So apparently, the signals control all the gates are the input

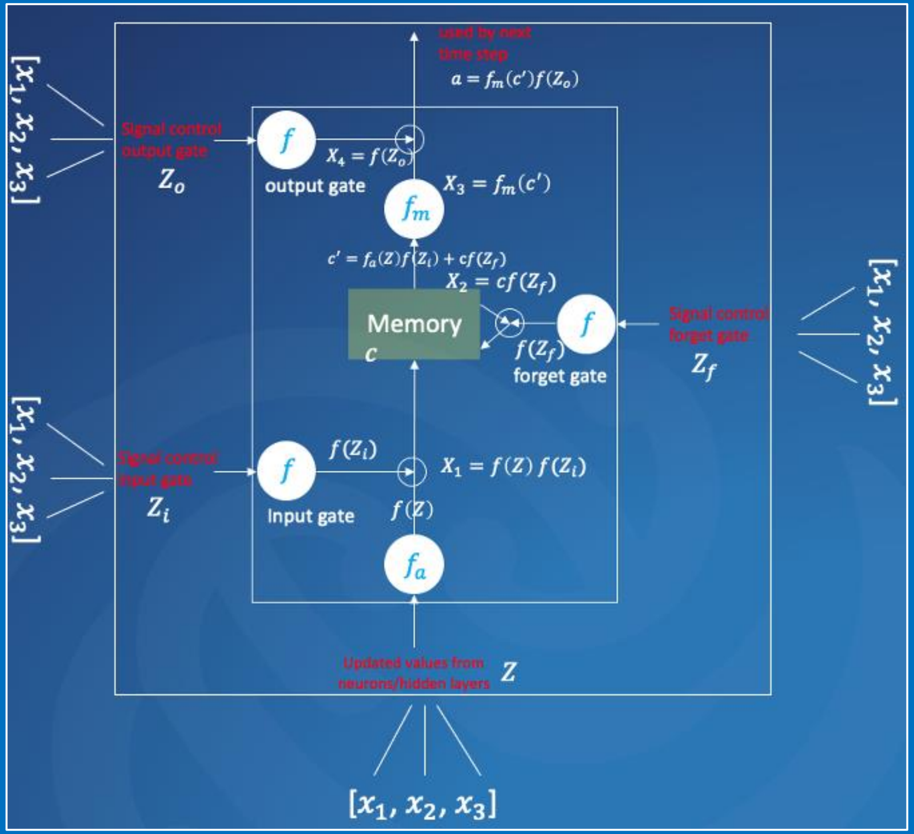
> e.g., for the input gate, given that we have the input [3,5,10], the weights are [0, 0, -1], then the input gate is ”-10” (via the sigmoid function, it is “0”) so the gate will be closed

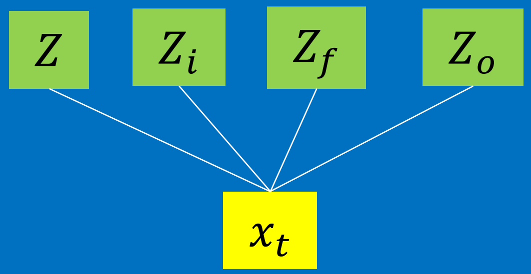
Note that those weights are obtained via the backpropagation training

How LSTM works

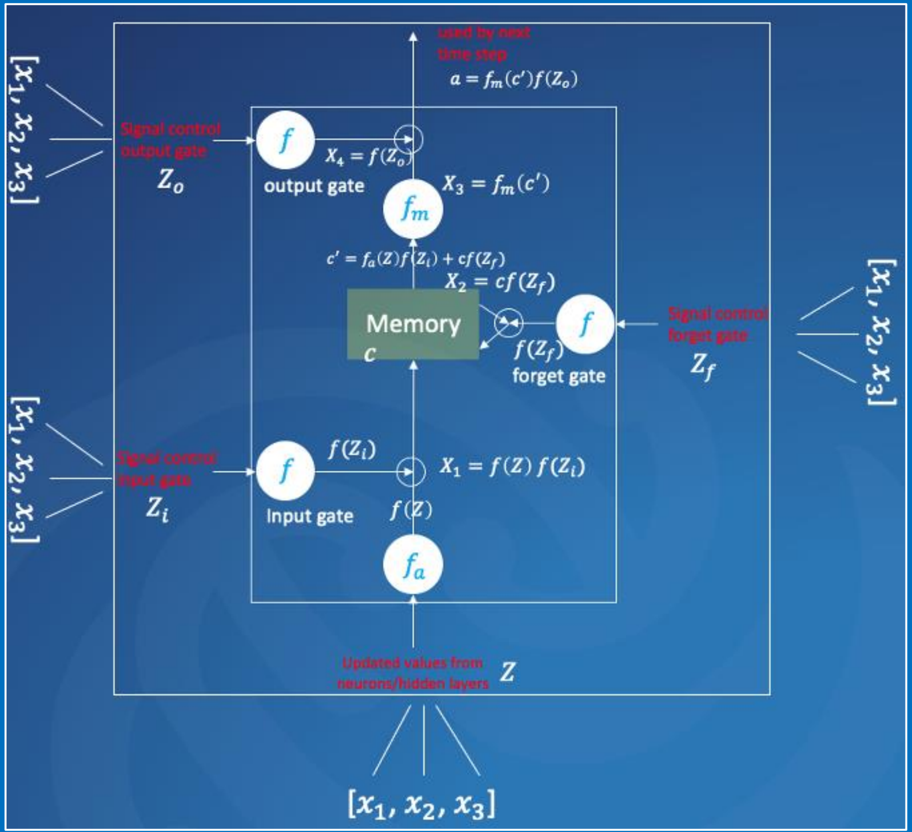
So for one cell of LSTM, the workflow can be represented as

(1) the input is multiplied by weights to form the inputs for different gates and the input itself





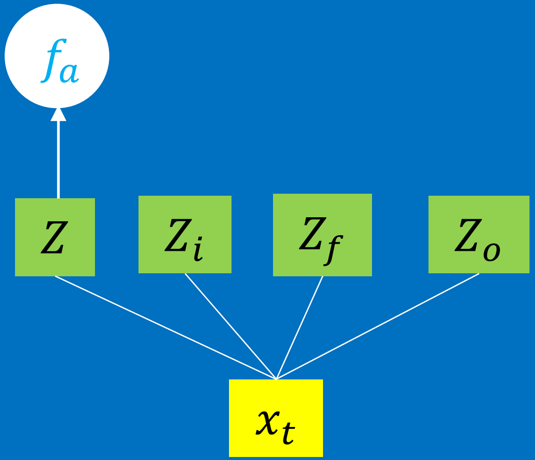
How LSTM works



So for one cell of LSTM, the workflow can be represented as

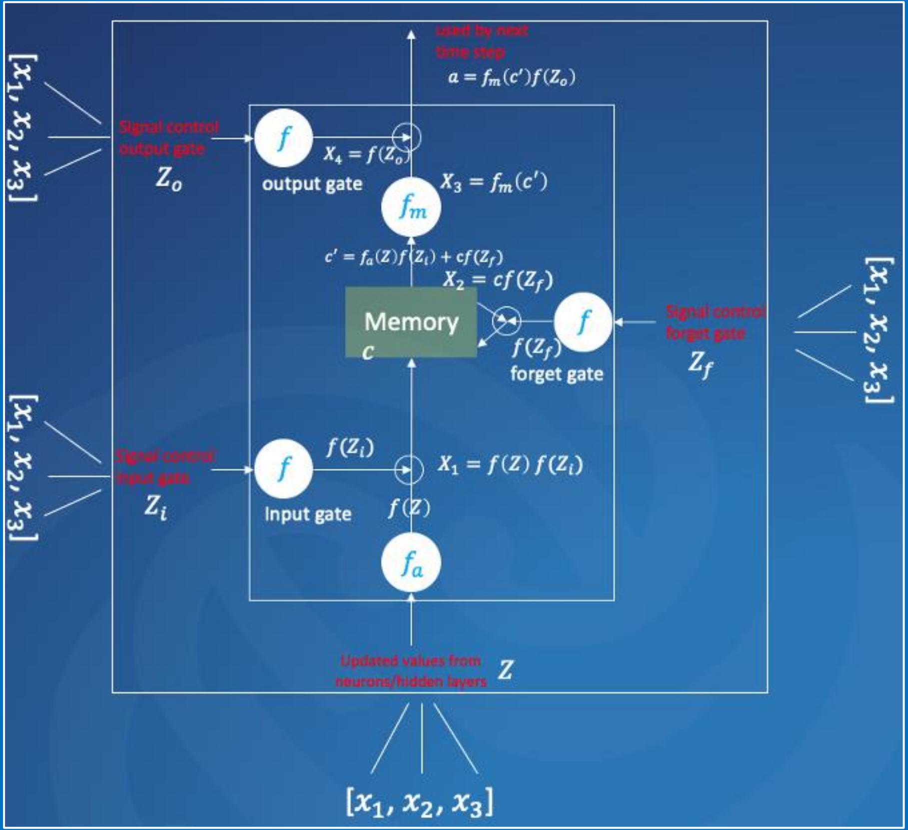
(1) the input is multiplied by weights to form the inputs for different gates and the input itself

(2) the input multiplied by a activation function f_a



How LSTM works

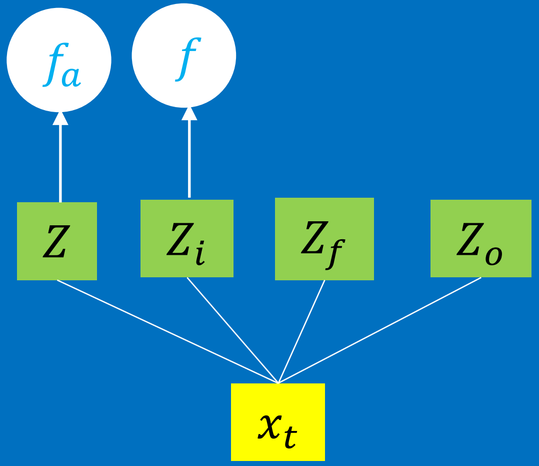
So for one cell of LSTM, the workflow can be represented as



(1) the input is multiplied by weights to form the inputs for different gates and the input itself

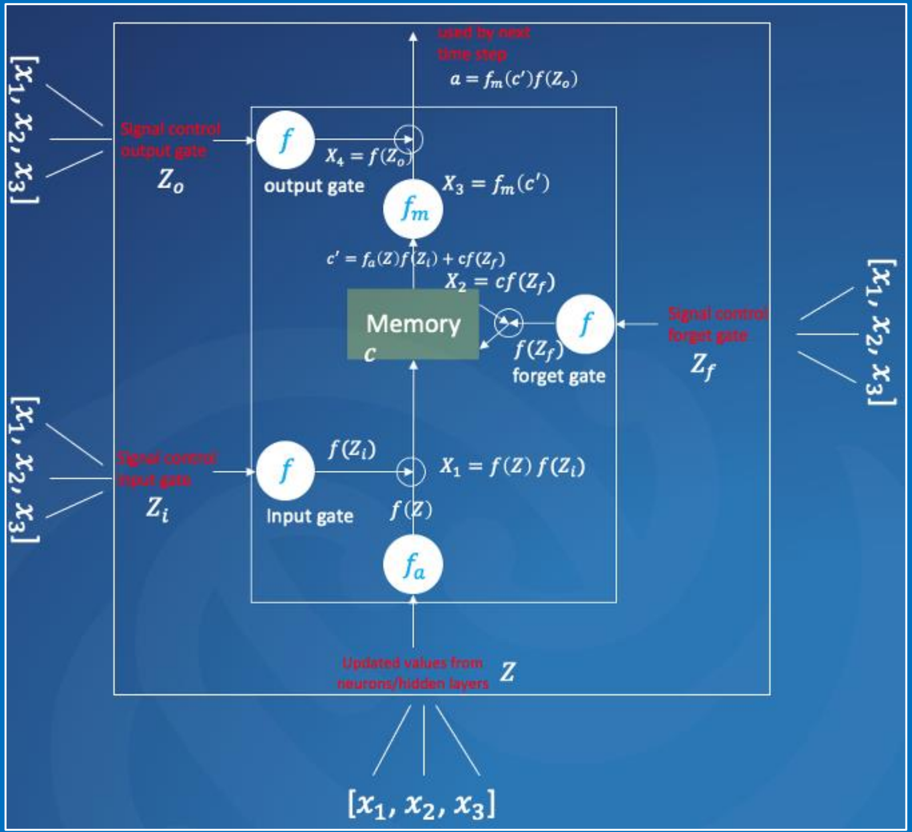
(2) the input multiplied by a activation function f_a

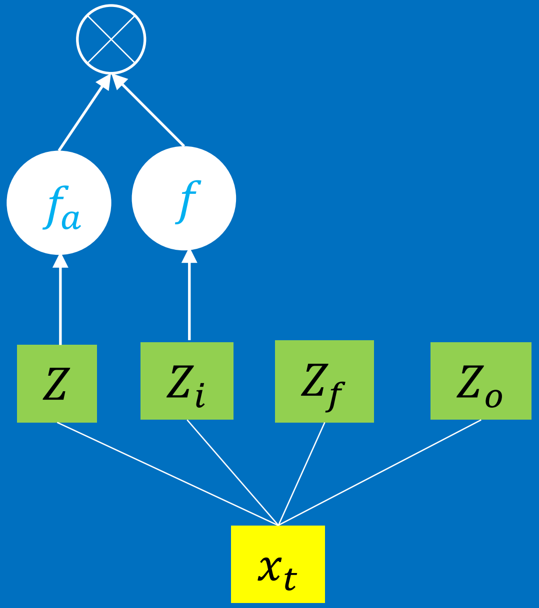
(3) the input gate is multiplied by a activation function f



How LSTM works

So for one cell of LSTM, the workflow can be represented as





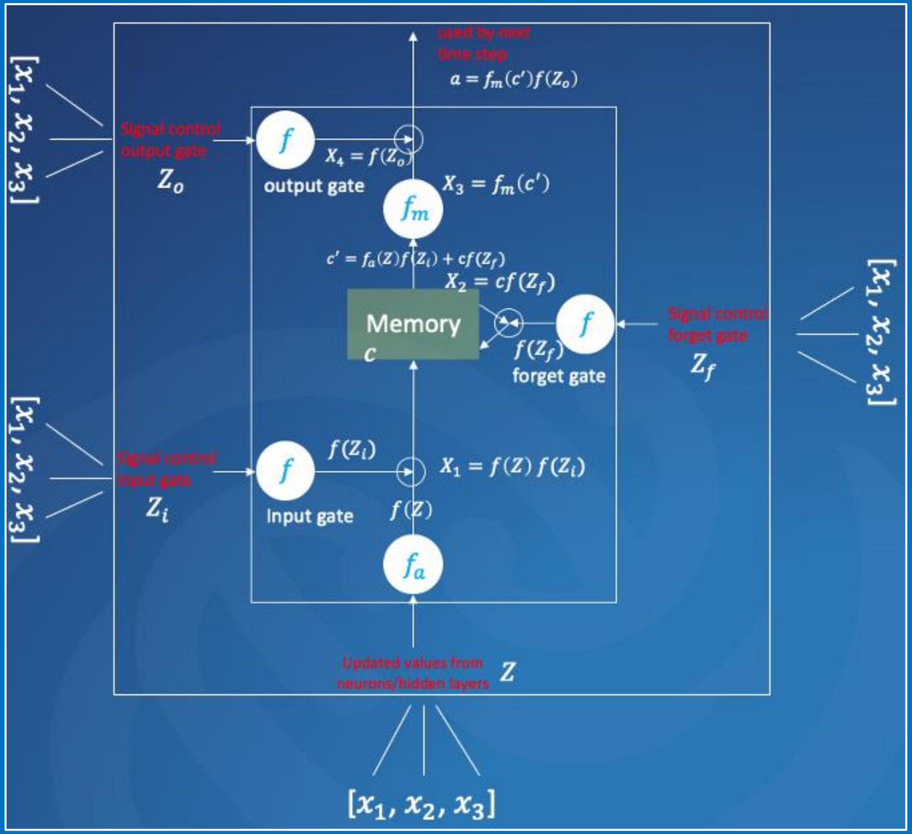
(1) the input is multiplied by weights to form the inputs for different gates and the input itself

(2) the input multiplied by a activation function f_a

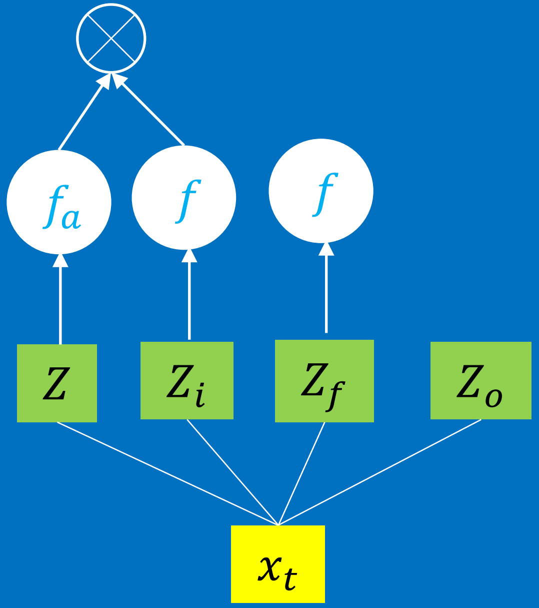
(3) the input gate is multiplied by a activation function f

(4) the output from (2) and (3) are multiplied together

How LSTM works



So for one cell of LSTM, the workflow can be represented as



(1) the input is multiplied by weights to form the inputs for different gates and the input itself

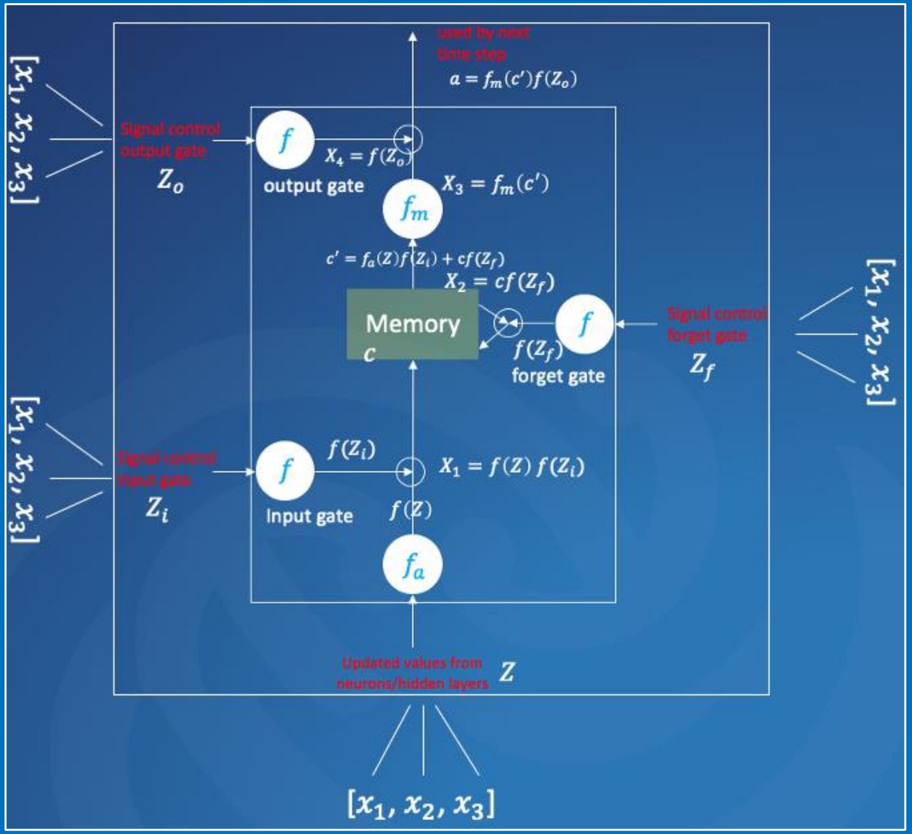
(2) the input multiplied by a activation function f_a

(3) the input gate is multiplied by a activation function f

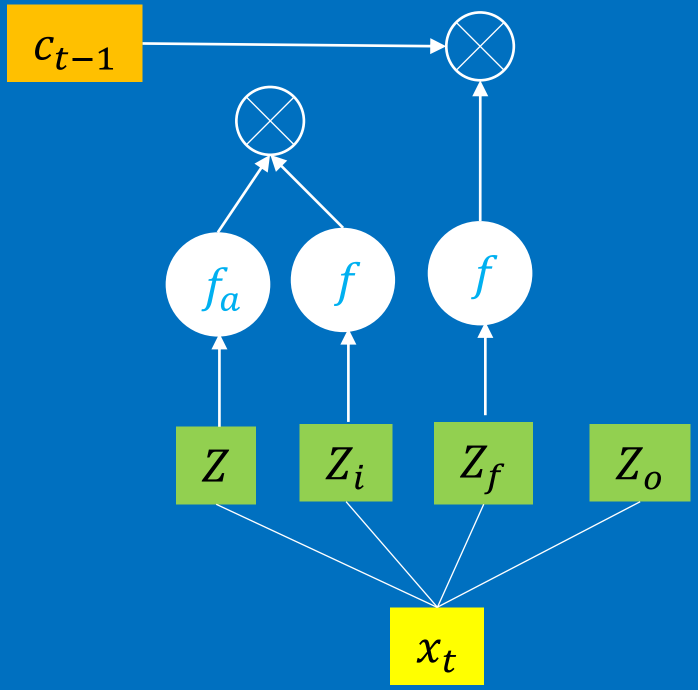
(4) the output from (2) and (3) are multiplied together

(5) the forget gate is multiplied by a activation function f

How LSTM works



So for one cell of LSTM, the workflow can be represented as



(1) the input is multiplied by weights to form the inputs for different gates and the input itself

(2) the input multiplied by a activation function f_a

(3) the input gate is multiplied by a activation function f

(4) the output from (2) and (3) are multiplied together

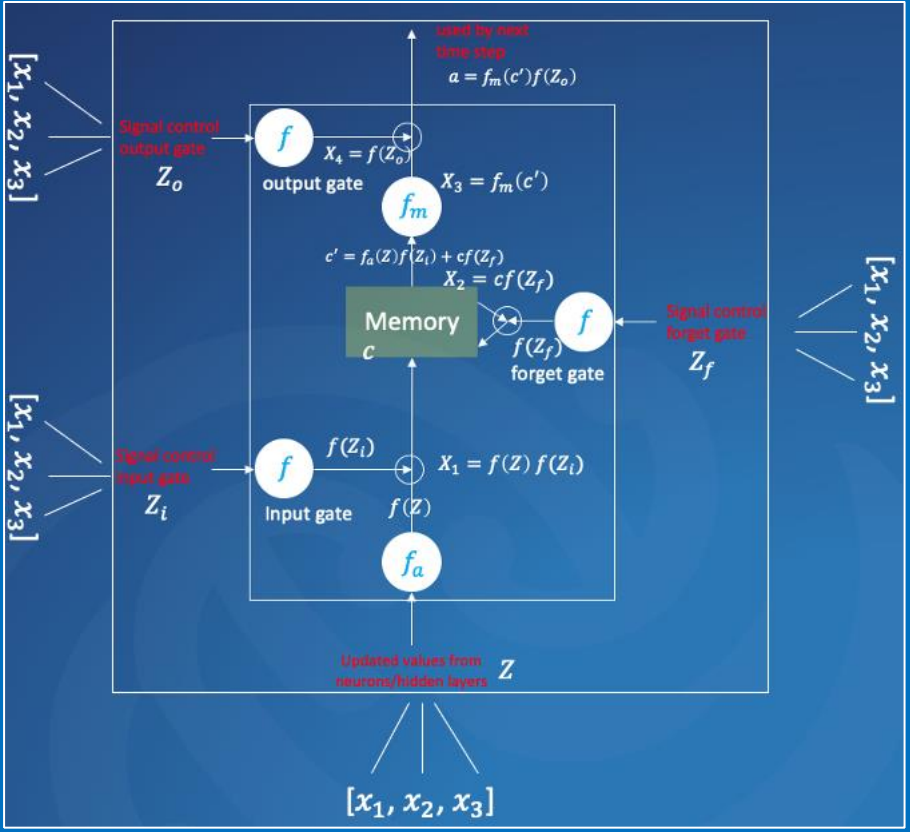
(5) the forget gate is multiplied by a activation function f

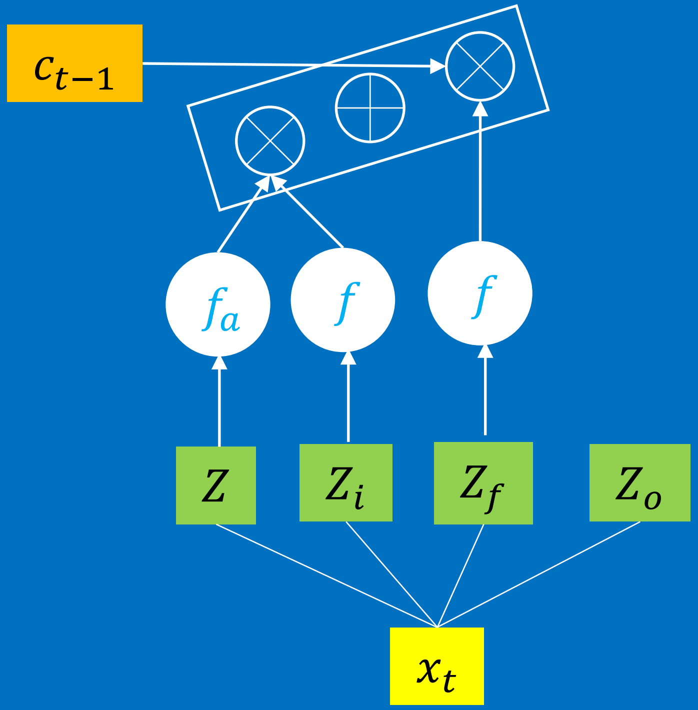
(6) Take the memory value from last time step c_{t-1}

(7) the output from (5) and (6) are multiplied together

How LSTM works

So for one cell of LSTM, the workflow can be represented as





(1) the input is multiplied by weights to form the inputs for different gates and the input itself

(2) the input multiplied by a activation function f_a

(3) the input gate is multiplied by a activation function f

(4) the output from (2) and (3) are multiplied together

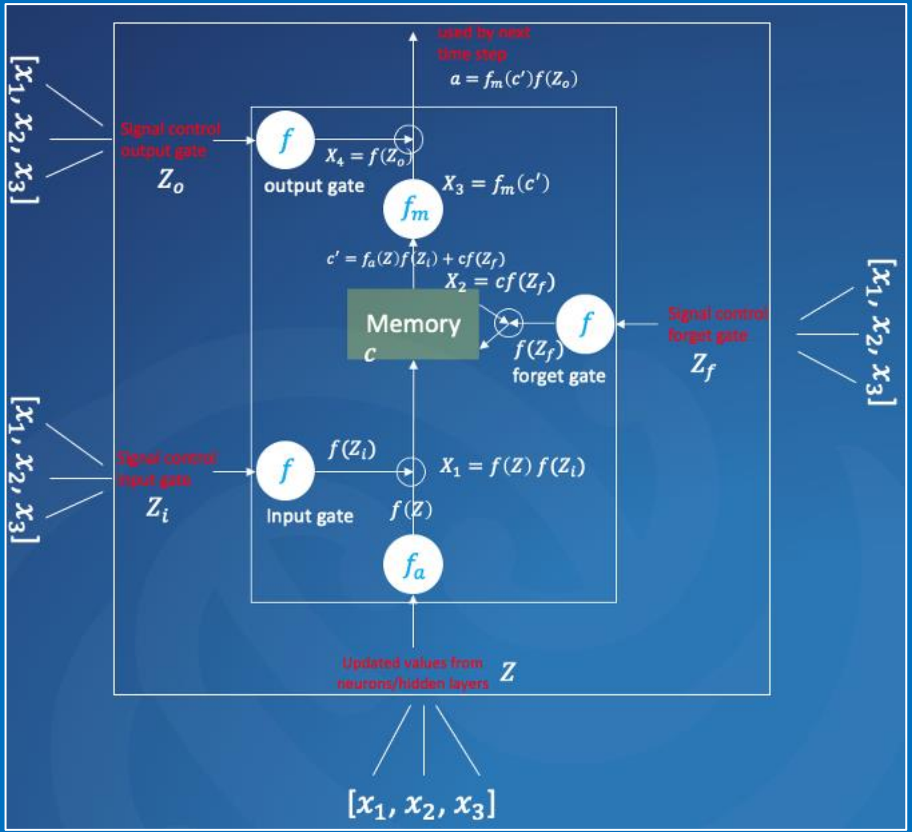
(5) the forget gate is multiplied by a activation function f

(6) Take the memory value from last time step c_{t-1}

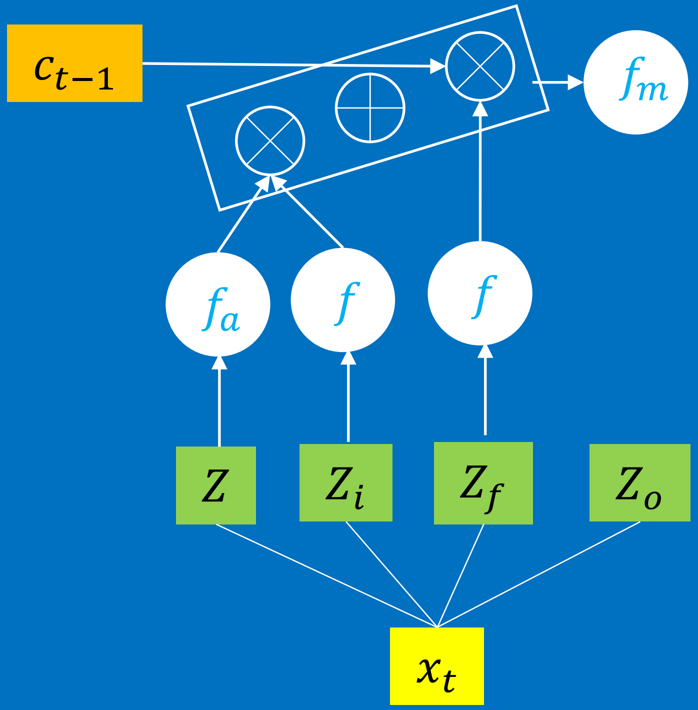
(7) the output from (5) and (6) are multiplied together

(8) the output from (4) and (7) are added together

How LSTM works



So for one cell of LSTM, the workflow can be represented as



(1) the input is multiplied by weights to form the inputs for different gates and the input itself

(2) the input multiplied by a activation function f_a

(3) the input gate is multiplied by a activation function f

(4) the output from (2) and (3) are multiplied together

(5) the forget gate is multiplied by a activation function f

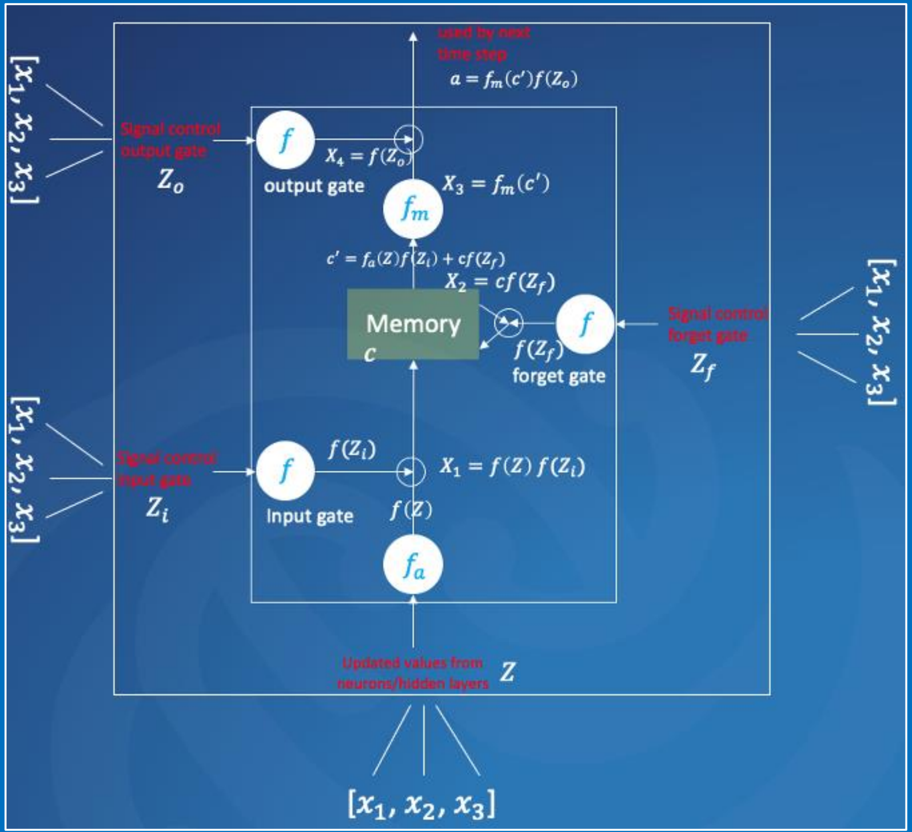
(6) Take the memory value from last time step c_{t-1}

(7) the output from (5) and (6) are multiplied together

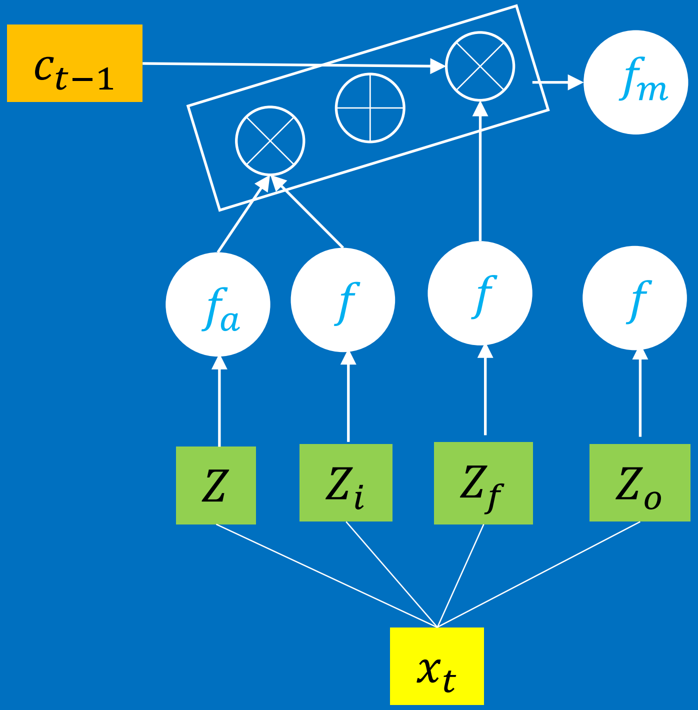
(8) the output from (4) and (7) are added together

(9) Apply the activation function f_m to the output of (8)

How LSTM works



So for one cell of LSTM, the workflow can be represented as



(1) the input is multiplied by weights to form the inputs for different gates and the input itself

(2) the input multiplied by a activation function f_a

(3) the input gate is multiplied by a activation function f

(4) the output from (2) and (3) are multiplied together

(5) the forget gate is multiplied by a activation function f

(6) Take the memory value from last time step c_{t-1}

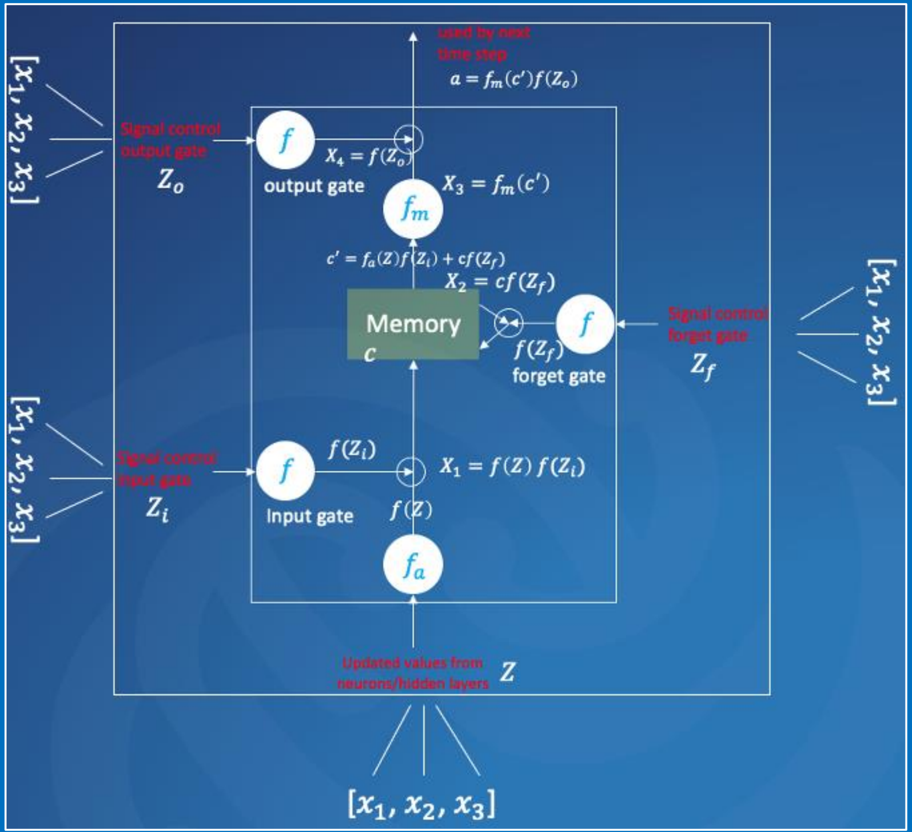
(7) the output from (5) and (6) are multiplied together

(8) the output from (4) and (7) are added together

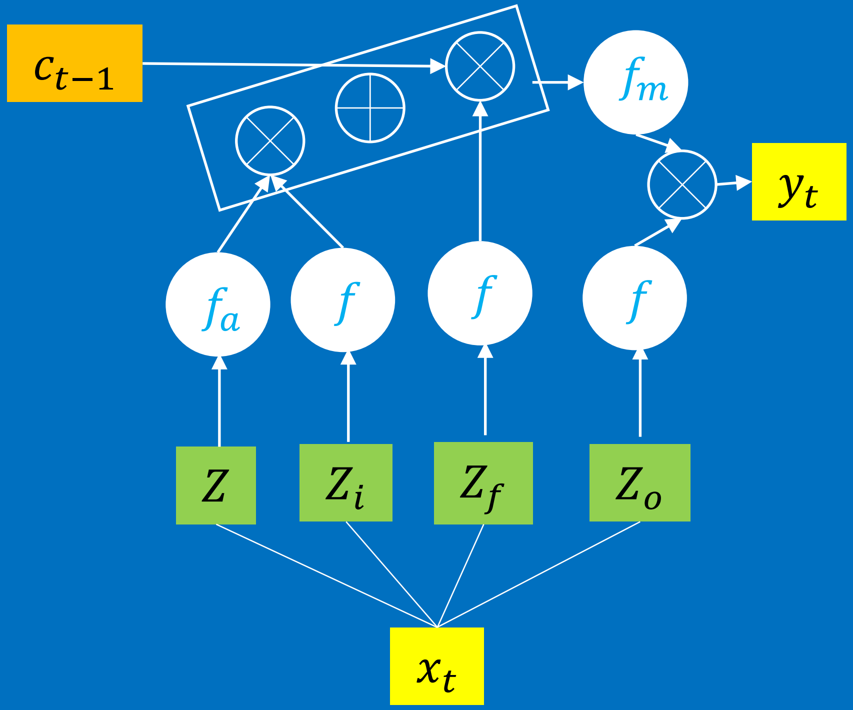
(9) Apply the activation function f_m to the output of (8)

(10) Apply the activation function f to the output gate

How LSTM works



So for one cell of LSTM, the workflow can be represented as



(1) the input is multiplied by weights to form the inputs for different gates and the input itself

(2) the input multiplied by a activation function f_a

(3) the input gate is multiplied by a activation function f

(4) the output from (2) and (3) are multiplied together

(5) the forget gate is multiplied by a activation function f

(6) Take the memory value from last time step c_{t-1}

(7) the output from (5) and (6) are multiplied together

(8) the output from (4) and (7) are added together

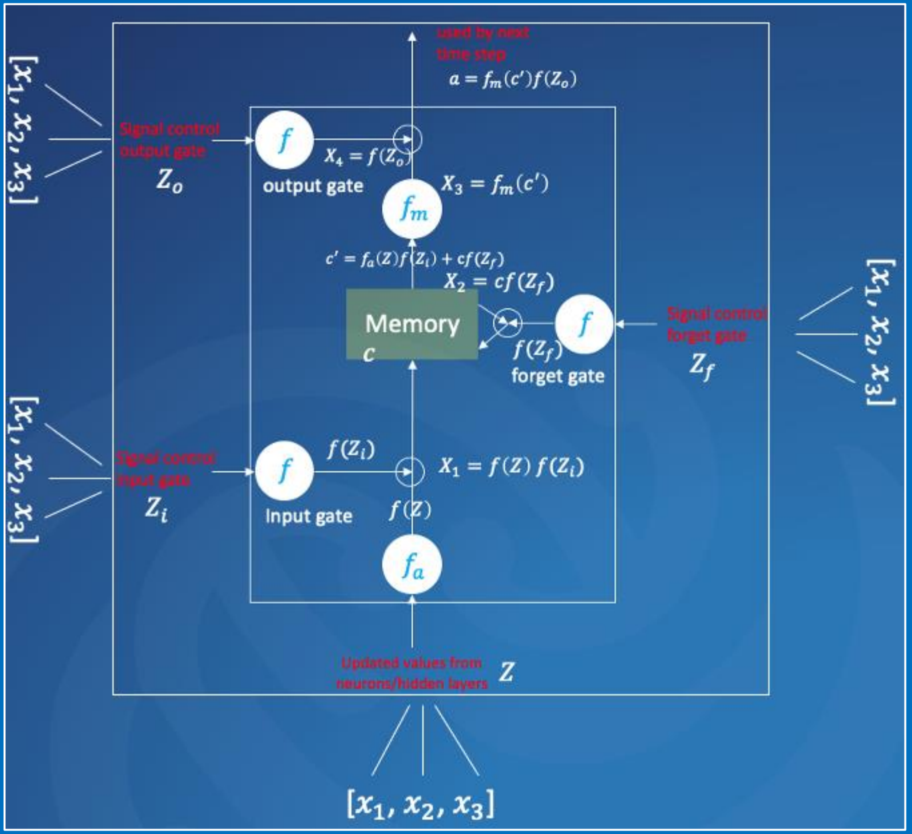
(9) Apply the activation function f_m to the output of (8)

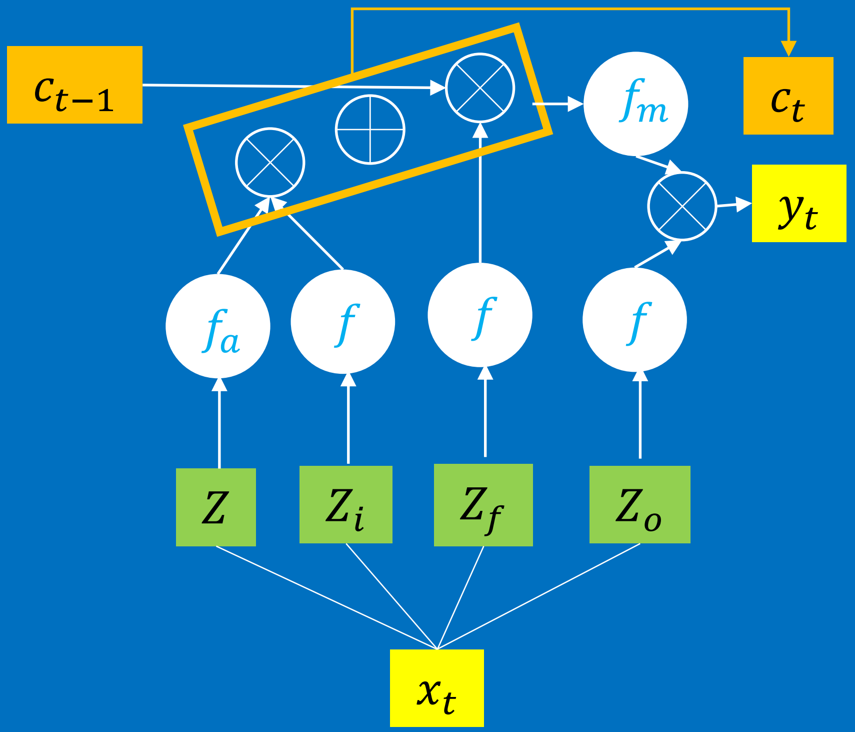
(10) Apply the activation function f to the output gate

(11) the output from (9) and (10) are multiplied together as the "output"

How LSTM works

So for one cell of LSTM, the workflow can be represented as





(1) the input is multiplied by weights to form the inputs for different gates and the input itself

(2) the input multiplied by a activation function f_a

(3) the input gate is multiplied by a activation function f

(4) the output from (2) and (3) are multiplied together

(5) the forget gate is multiplied by a activation function f

(6) Take the memory value from last time step c_{t-1}

(7) the output from (5) and (6) are multiplied together

(8) the output from (4) and (7) are added together

(9) Apply the activation function f_m to the output of (8)

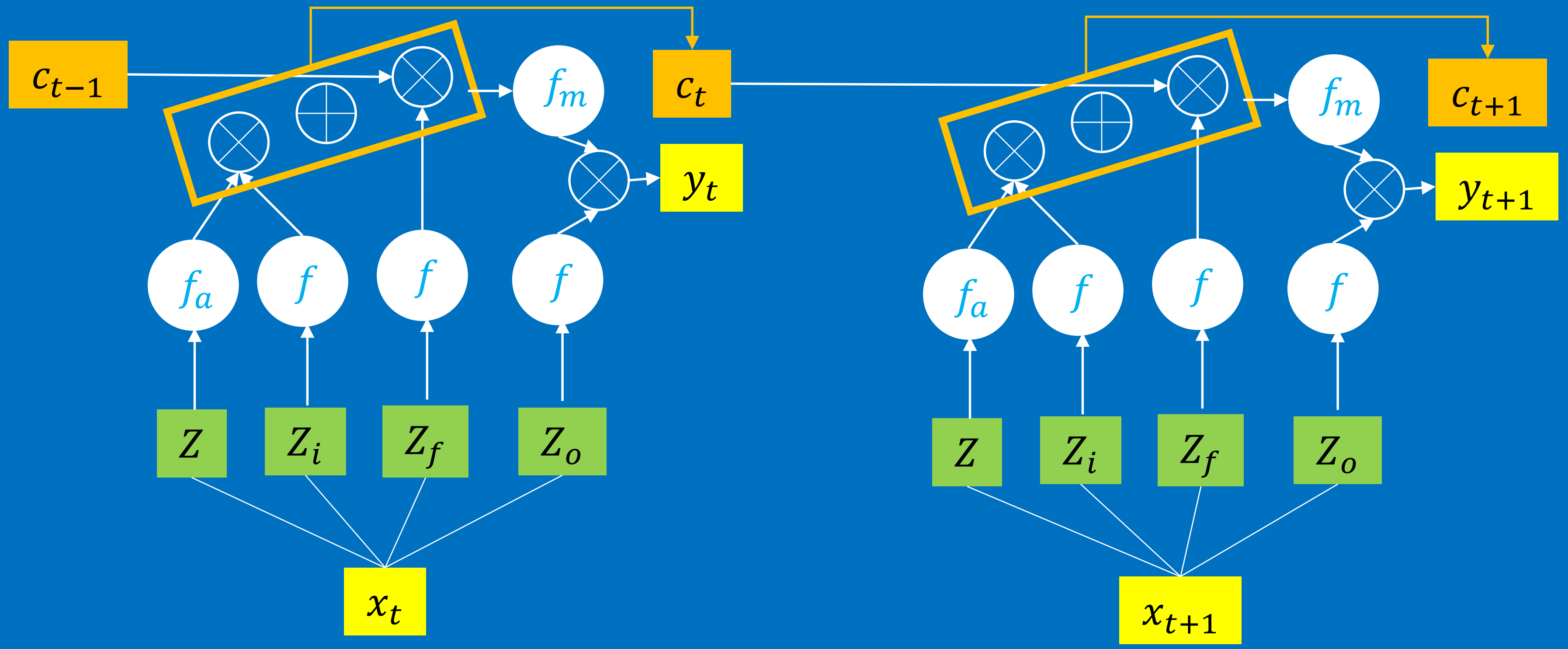
(10) Apply the activation function f to the output gate

(11) the output from (9) and (10) are multiplied together as the “output”

(12) store the updated memory value for next step

How LSTM works

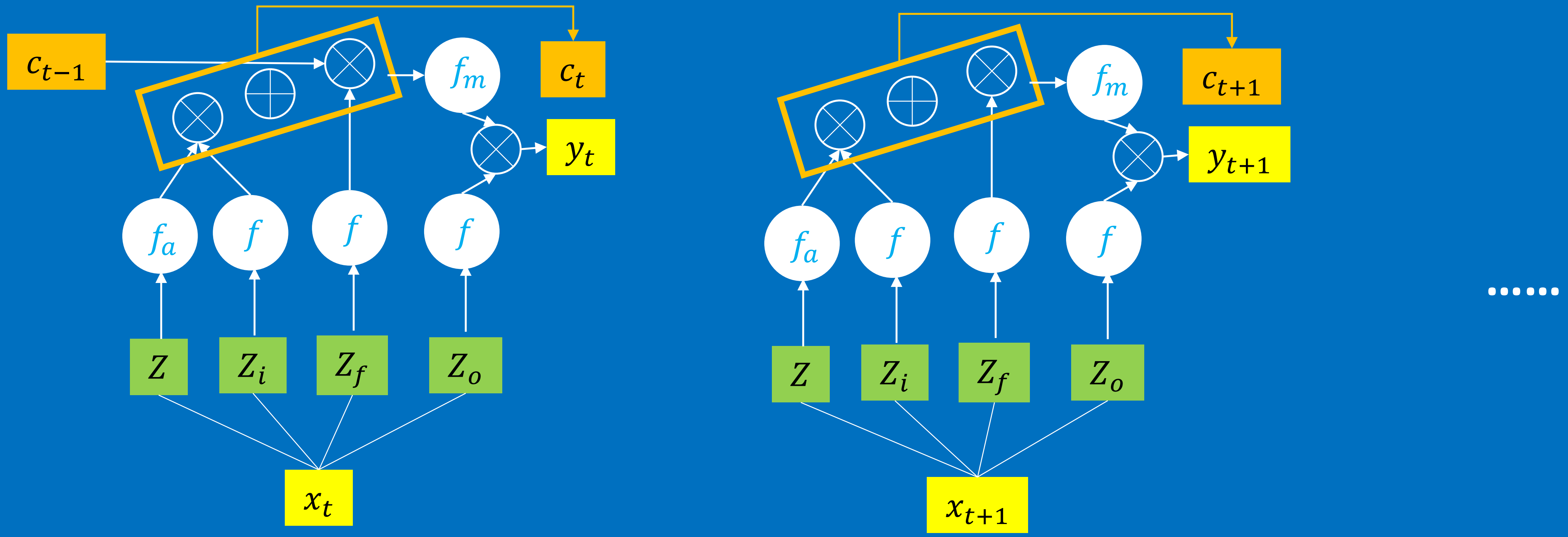
So for the subsequent time step, we have



......

How LSTM works

And for a LSTM with multiple neurons



How LSTM works

And for a LSTM with multiple neurons