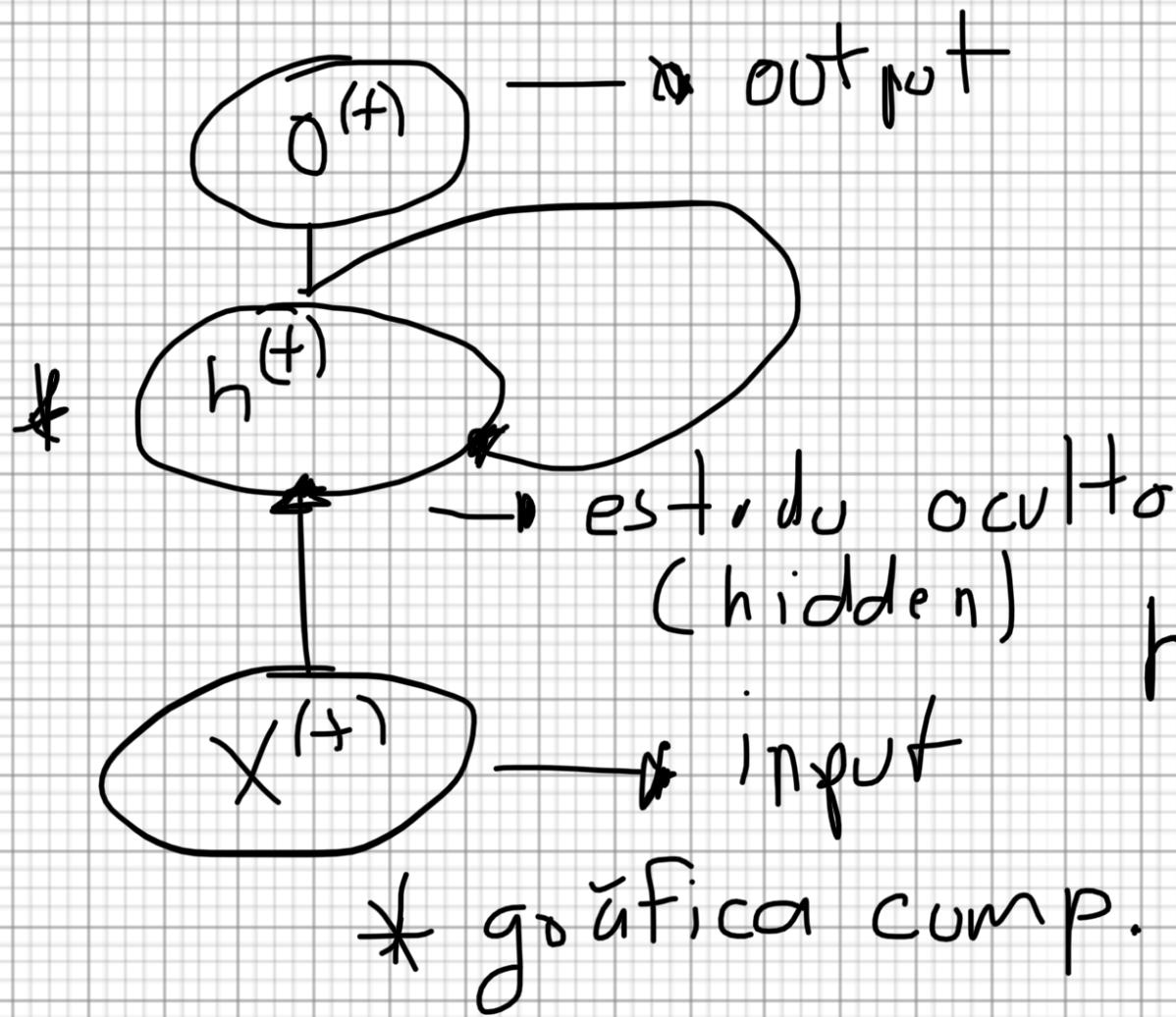


# Rredes recurrentes [RNN]



En alguna capa de la unidad.

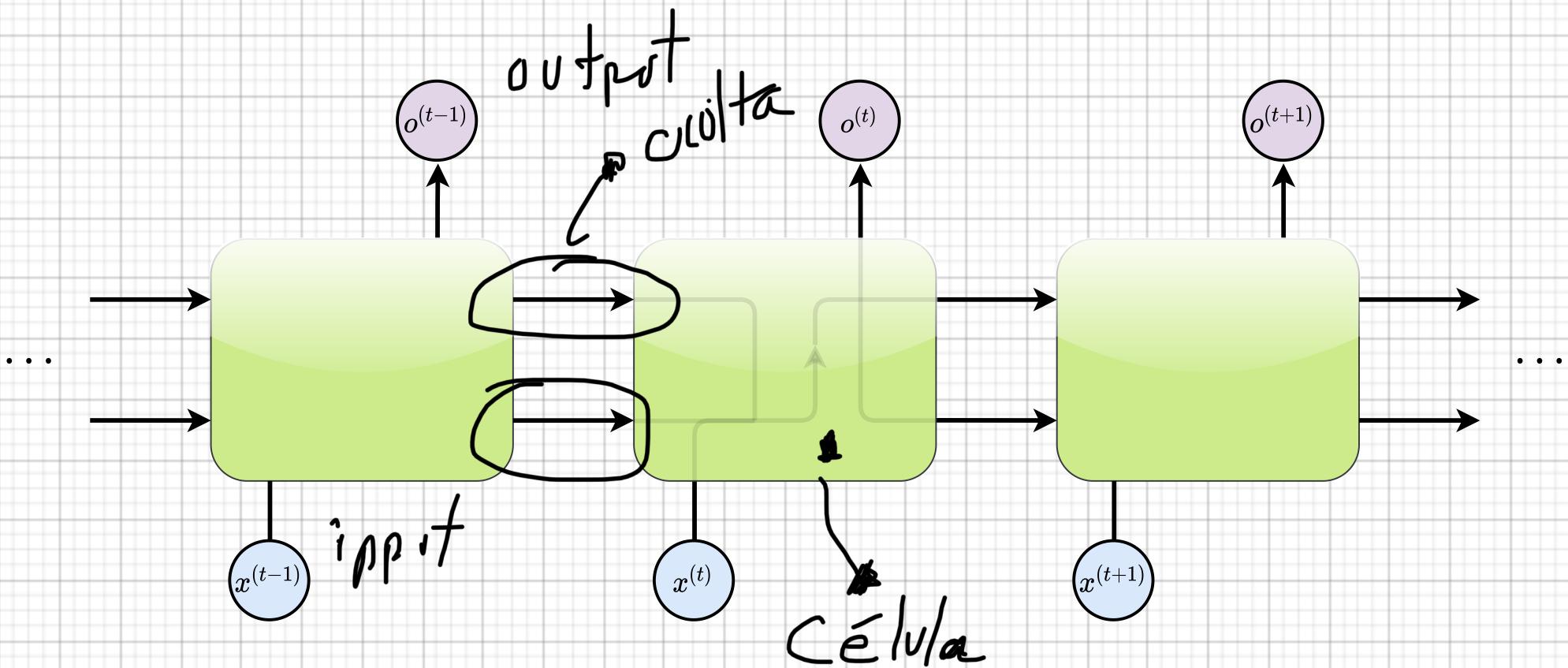
$$a^{(t)} = W_x^{(t)} + Vh^{(t-1)} + b$$

$$h^{(t)} = \text{Relu}(a^{(t)})$$

$h^{(t)}$  va a ser el resultado de multiplicar muchas veces una matriz.

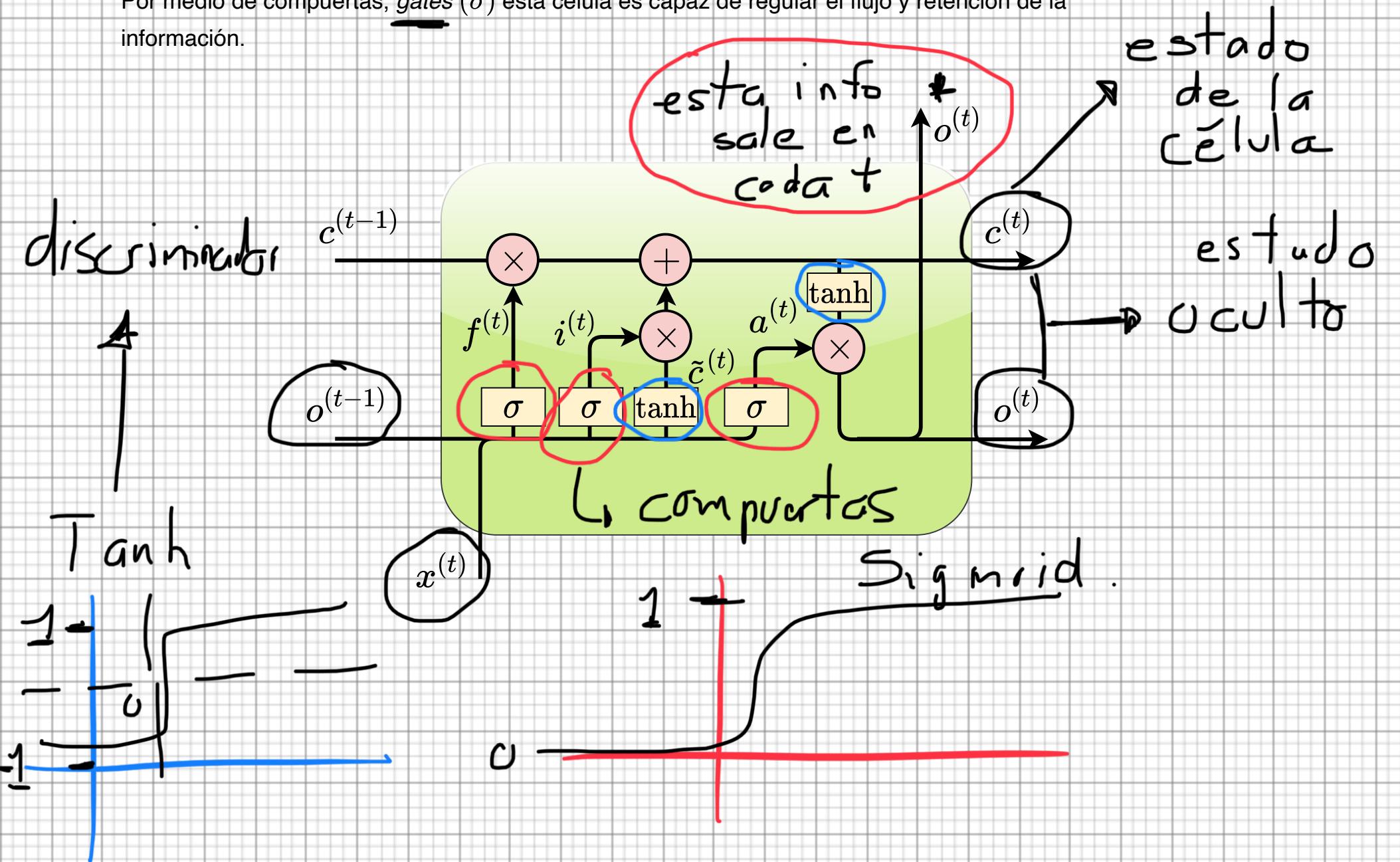
# LSTM

*Long Short Term Memory networks* son un tipo especial de RNNs capaces de aprender a preservar información a largo plazo, precisamente estan diseñadas para lidear con ese problema de retención por un largo período de tiempo.



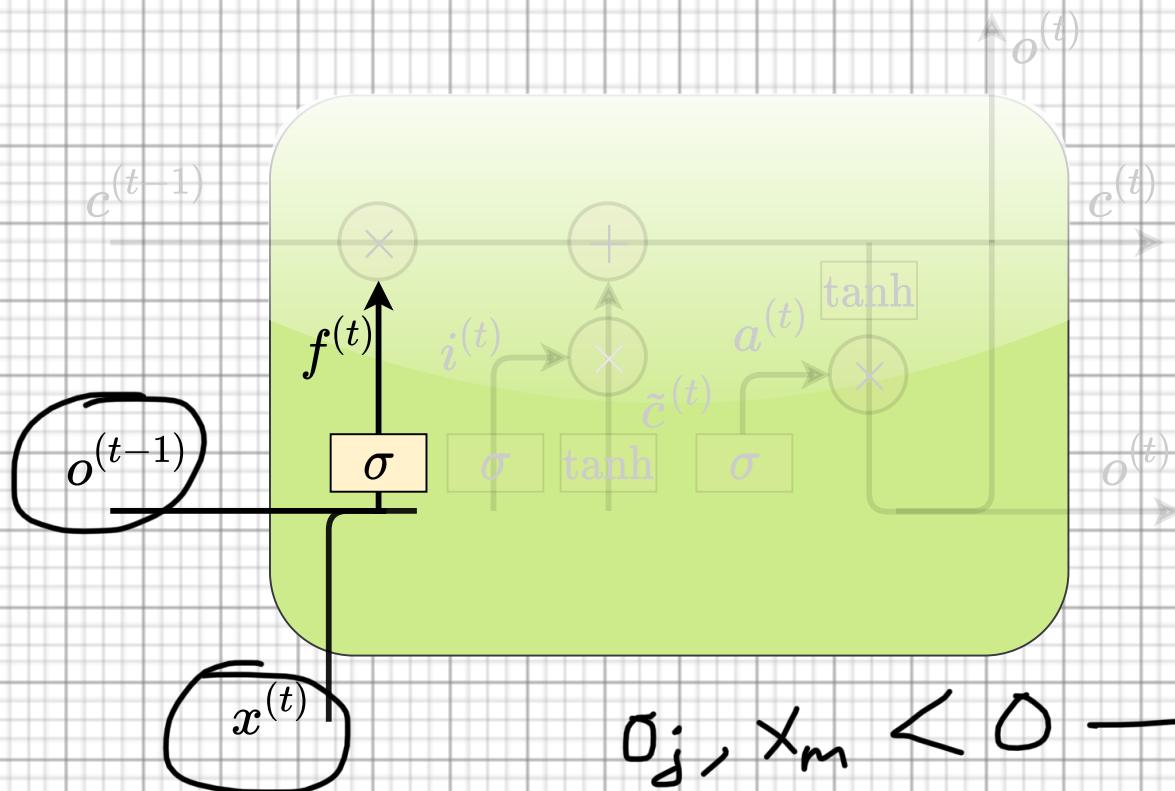
# Célula LSTM

Por medio de compuertas, *gates* ( $\sigma$ ) esta célula es capaz de regular el flujo y retención de la información.



# Paso 1

Aquí decide que tanta información se conserva del estado de la unidad anterior ( $t - 1$ ), así como la información que viene del *input* al tiempo  $t$ .



$$o_j, x_m < 0 \rightarrow$$

No se van a retener tanto

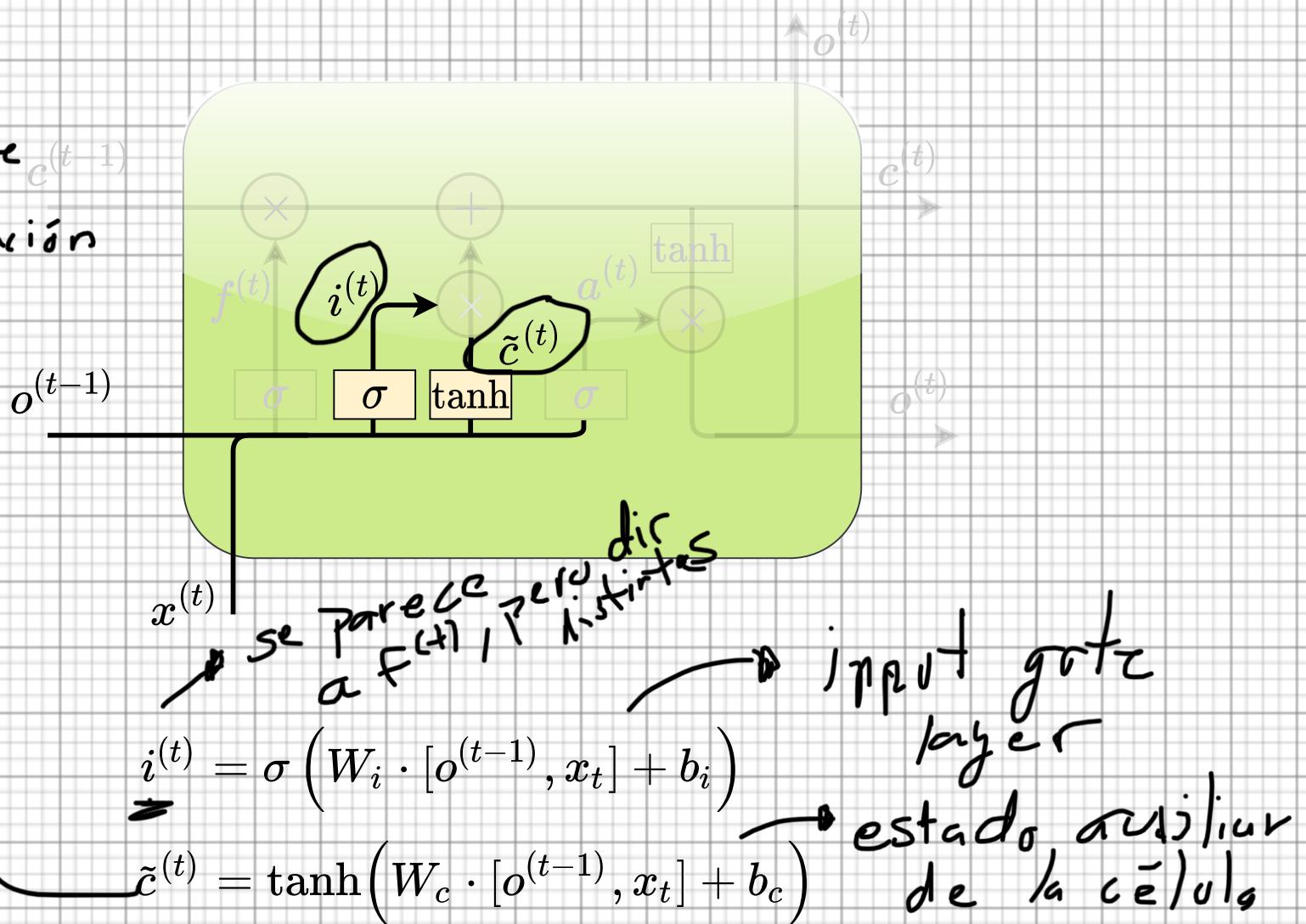
$$f^{(t)} = \underline{\sigma} \left( \underline{W_f} \cdot [o^{(t-1)}, x^{(t)}] + \underline{b_f} \right)$$

Ajustar  $W_f$

# Paso 2

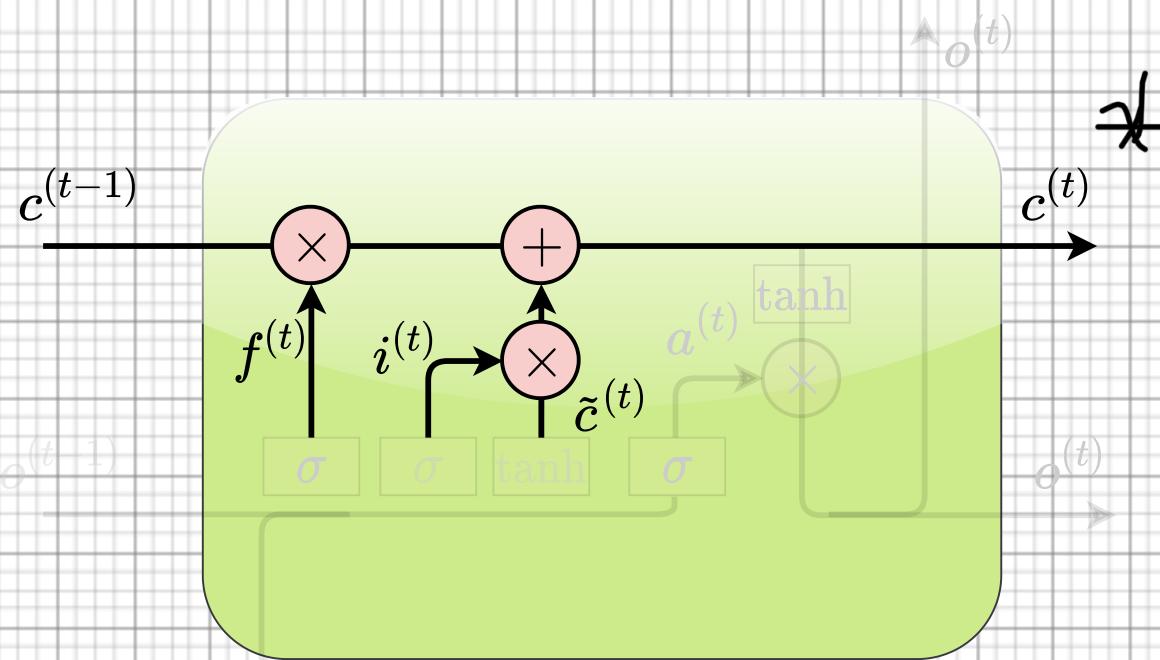
Aquí la primera compuerta, *input gate layer*, decide que información se va actualizar. La segunda compuerta selecciona valores que son candidatos a estar en el estado de la célula.

Discriminador  
↓ influye  
la <sup>en</sup> retención  
Compuerda



# Paso 3

Aquí se actualiza el estado de la célula para el tiempo  $t$ . Se realiza una combinación entre el estado anterior ( $t - 1$ ) y el estado auxiliar de la célula al tiempo  $t$ .



$$e \in [0, 1]$$

$$f^{(t)} \cdot c^{(t-1)} + i^{(t)} \cdot \tilde{c}^{(t)}$$

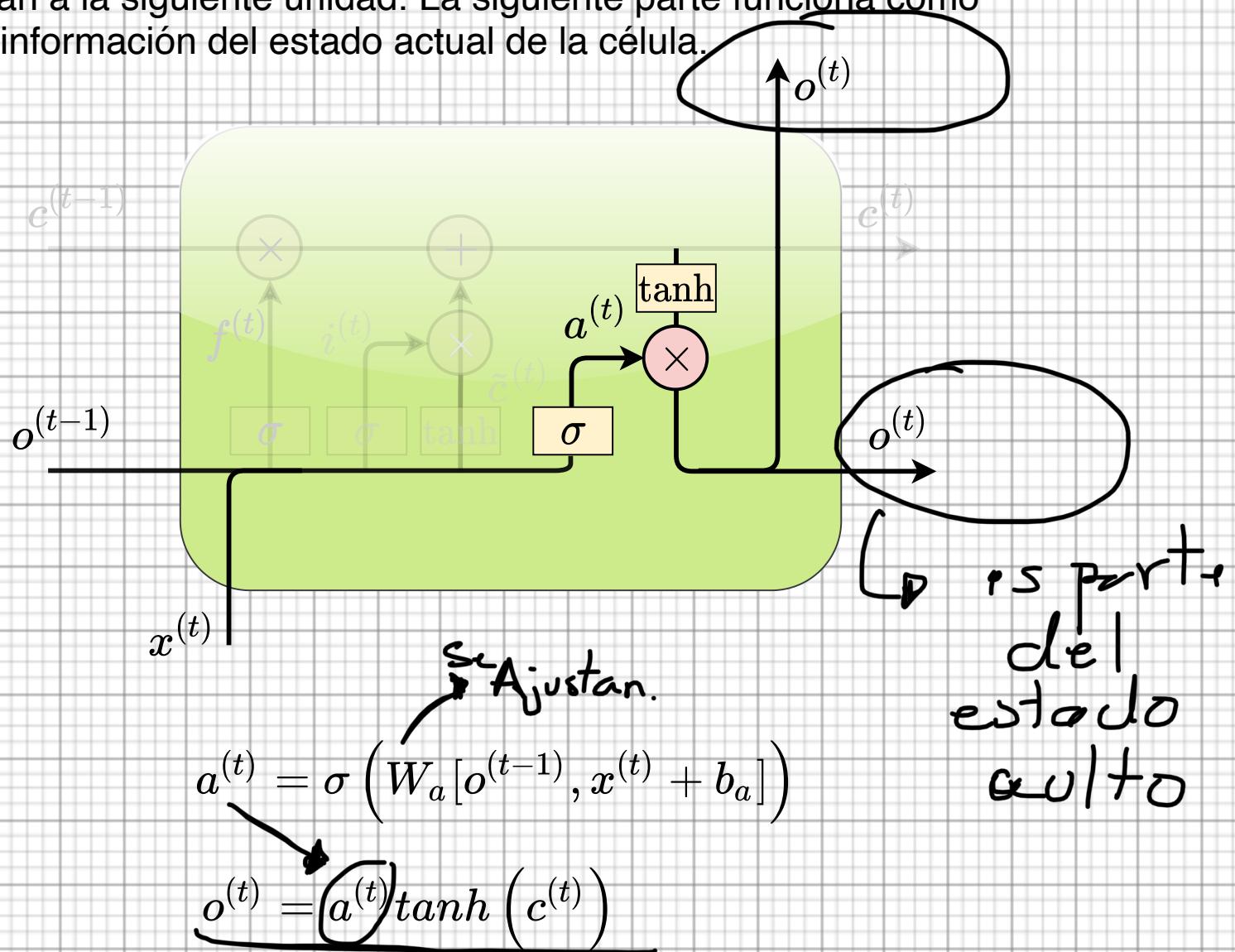
↓ Gén anterior ↳ Paso 2

Paso 1

\* Pasa a la generación siguiente

# Paso 4

Finalmente se decide el *output* de la célula, que será una versión filtrada del estado de la célula al tiempo  $t$ . La primera compuerta decide que valores de la información actual y el estado anterior pasan a la siguiente unidad. La siguiente parte funciona como discriminador de la información del estado actual de la célula.



# En resumen ...

$$f^{(t)} = \sigma \left( W_f \cdot [o^{(t-1)}, x^{(t)}] + b_f \right)$$

$$i^{(t)} = \sigma \left( W_i \cdot [o^{(t-1)}, x_t] + b_i \right)$$

$$c^{(t)} = f^{(t)} \cdot c^{(t-1)} + i^{(t)} \cdot \tilde{c}^{(t)}$$

$$\tilde{c}^{(t)} = \tanh \left( W_c \cdot [o^{(t-1)}, x_t] + b_c \right)$$

$$a^{(t)} = \sigma \left( W_a [o^{(t-1)}, x^{(t)}] + b_a \right)$$

$$o^{(t)} = a^{(t)} \tanh(c^{(t)})$$

LSTM representa una mejora con respecto a las arquitecturas *vanilla RNN*, pues lida con el problema de retención sin que explote o desvanezca el gradiente.

