Manuel Mager

Redes Neuronales para datos secuenciales

July 2, 2019

Institute for Natural Language Processing
University of Stuttgart

Contenido

Introducción

Introducción al procesamiento secuencial

Datos no secuenciales

Las primeras redes neuronales lograron clasificar datos no secuenciales.

Ejemplo

Clasificar flores basándonos en 4 características: largo del pétalo, ancho del pétalo, largo del sépalo, ancho del sépalo. Se clasifica las flores en 3 especies.

Datos no secuenciales

Las primeras redes neuronales lograron clasificar datos no secuenciales.

Ejemplo

Clasificar flores basándonos en 4 características: largo del pétalo, ancho del pétalo, largo del sépalo, ancho del sépalo. Se clasifica las flores en 3 especies.

Datos secuenciales

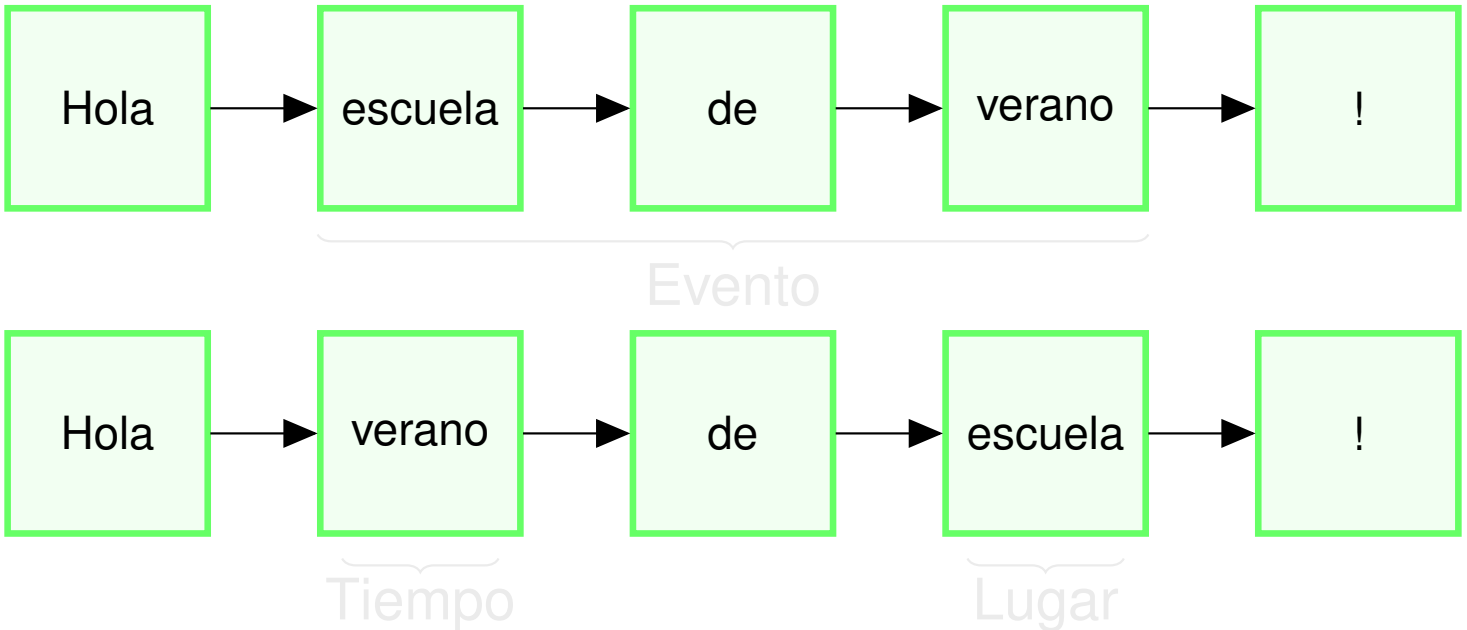
Ciertos datos, están se relacionan entre sí sobre una serie de tiempo.

- Desarrollo de la economía de un país
- La temperatura
- Lenguaje Humano
- La organización molecular de las proteínas

Es una secuencia que contiene vectores x_t que ocurre en un tiempo t que occure de $1..T$.

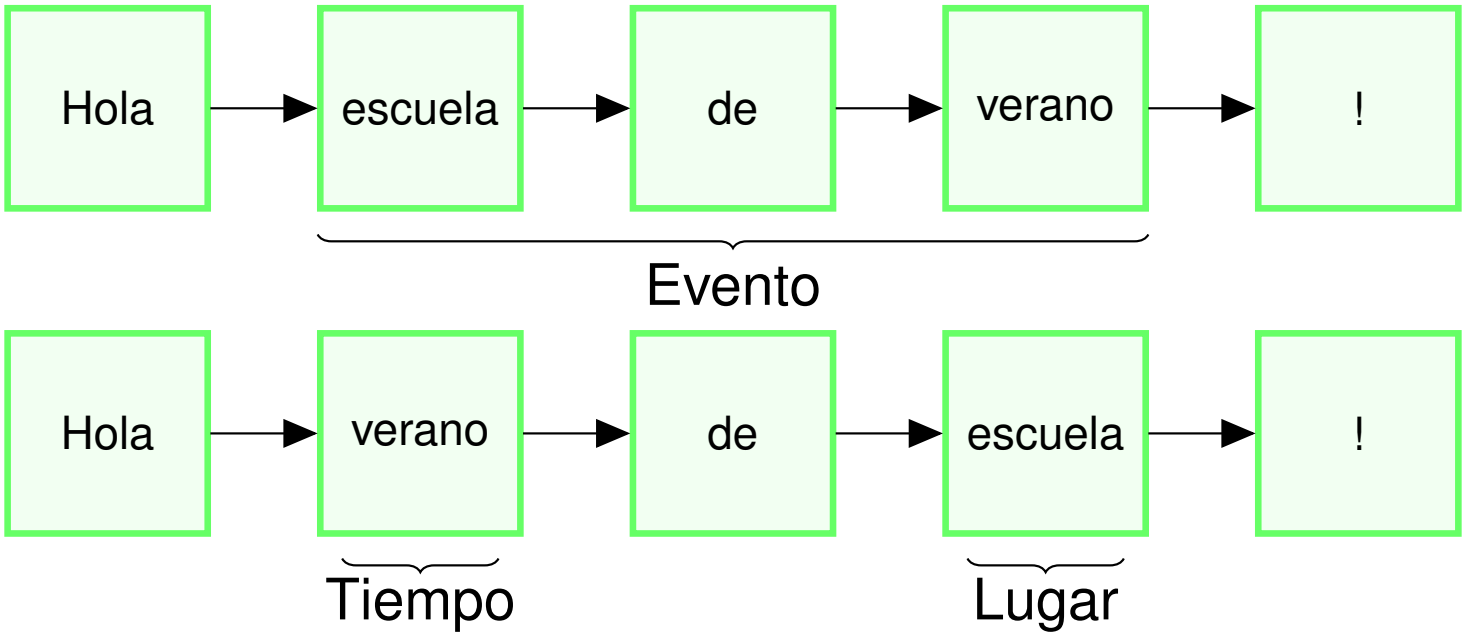
Ejemplo con lenguaje

Tratemos de encontrar la entidad de nombre en la siguiente frase.



Ejemplo con lenguaje

Tratemos de encontrar la entidad de nombre en la siguiente frase.



Redes Neuronales Recursivas (RNN)

Es una clase especial de redes neuronales artificiales que tiene conexiónes entre nodos de un grafo dirigido a lo largo de una secuencia temporal.

Desplegar un grafo computacional

Un RNN puede ser entendido como un grafo computacional con estructuras repetitivas, que corresponde a una cadena de eventos. El sistema dinámico

$$h_t = f(h_{t-1})$$

puede ser expanido de la siguiente forma para $t + 1$.

$$\begin{aligned} h_{t+1} &= f(h_t) \\ &= f(f(h_{t-1})) \end{aligned}$$

Gráficamente podemos visualizarlo de la siguiente manera:

Desplegar un grafo computacional

Un RNN puede ser entendido como un grafo computacional con estructuras repetitivas, que corresponde a una cadena de eventos. El sistema dinámico

$$h_t = f(h_{t-1})$$

puede ser expanido de la siguiente forma para $t + 1$.

$$\begin{aligned} h_{t+1} &= f(h_t) \\ &= f(f(h_{t-1})) \end{aligned}$$

Gráficamente podemos visualizarlo de la siguiente manera:

Desplegar un grafo computacional

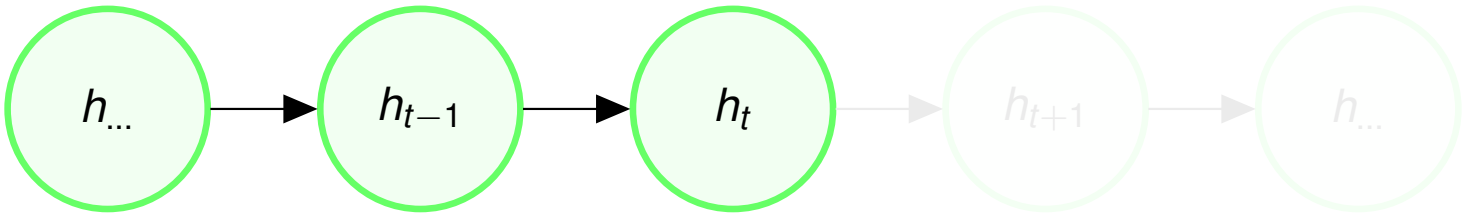
Un RNN puede ser entendido como un grafo computacional con estructuras repetitivas, que corresponde a una cadena de eventos. El sistema dinámico

$$h_t = f(h_{t-1})$$

puede ser expanido de la siguiente forma para $t + 1$.

$$\begin{aligned} h_{t+1} &= f(h_t) \\ &= f(f(h_{t-1})) \end{aligned}$$

Gráficamente podemos visualizarlo de la siguiente manera:



Desplegar un grafo computacional

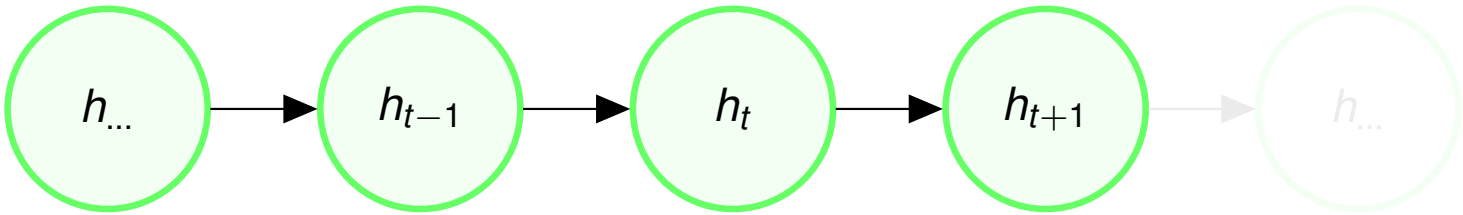
Un RNN puede ser entendido como un grafo computacional con estructuras repetitivas, que corresponde a una cadena de eventos. El sistema dinámico

$$h_t = f(h_{t-1})$$

puede ser expanido de la siguiente forma para $t + 1$.

$$\begin{aligned} h_{t+1} &= f(h_t) \\ &= f(f(h_{t-1})) \end{aligned}$$

Gráficamente podemos visualizarlo de la siguiente manera:



Desplegar un grafo computacional

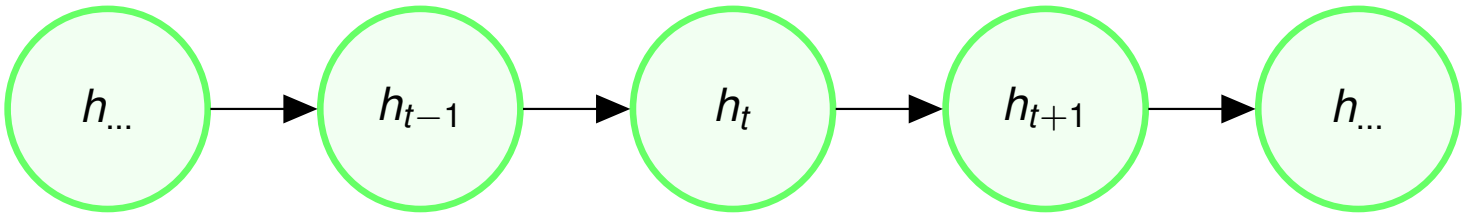
Un RNN puede ser entendido como un grafo computacional con estructuras repetitivas, que corresponde a una cadena de eventos. El sistema dinámico

$$h_t = f(h_{t-1})$$

puede ser expanido de la siguiente forma para $t + 1$.

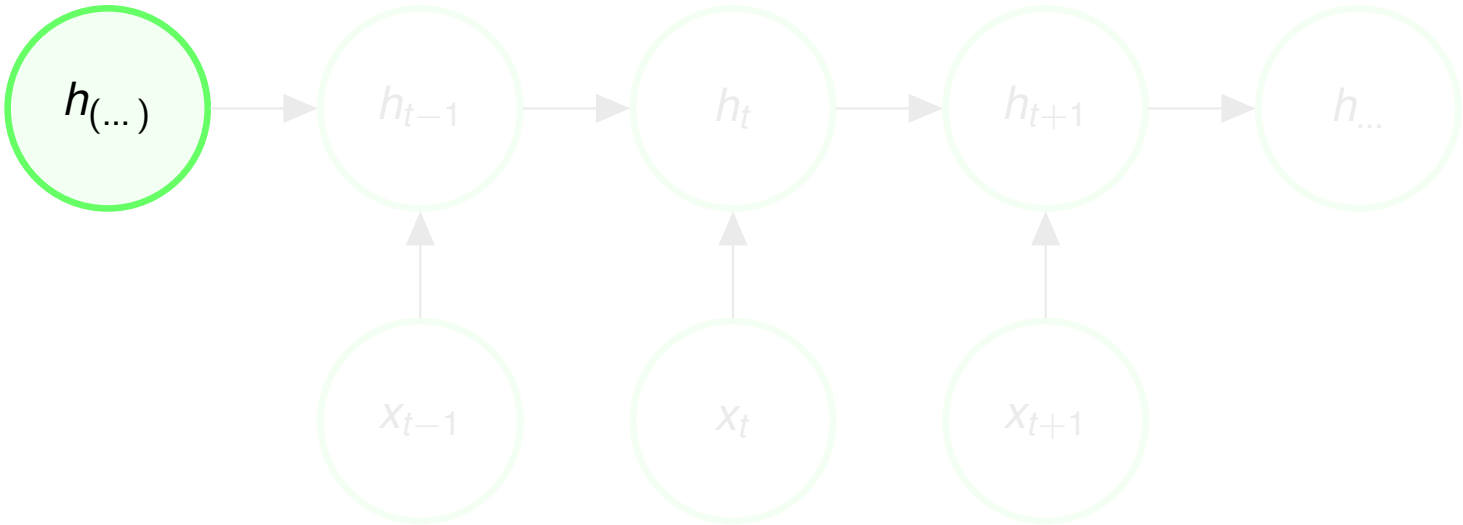
$$\begin{aligned} h_{t+1} &= f(h_t) \\ &= f(f(h_{t-1})) \end{aligned}$$

Gráficamente podemos visualizarlo de la siguiente manera:



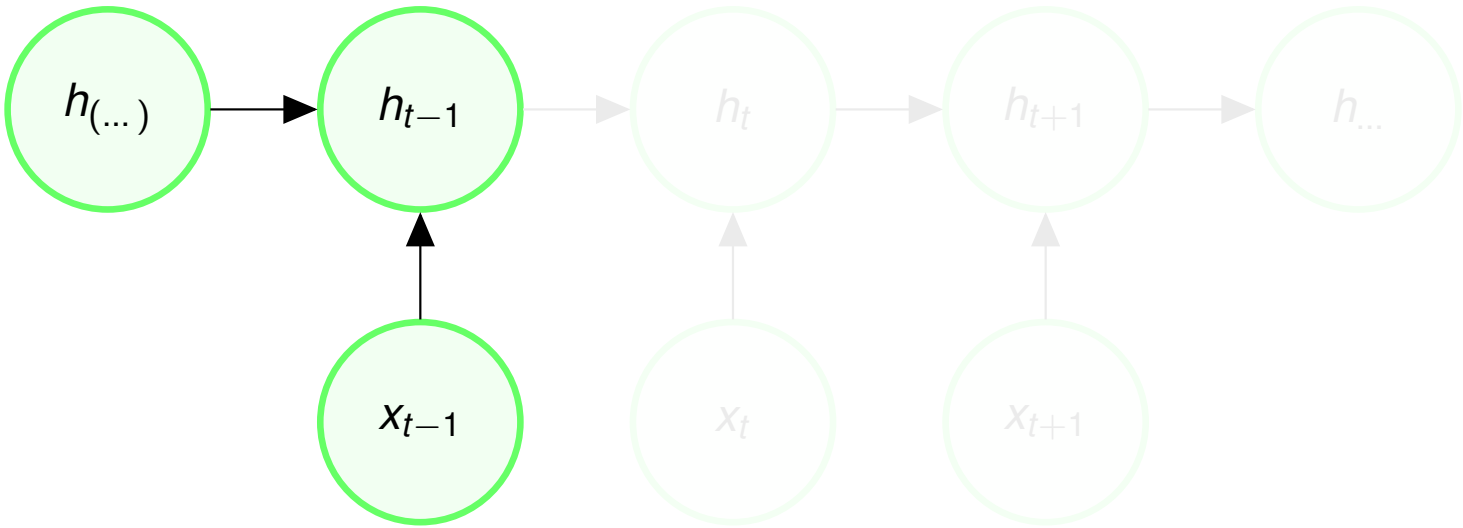
Grafo computacional con una señal externa x en cada tiempo t.

$$h(t) = f(h_{t-1}, x_t)$$



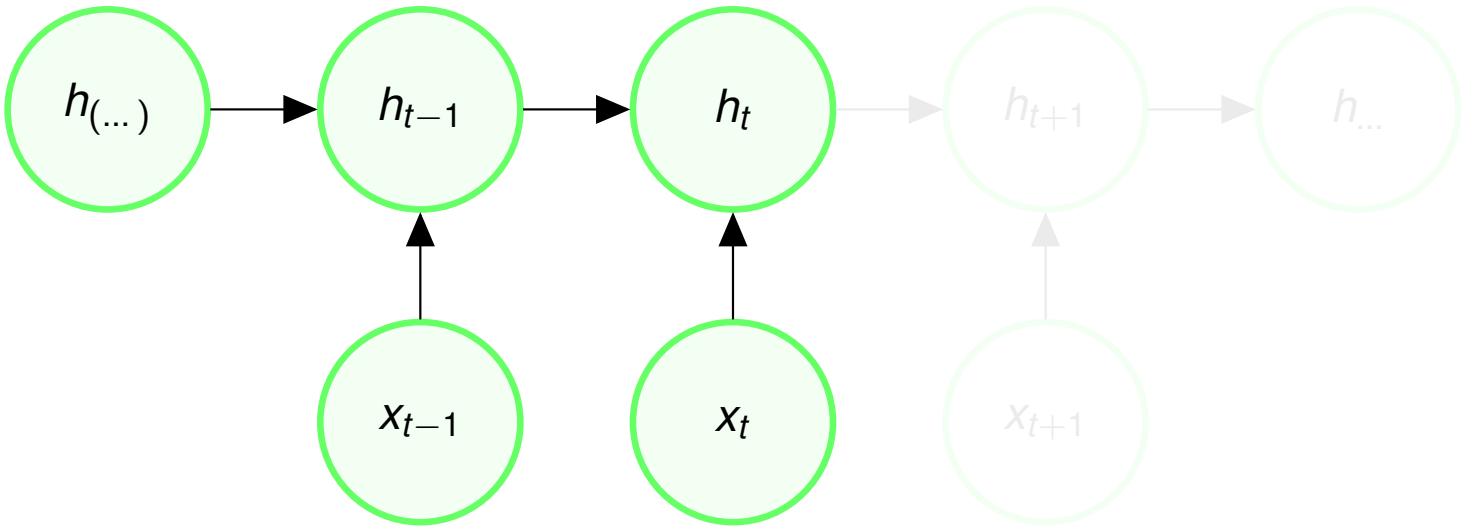
Grafo computacional con una señal externa x en cada tiempo t.

$$h(t) = f(h_{t-1}, x_t)$$



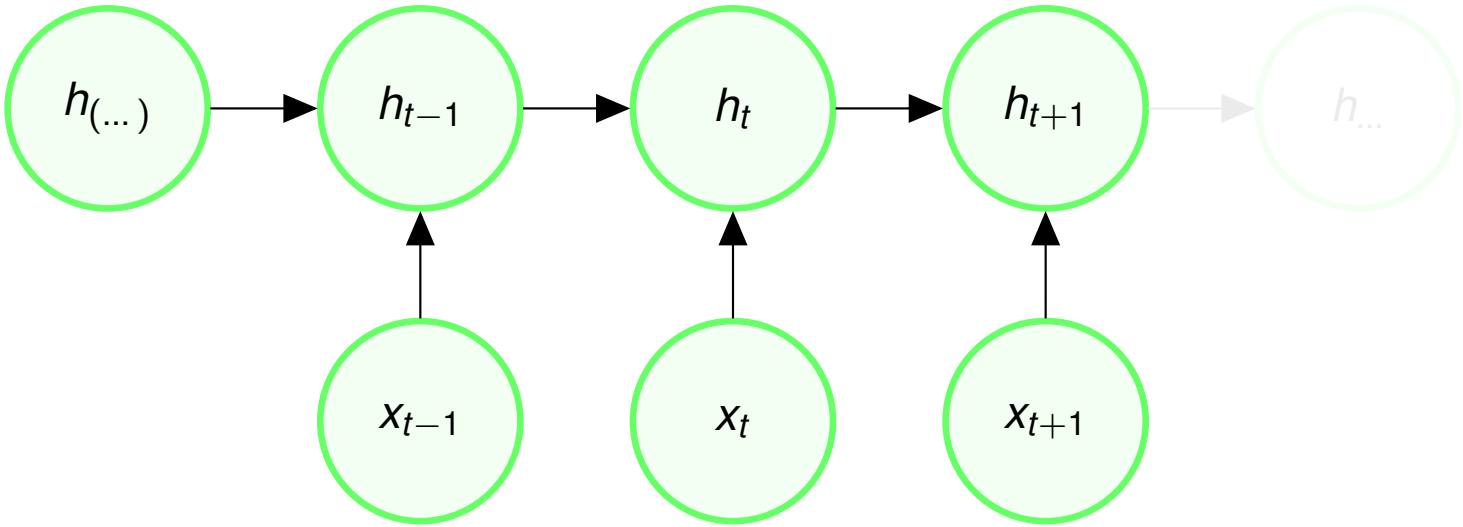
Grafo computacional con una señal externa x en cada tiempo t.

$$h(t) = f(h_{t-1}, x_t)$$



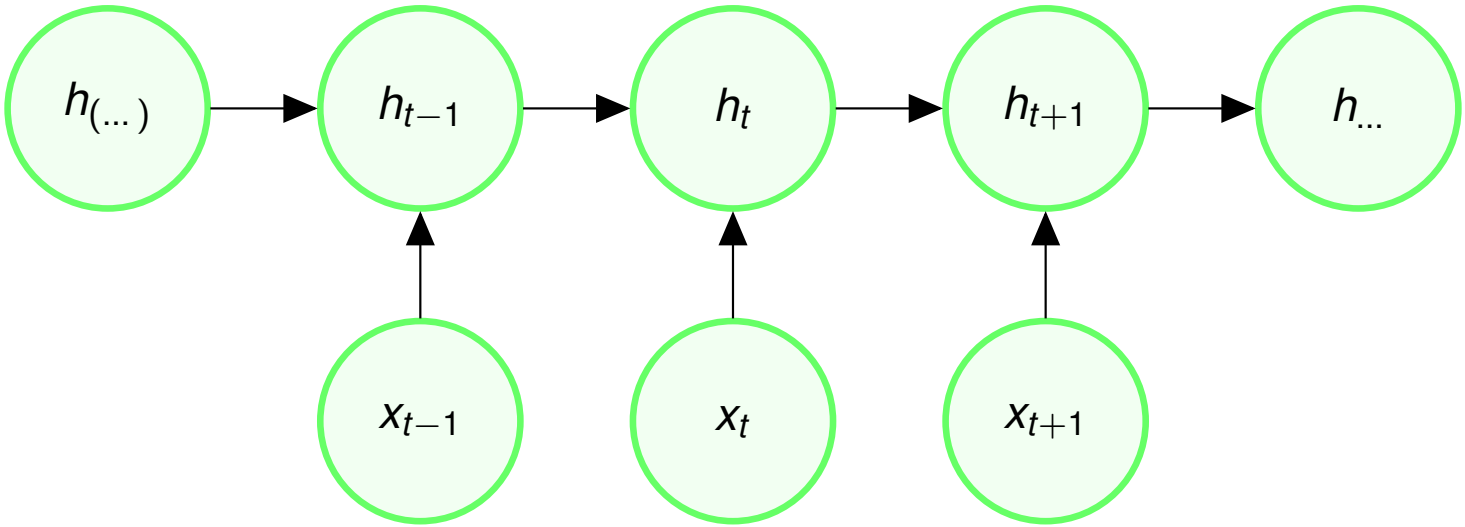
Grafo computacional con una señal externa x en cada tiempo t.

$$h(t) = f(h_{t-1}, x_t)$$



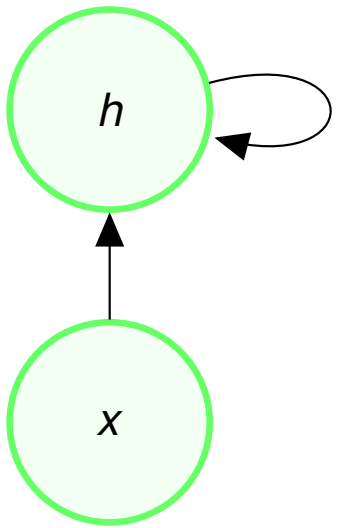
Grafo computacional con una señal externa x en cada tiempo t.

$$h(t) = f(h_{t-1}, x_t)$$

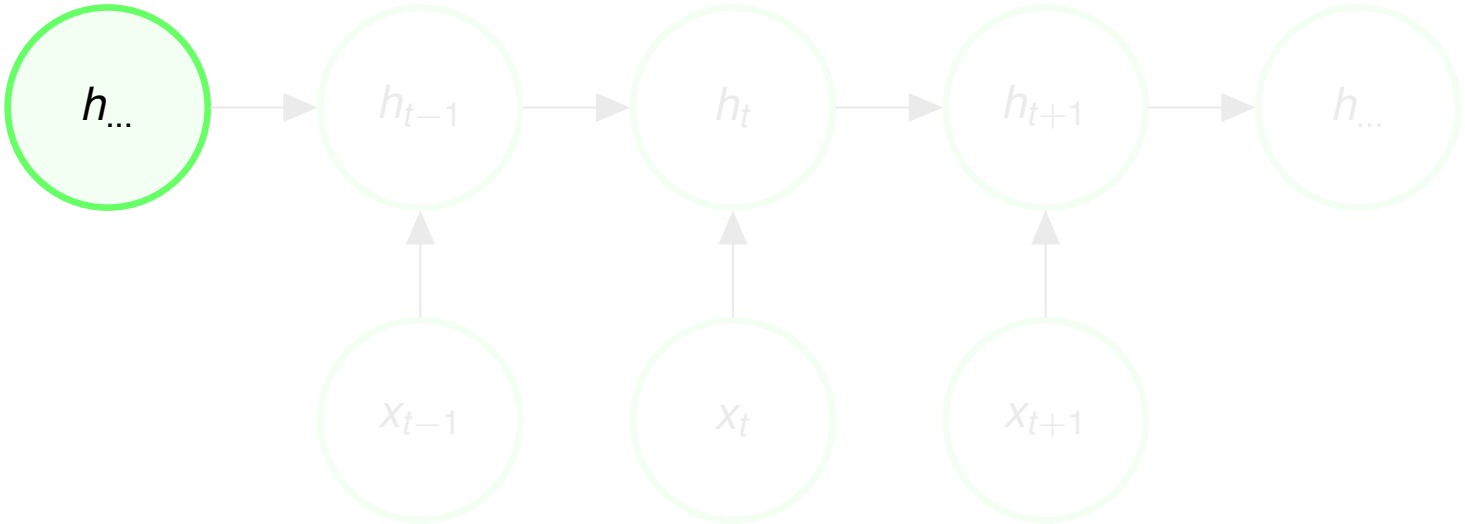


Grafo computacional con una senal externa x en cada tiempo t.

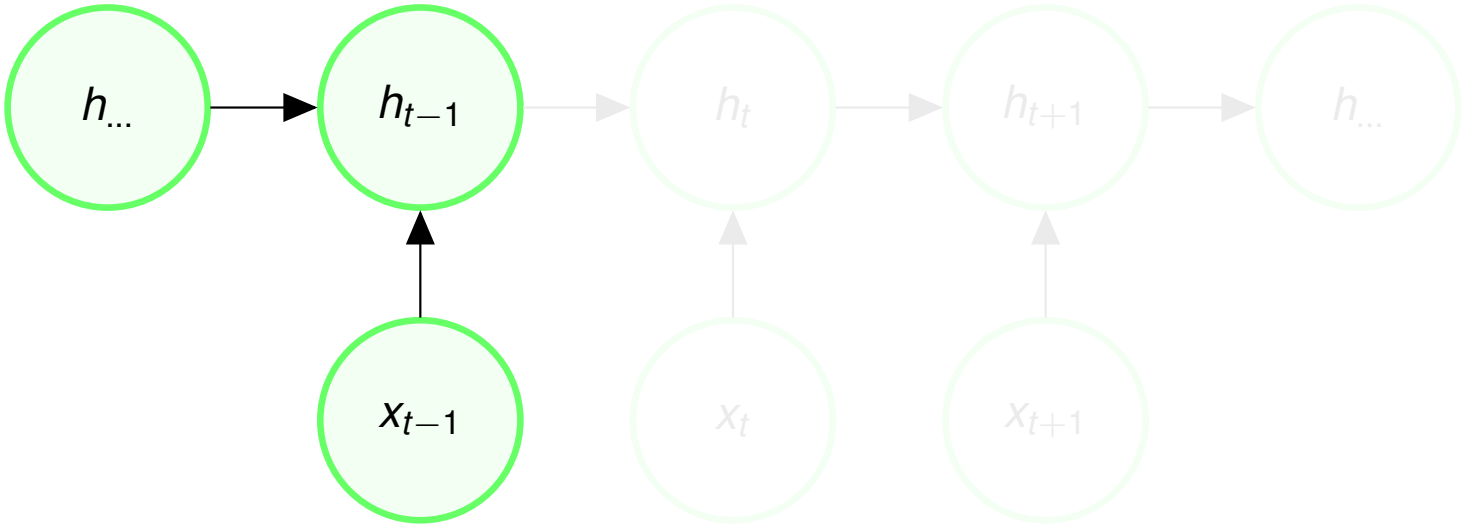
$$h(t) = f(h_{t-1}, x_t)$$



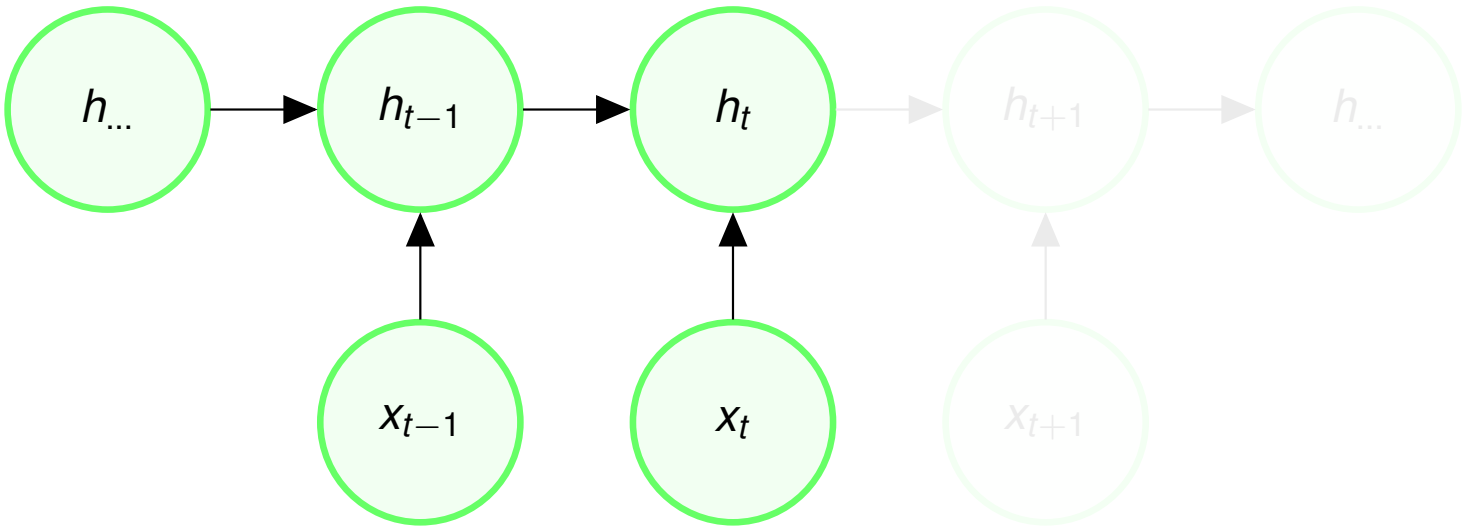
Desplegar un RNN universal



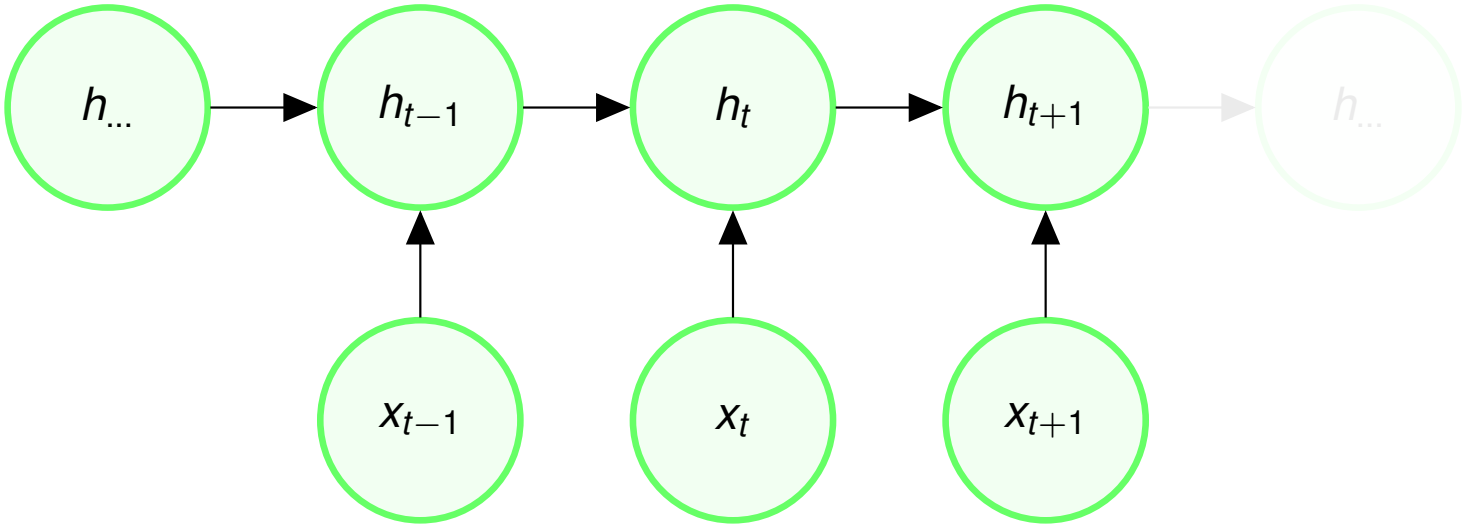
Desplegar un RNN universal



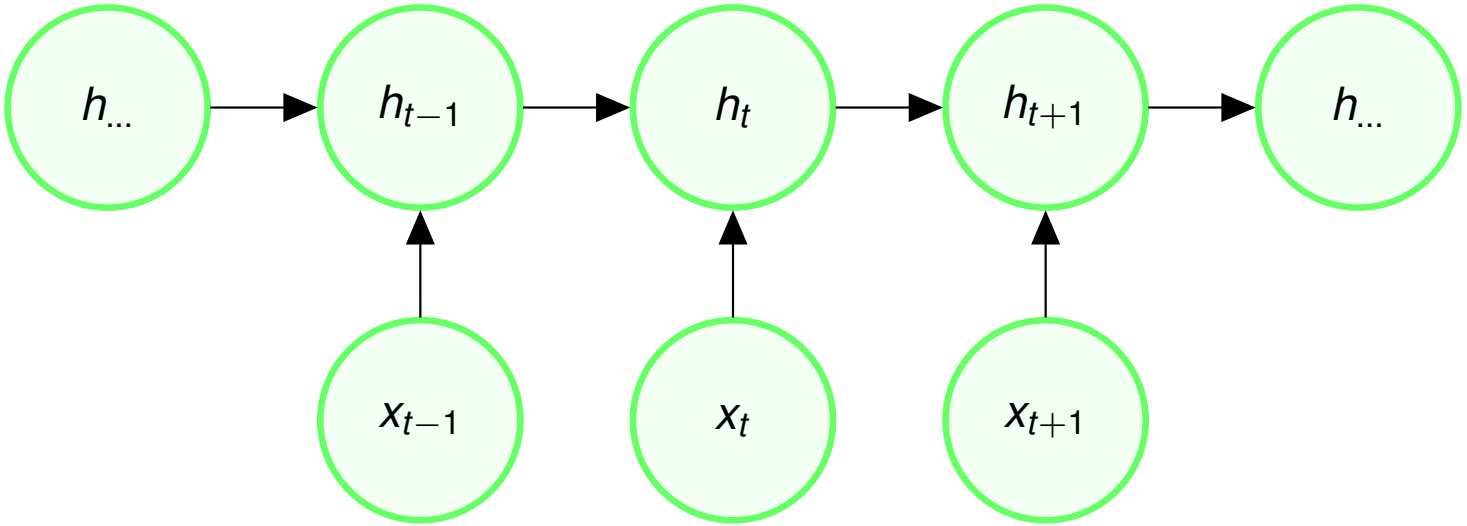
Desplegar un RNN universal



Desplegar un RNN universal



Desplegar un RNN universal

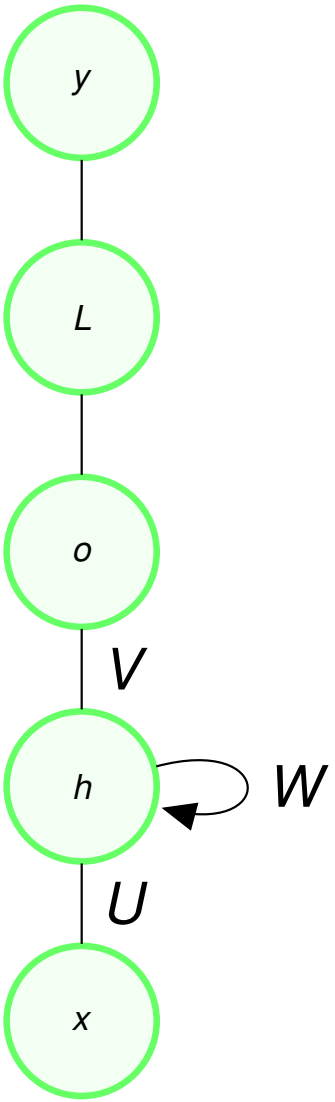


$$\mathbf{a}_t = \mathbf{b} + \mathbf{W}\mathbf{h}_{t-1} + Ux_t$$

$$\mathbf{h}_t = \tanh(\mathbf{a}_t)$$

$$\mathbf{o}_t = \mathbf{c} + \mathbf{V}\mathbf{h}_t$$

$$\mathbf{h}_t = \text{softmax}(\mathbf{o}_t)$$

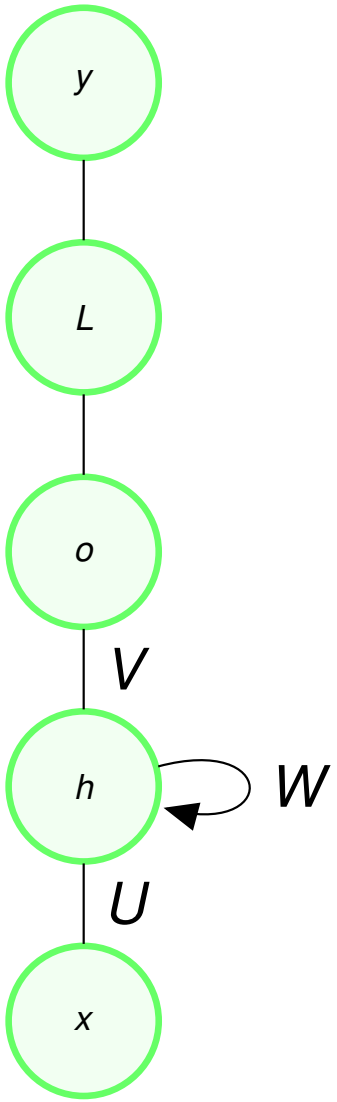


$$\mathbf{a}_t = \mathbf{b} + \mathbf{W}\mathbf{h}_{t-1} + Ux_t$$

$$\mathbf{h}_t = \tanh(\mathbf{a}_t)$$

$$\mathbf{o}_t = \mathbf{c} + \mathbf{V}\mathbf{h}_t$$

$$\mathbf{h}_t = \text{softmax}(\mathbf{o}_t)$$

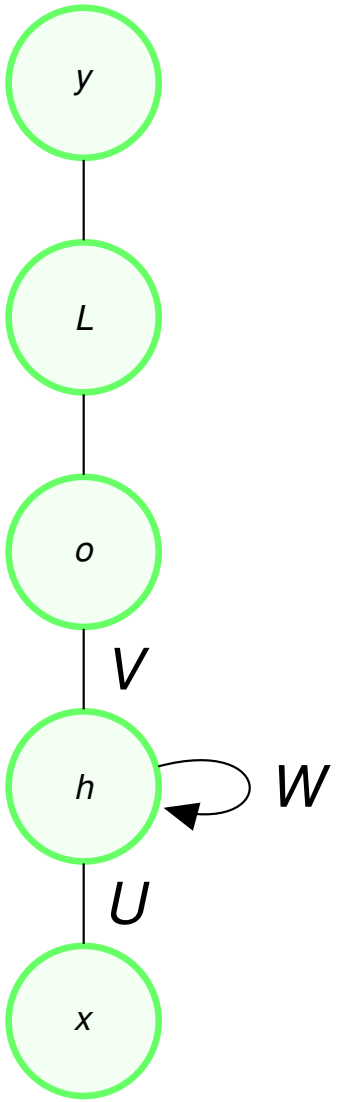


$$\mathbf{a}_t = \mathbf{b} + \mathbf{W}\mathbf{h}_{t-1} + Ux_t$$

$$\mathbf{h}_t = \tanh(\mathbf{a}_t)$$

$$\mathbf{o}_t = \mathbf{c} + \mathbf{V}\mathbf{h}_t$$

$$\mathbf{h}_t = \text{softmax}(\mathbf{o}_t)$$

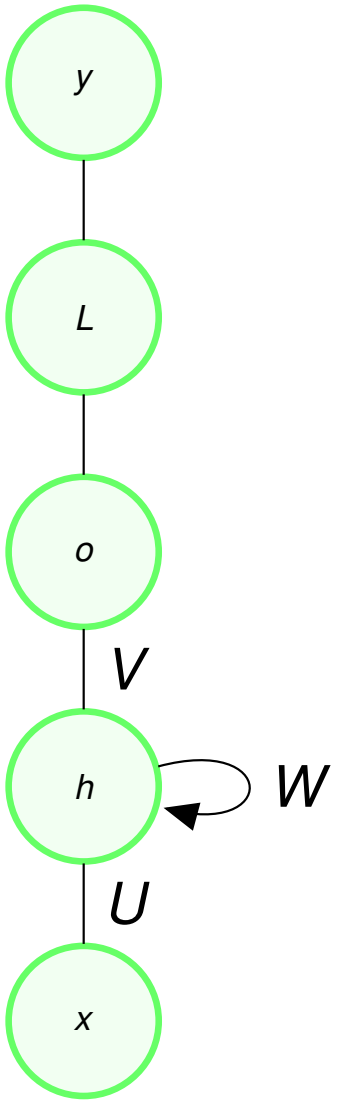


$$\mathbf{a}_t = \mathbf{b} + \mathbf{W}\mathbf{h}_{t-1} + Ux_t$$

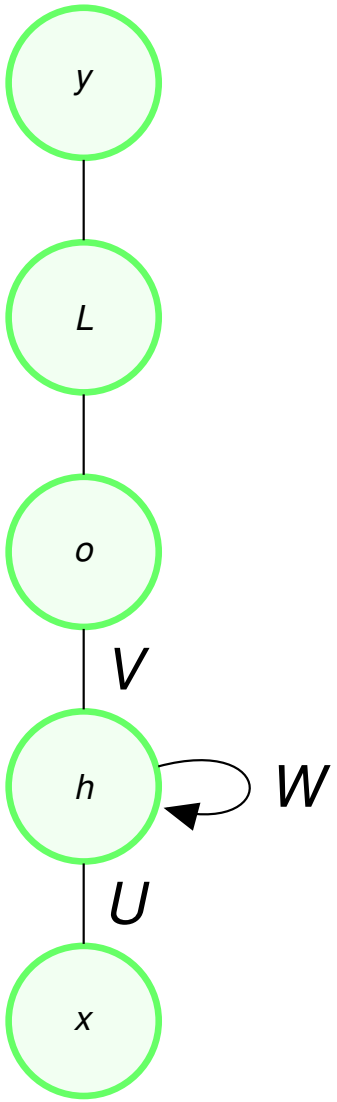
$$\mathbf{h}_t = \tanh(\mathbf{a}_t)$$

$$\mathbf{o}_t = \mathbf{c} + \mathbf{V}\mathbf{h}_t$$

$$\mathbf{h}_t = \text{softmax}(\mathbf{o}_t)$$



$$\mathbf{a}_t = \mathbf{b} + \mathbf{W}\mathbf{h}_{t-1} + Ux_t$$
$$\mathbf{h}_t = \tanh(\mathbf{a}_t)$$
$$\mathbf{o}_t = \mathbf{c} + \mathbf{V}\mathbf{h}_t$$
$$\mathbf{h}_t = \text{softmax}(\mathbf{o}_t)$$



Loss function

la función de pérdida es la suma de la función de pérdida sobre todos los momentos en el tiempo.

$$L(y, \hat{y}) = \sum_t L^{(t)}$$

Gradiente para RNNs

Para ilustrar el back propagation por el tiempo vamos a usar una RNN bastante simple.

$$s_t = \tanh(Ux_t + Ws_{t-1}) \tag{1}$$
$$\hat{y}_t = \text{softmax}(Vs_t) \tag{2}$$

Y definimos nuestra función de loss cómo un cross entropy.

$$E_t(y_t, \hat{y}_t) = -y_t \log \hat{y}_t \tag{3}$$
$$E(y, \hat{y}) = \sum_t E_t(y_t, \hat{h}_t) \tag{4}$$
$$= -\sum_t y_t \log \hat{y}_t \tag{5}$$

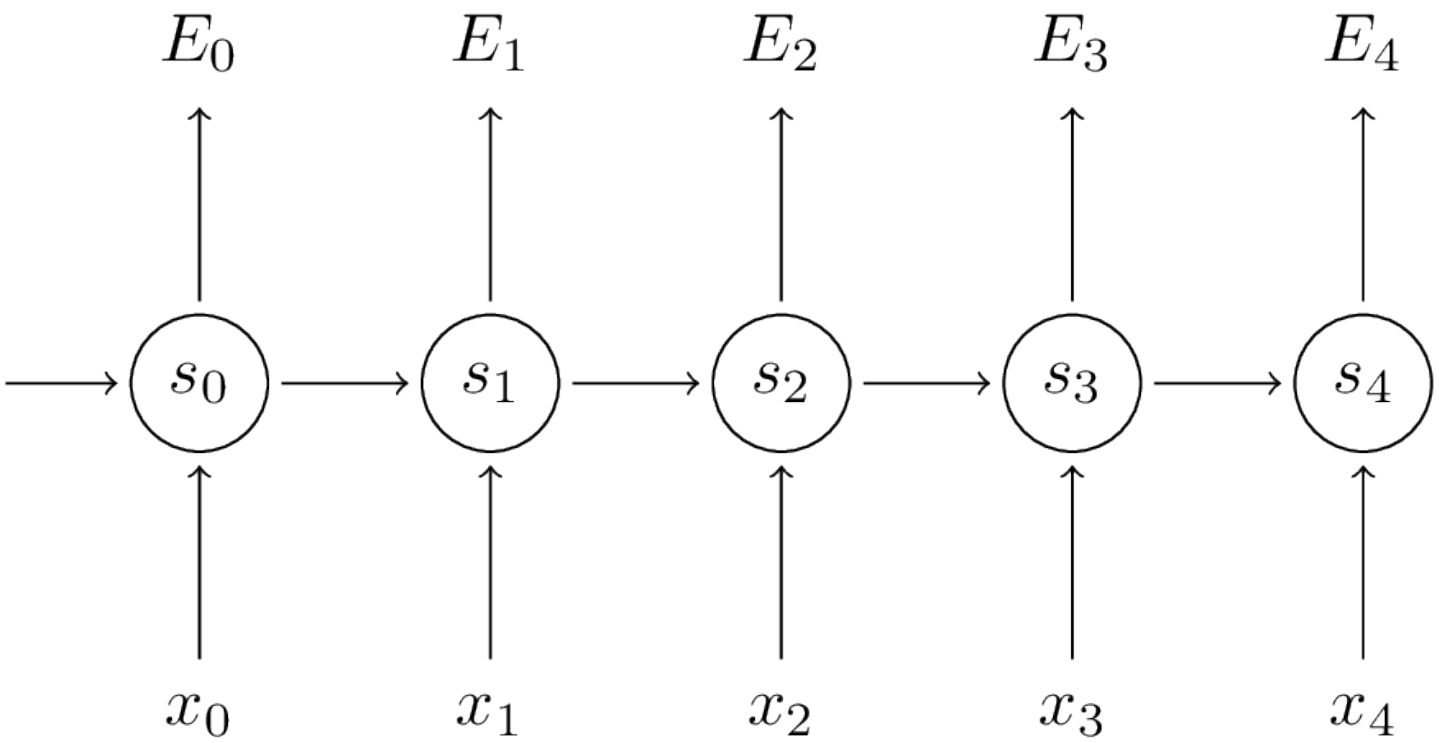


Figure: Fuente: `http://www.wildml.com`

Ahora queremos calcular las gradientes del error respecto a los parámetros U, V y W. Para ello sumamos las gradientes de cada paso en el tiempo para cada ejemplo de entrenamiento:

$$\frac{\partial E}{\partial W} = \sum_t \frac{\partial E_t}{\partial W} \tag{6}$$

Para $\frac{\partial E_t}{\partial V}$ resolver esto es fáci, ya que únicamente requerimos conocer $\hat{y}_t, y_t, s_t$. Para $t = 3$ tenemos la sigueinte derivada.

$$\frac{\partial E_3}{\partial V} = \frac{\partial E_3}{\partial \hat{y}_3} \frac{\partial \hat{y}_3}{\partial V} \tag{7}$$

$$= \frac{\partial E_3}{\partial \hat{y}_3} \frac{\partial \hat{y}_3}{\partial z_3} \frac{\partial z_3}{\partial V} \tag{8}$$

$$= (\hat{y}_3 - y_3) \otimes s_3 \tag{9}$$

Donde $z_3 = Vs_3$ y $\otimes$ es el producto exterior de dos vectores.

pero para W y U las cosas nos son tan simples.

$$\frac{\partial E_3}{\partial W} = \frac{\partial E_3}{\partial \hat{y}_3}\frac{\partial \hat{y}_3}{\partial s_3}\frac{\partial s_3}{\partial W} \tag{10}$$

$$\tag{11}$$

Dado que $s_3 = \tanh(Ux_3 + Ws_2)$ este depende de s_2 que depende de W y $s1$, etc. Por ello debemos expandir la regla de la cadena.

$$\frac{\partial E_3}{\partial W} = \frac{\partial E_3}{\partial \hat{y}_3}\frac{\partial \hat{y}_3}{\partial s_3}\frac{\partial s_3}{\partial s_k}\frac{\partial s_k}{\partial W} \tag{12}$$

$$\tag{13}$$

La diferecia con una red neuronal no recurrente es que en las RNN sumamos las grdientes para W en cada paso de tiempo.

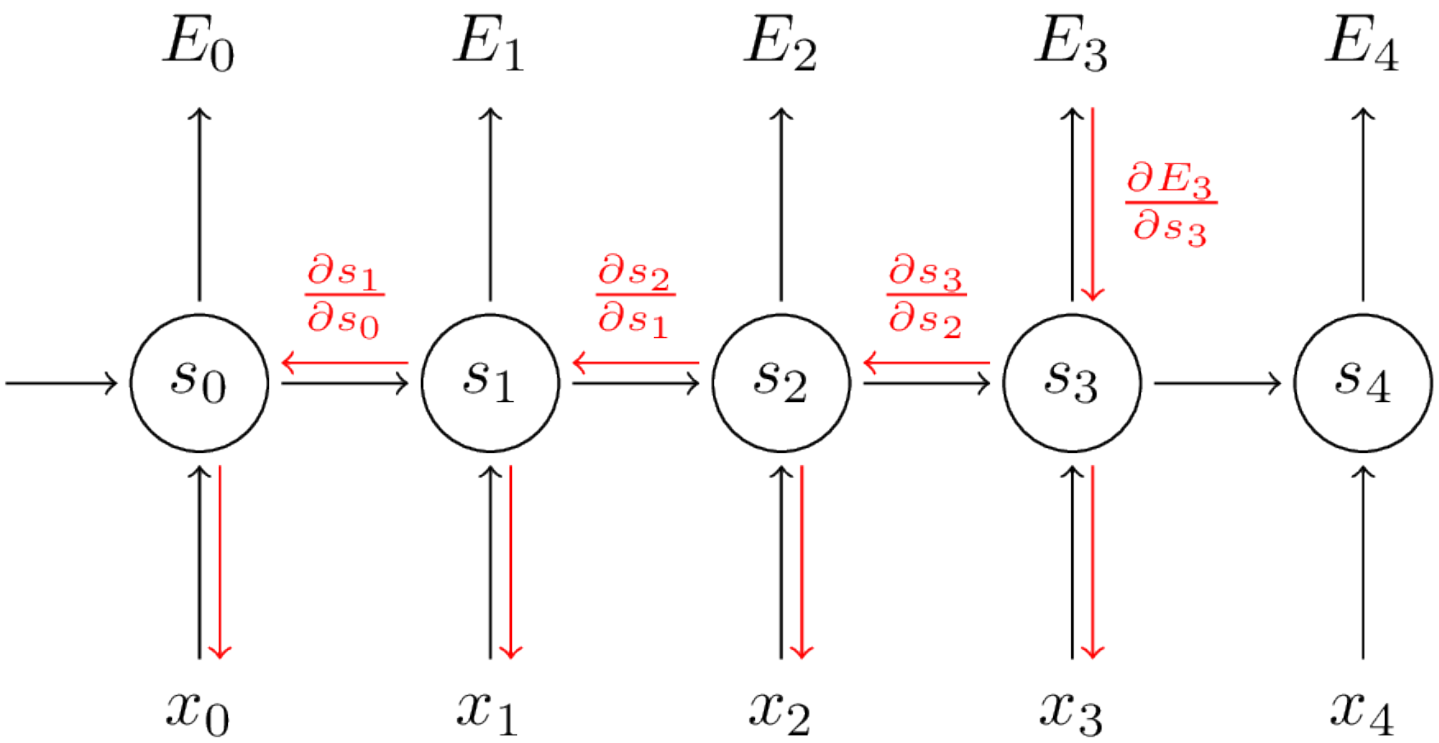


Figure: Fuente: `http://www.wildml.com`

Problemas de las RNN

1. RNN son difíciles de entrnar.
2. No logran aprender dependencias sobre secuencias largas.
3. Esto se debe al problema del desvanecimiento del gradiente

El problema del desvanecimiento del gradiente

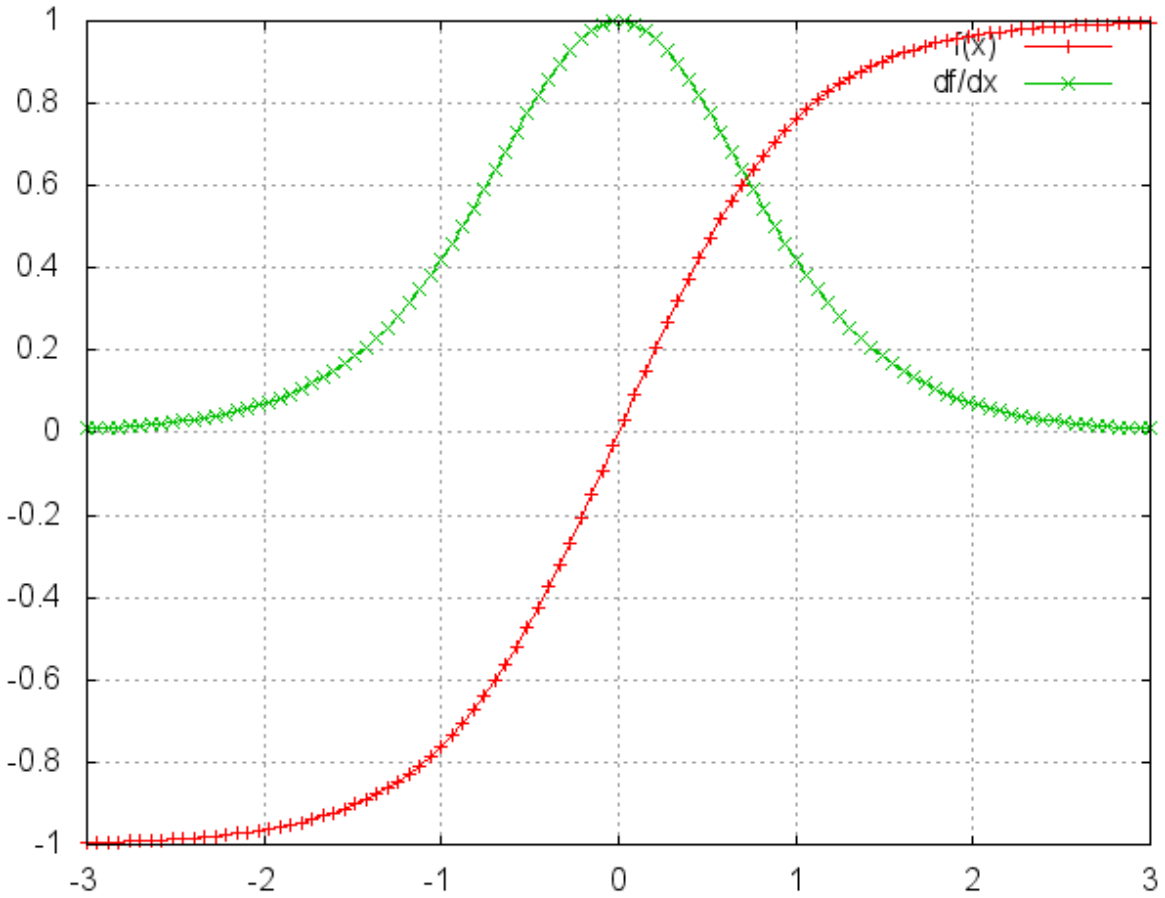


Figure: Fuente: `http://nn.readthedocs.org/en/rtd/transfer/`

¿Qué es una LSTM?

1. Una red Long-Short Term Memory (LSTM) es un tipo de red neuronal recursova.
2. fue inventada por Hochreiter & Schmidhuber (1997).
3. Fue diseñada espesíficamente para evitar el problema de aprender largas dependencias.

Excelente lectura: `https://colah.github.io/posts/2015-08-Understanding-LSTMs/`

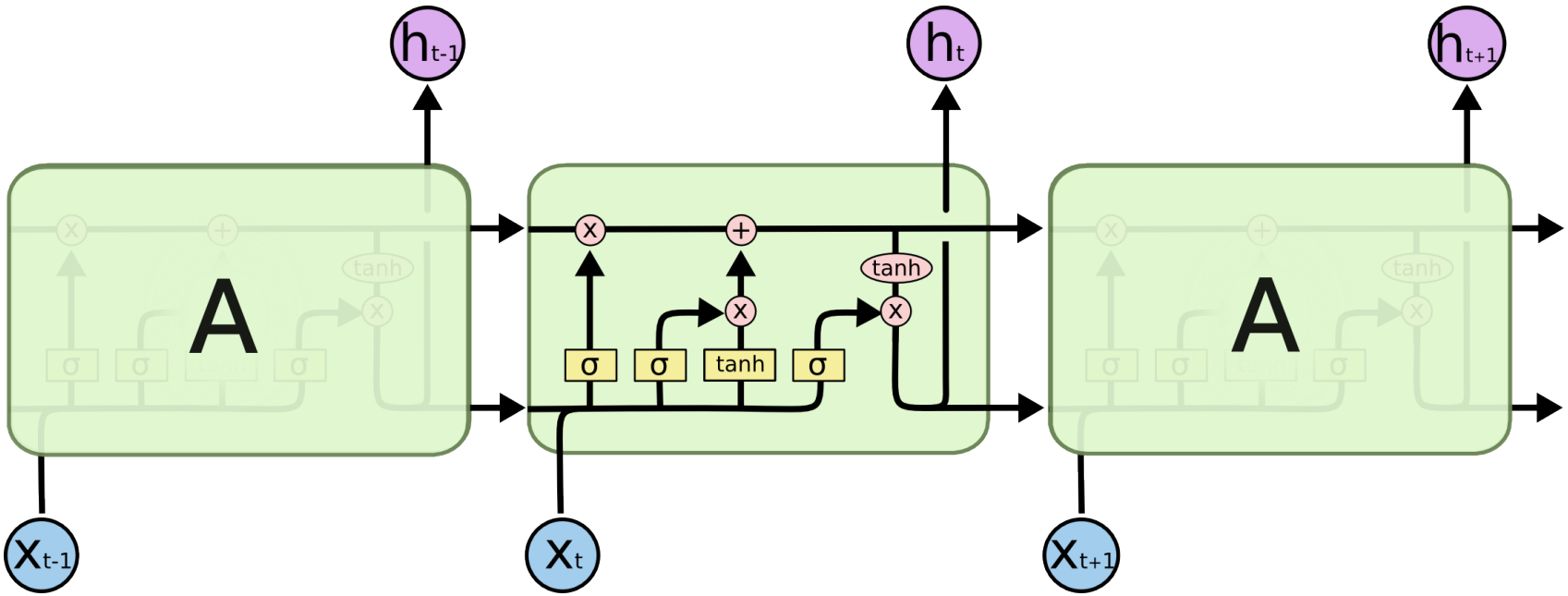
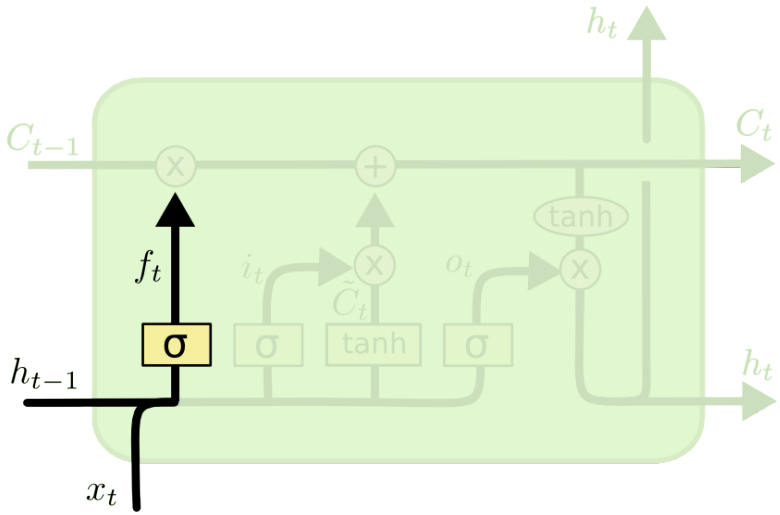


Figure: Fuente: `https://colah.github.io/`

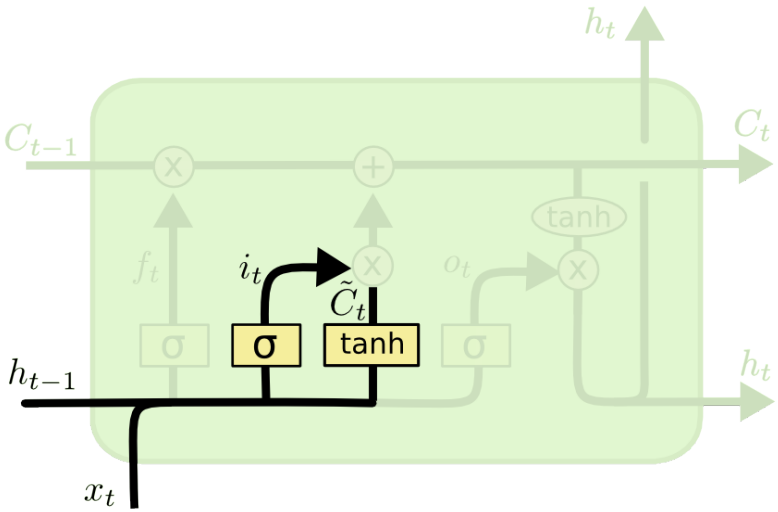
Forget gate layer



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

Figure: Fuente: `https://colah.github.io/`

Input gate layer

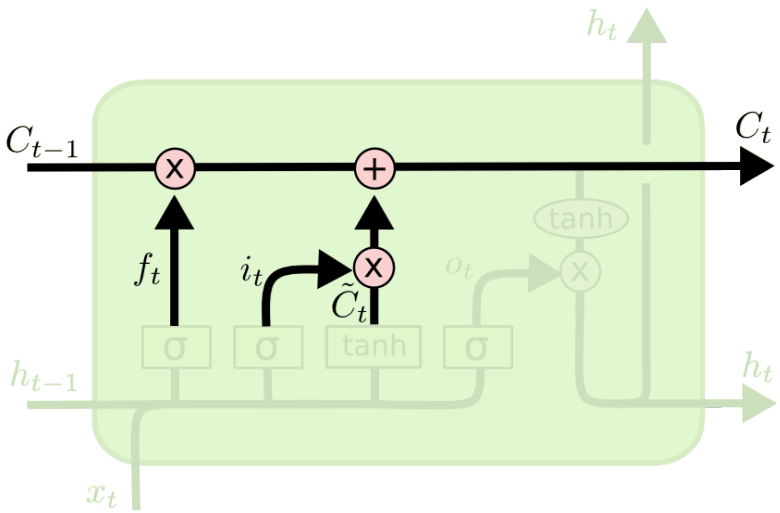


$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

Figure: Fuente: `https://colah.github.io/`

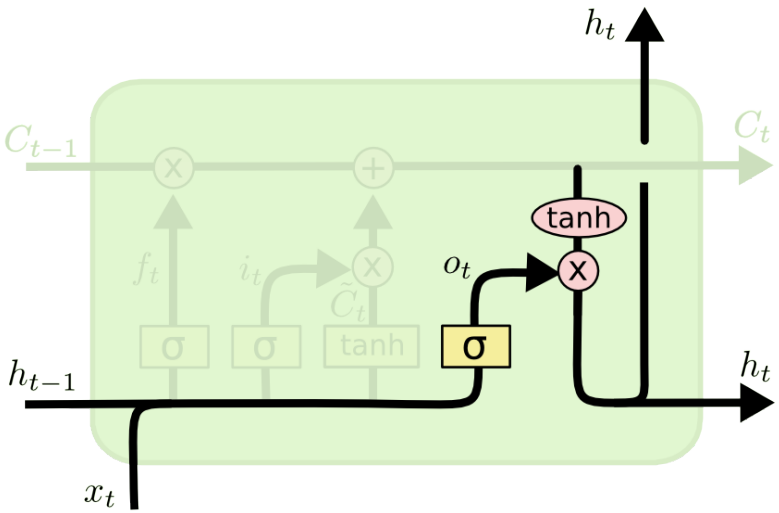
Actualizar C_t



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Figure: Fuente: `https://colah.github.io/`

Salida

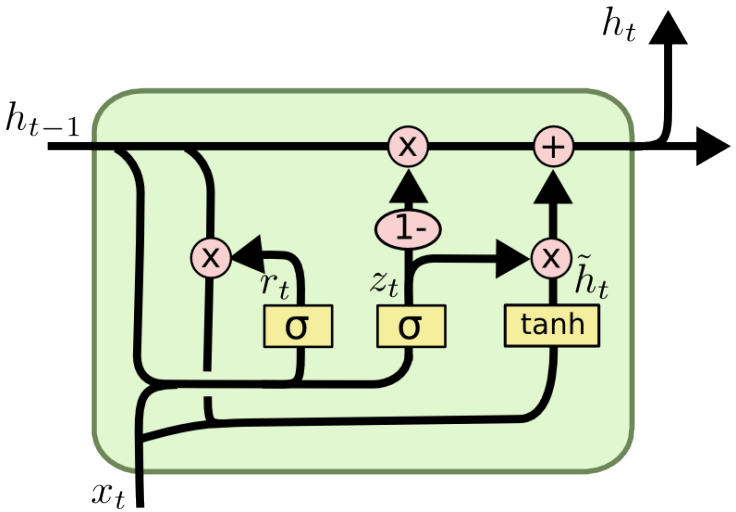


$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right)$$

$$h_t = o_t * \tanh\left(C_t\right)$$

Figure: Fuente: `https://colah.github.io/`

GRU



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

Figure: Fuente: `https://colah.github.io/`

Tipos de arquitecturas

- Uno a uno (Etiquetado de secuencias, inferencia de fenómenos ocultos.)
- Muchos a uno (Generación de descripciones de imágenes)
- Uno a muchos (Clasificación de textos)
- Muchos a muchos (Traducción, resumen)

Tipos de arquitecturas

- Uno a uno (Etiquetado de secuencias, inferencia de fenómenos ocultos.)
- Muchos a uno (Generación de descripciones de imágenes)
- Uno a muchos (Clasificación de textos)
- Muchos a muchos (Traducción, resumen)

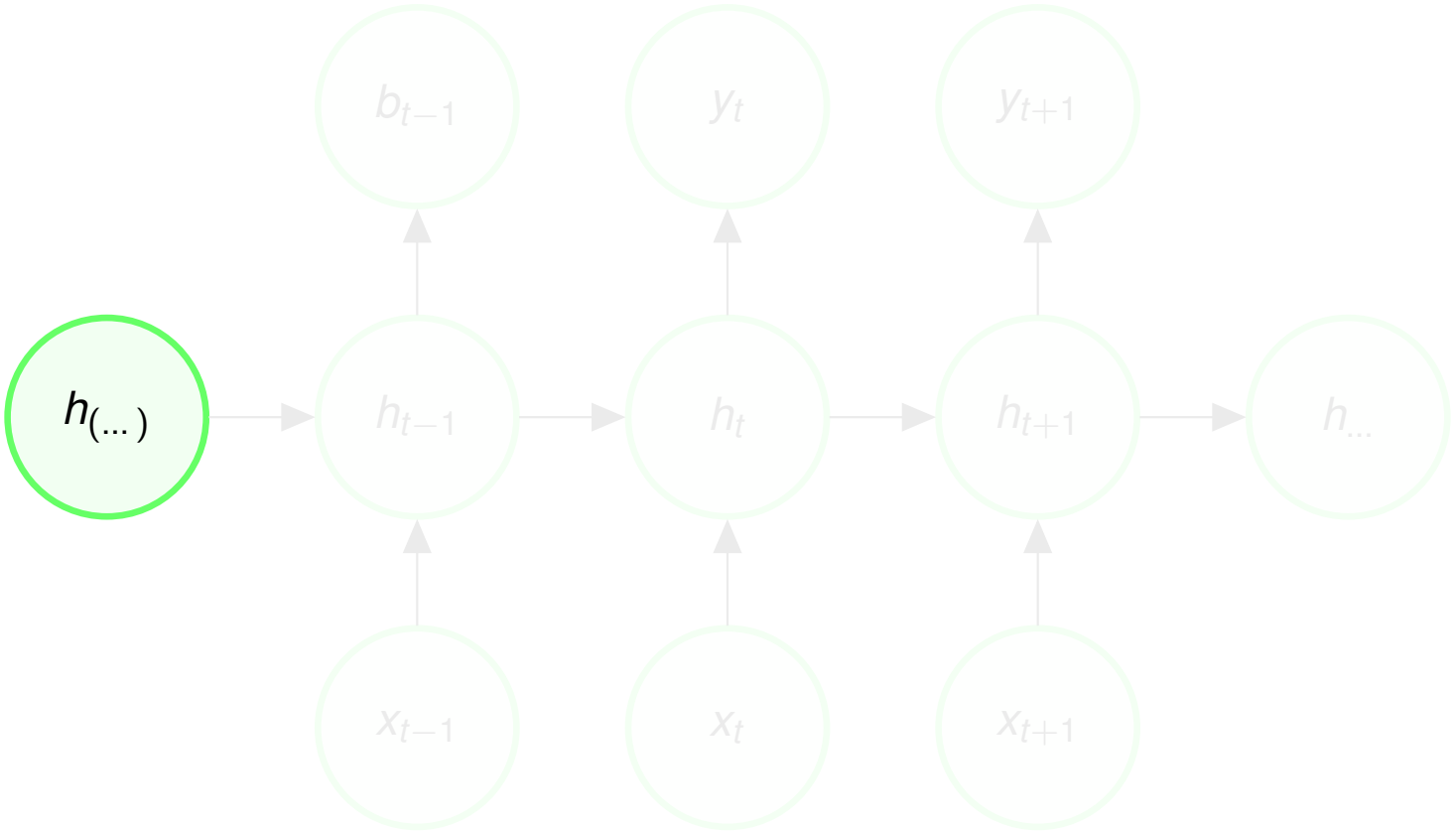
Tipos de arquitecturas

- Uno a uno (Etiquetado de secuencias, inferencia de fenómenos ocultos.)
- Muchos a uno (Generación de descripciones de imágenes)
- Uno a muchos (Clasificación de textos)
- Muchos a muchos (Traducción, resumen)

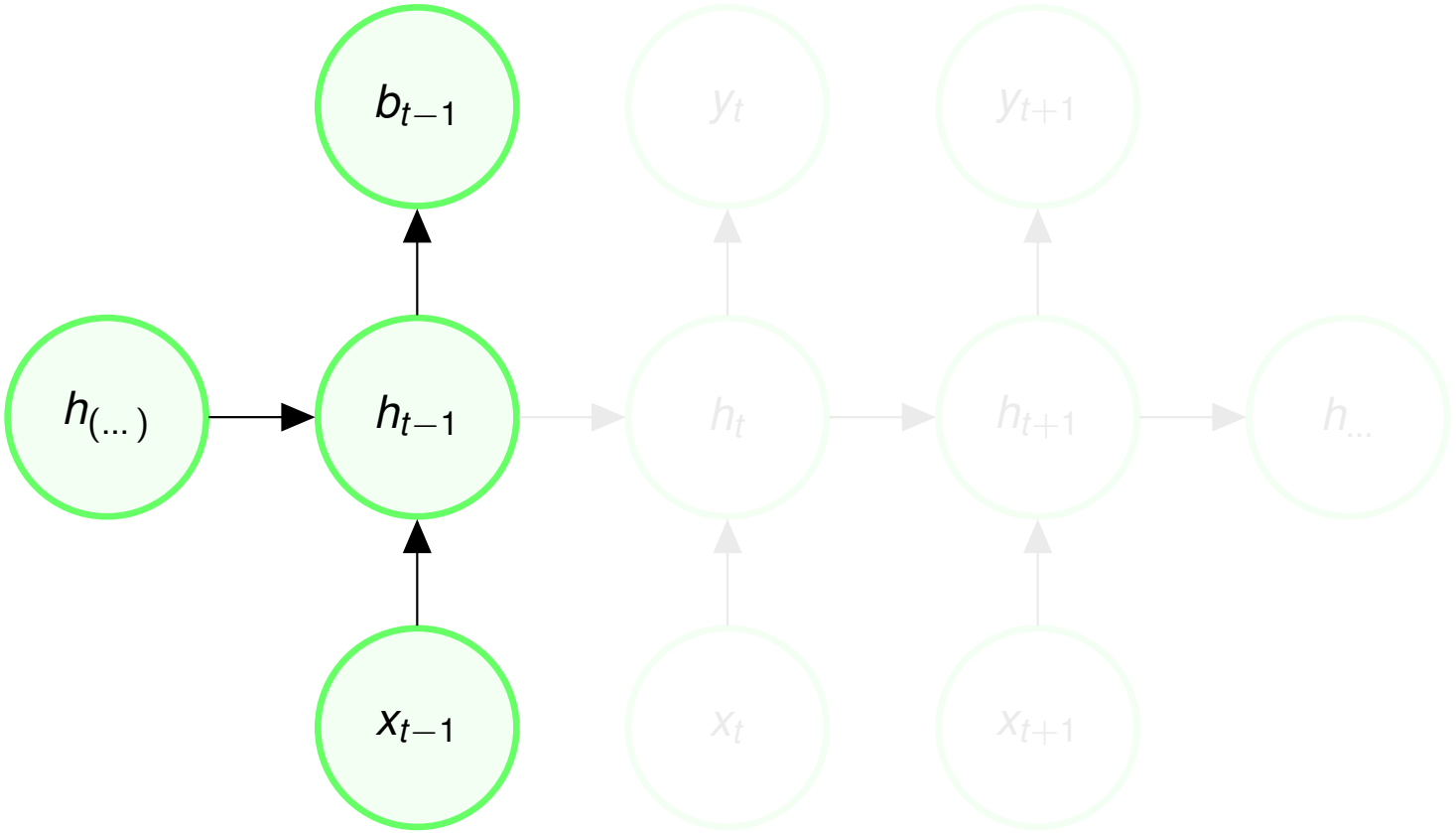
Tipos de arquitecturas

- Uno a uno (Etiquetado de secuencias, inferencia de fenómenos ocultos.)
- Muchos a uno (Generación de descripciones de imágenes)
- Uno a muchos (Clasificación de textos)
- Muchos a muchos (Traducción, resumen)

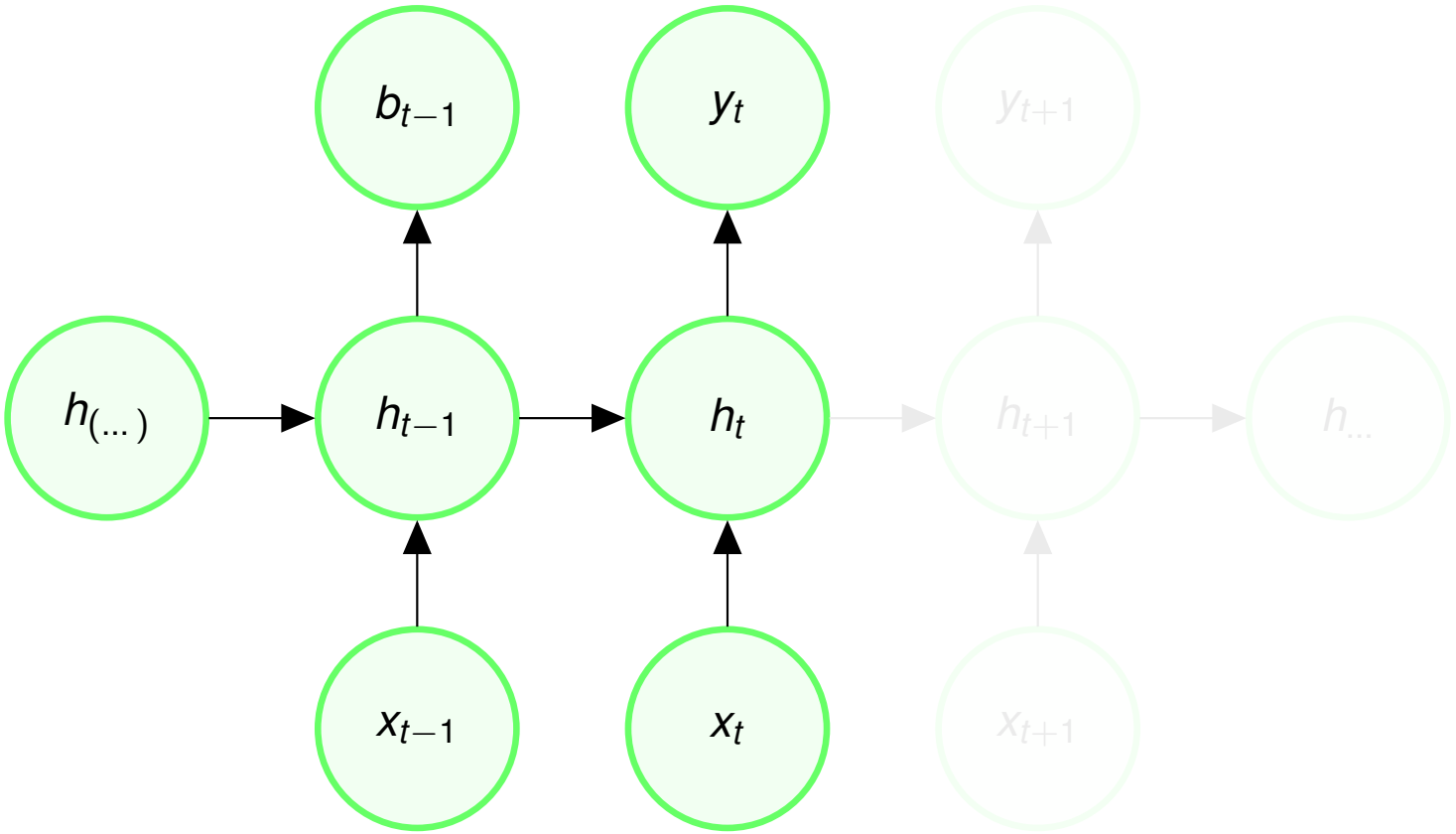
Uno a uno



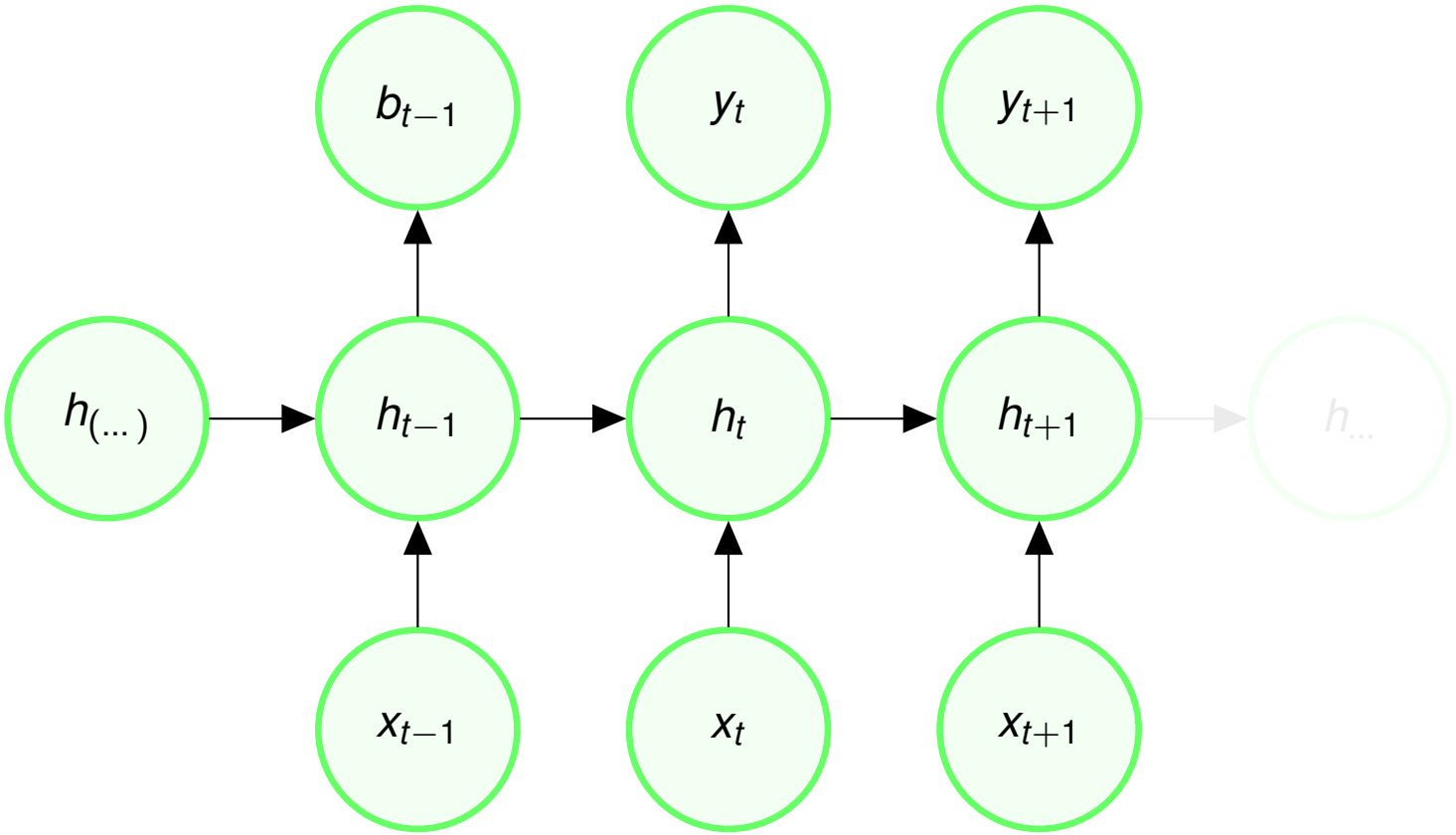
Uno a uno



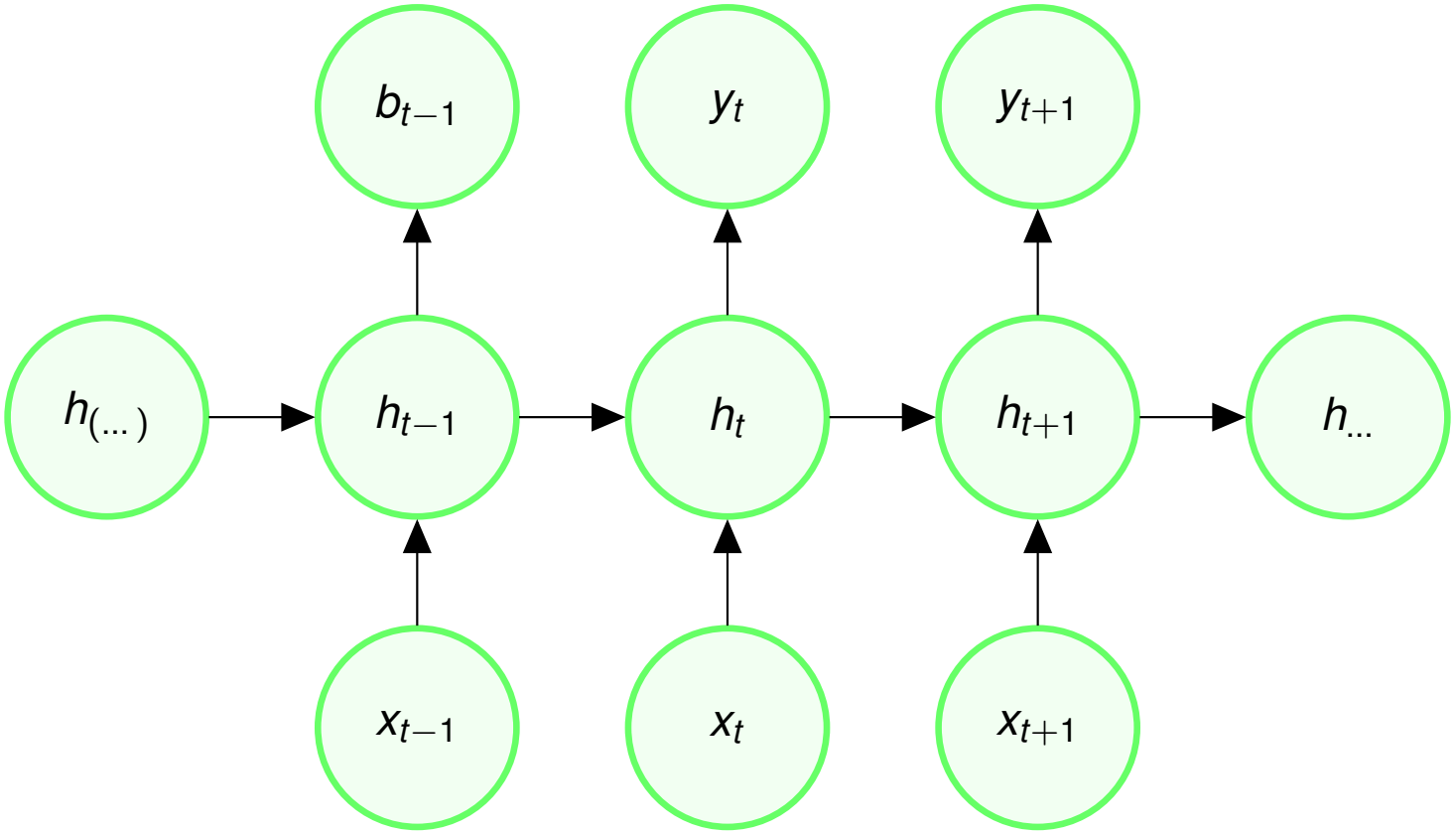
Uno a uno



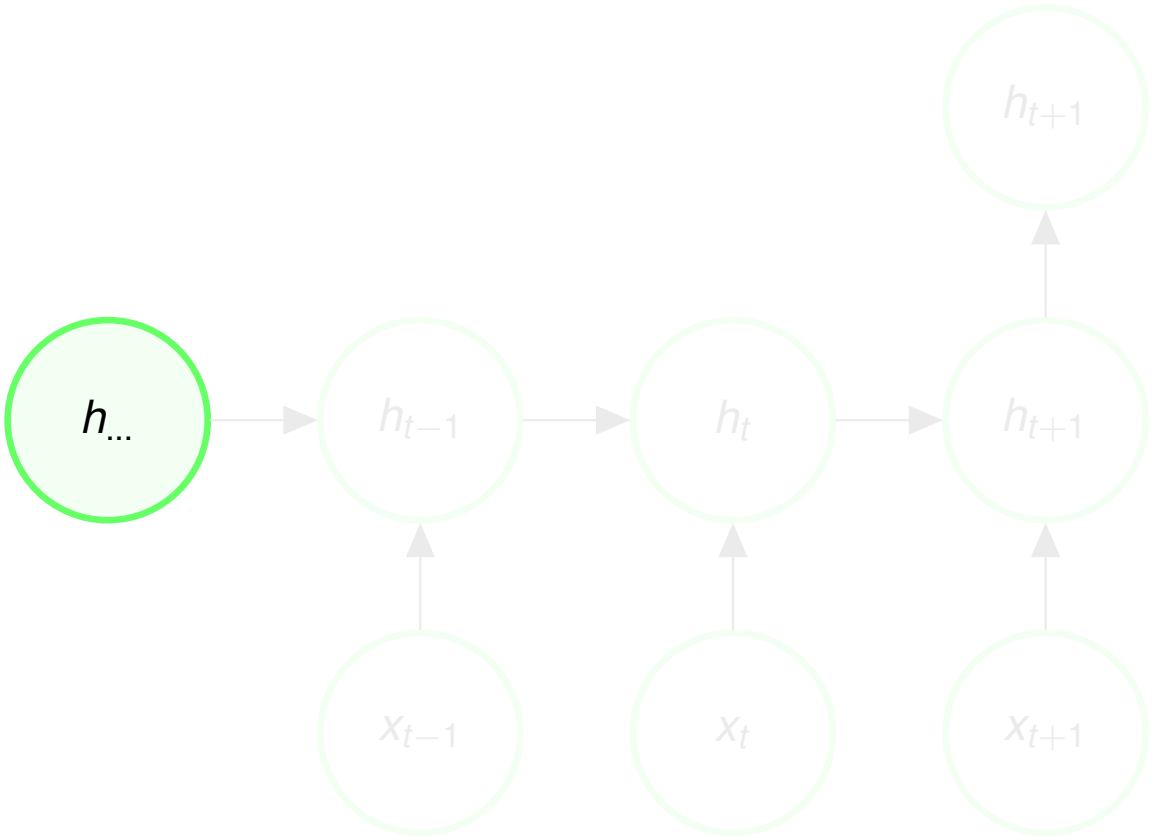
Uno a uno



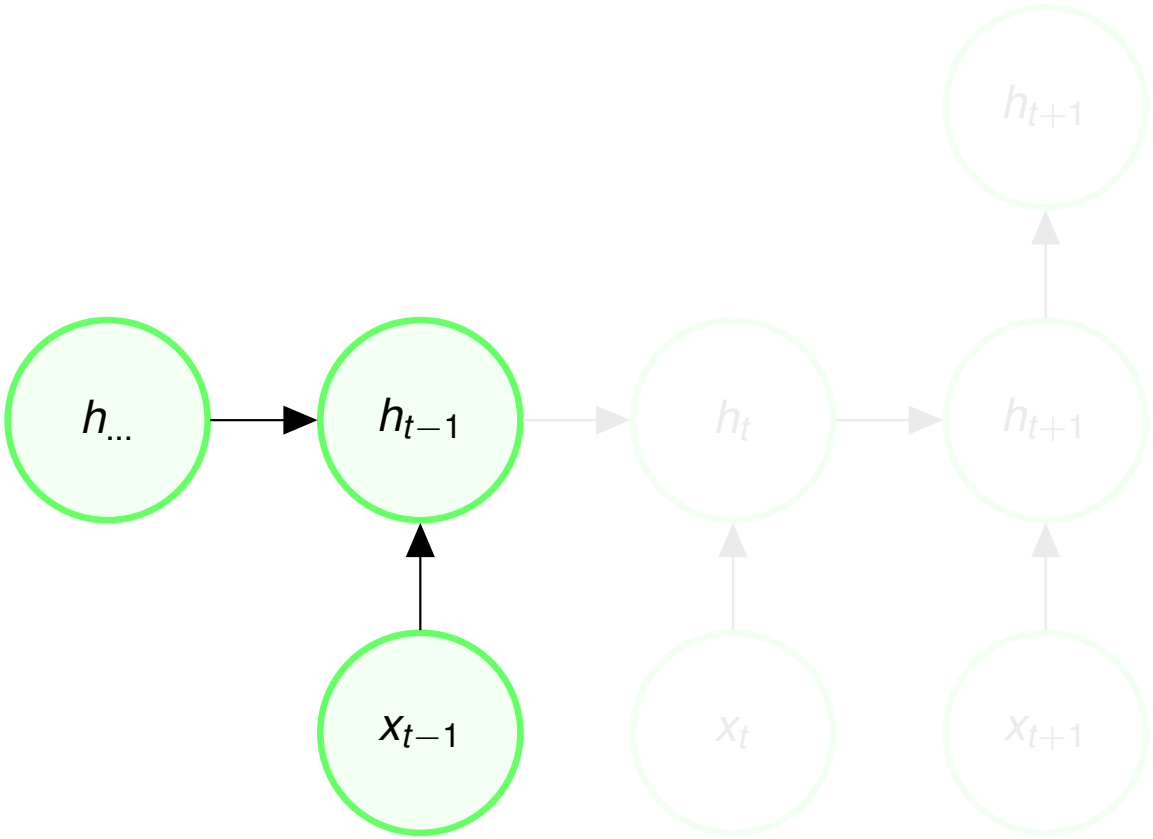
Uno a uno



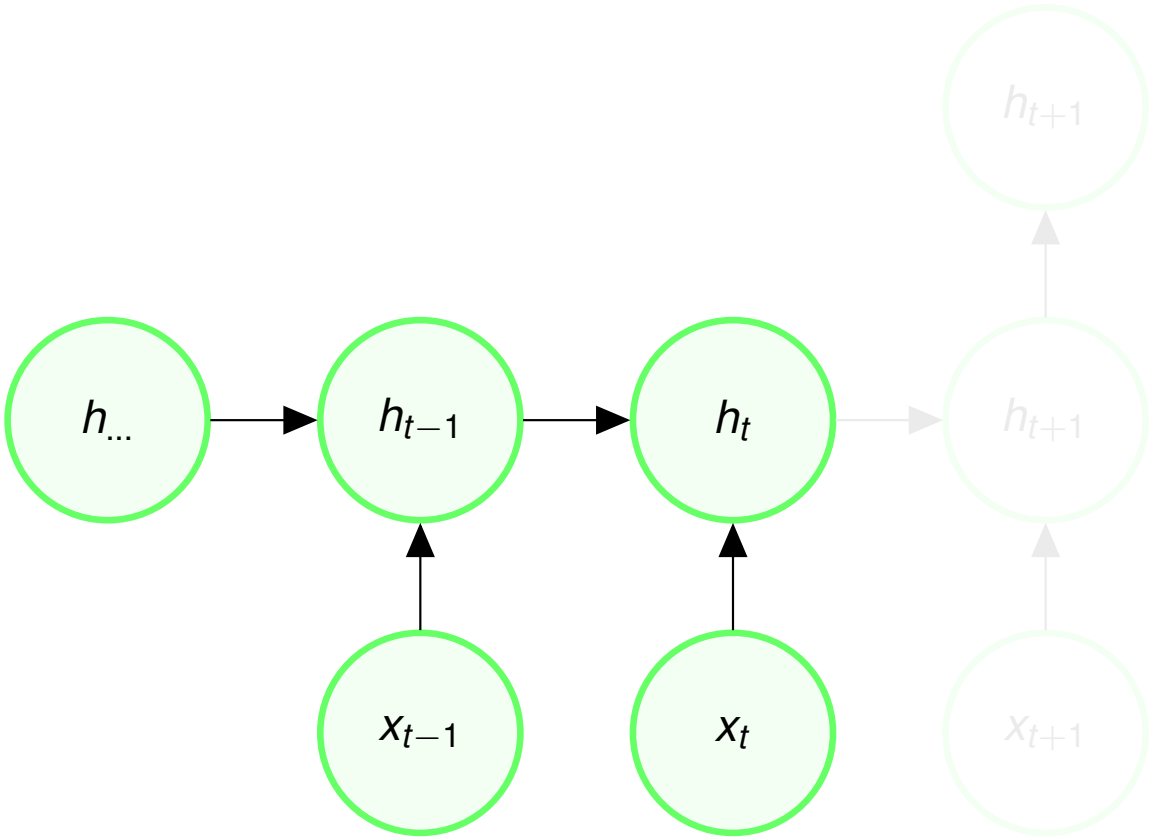
Varios a uno



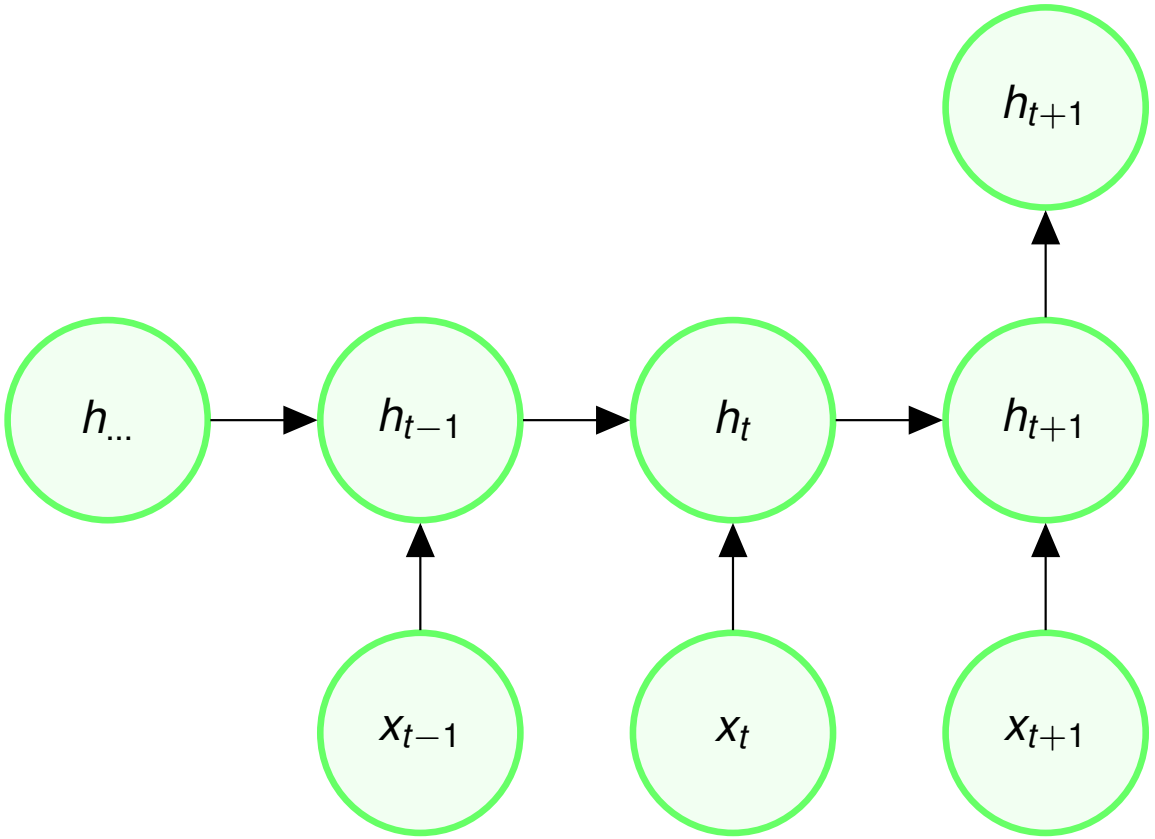
Varios a uno



Varios a uno

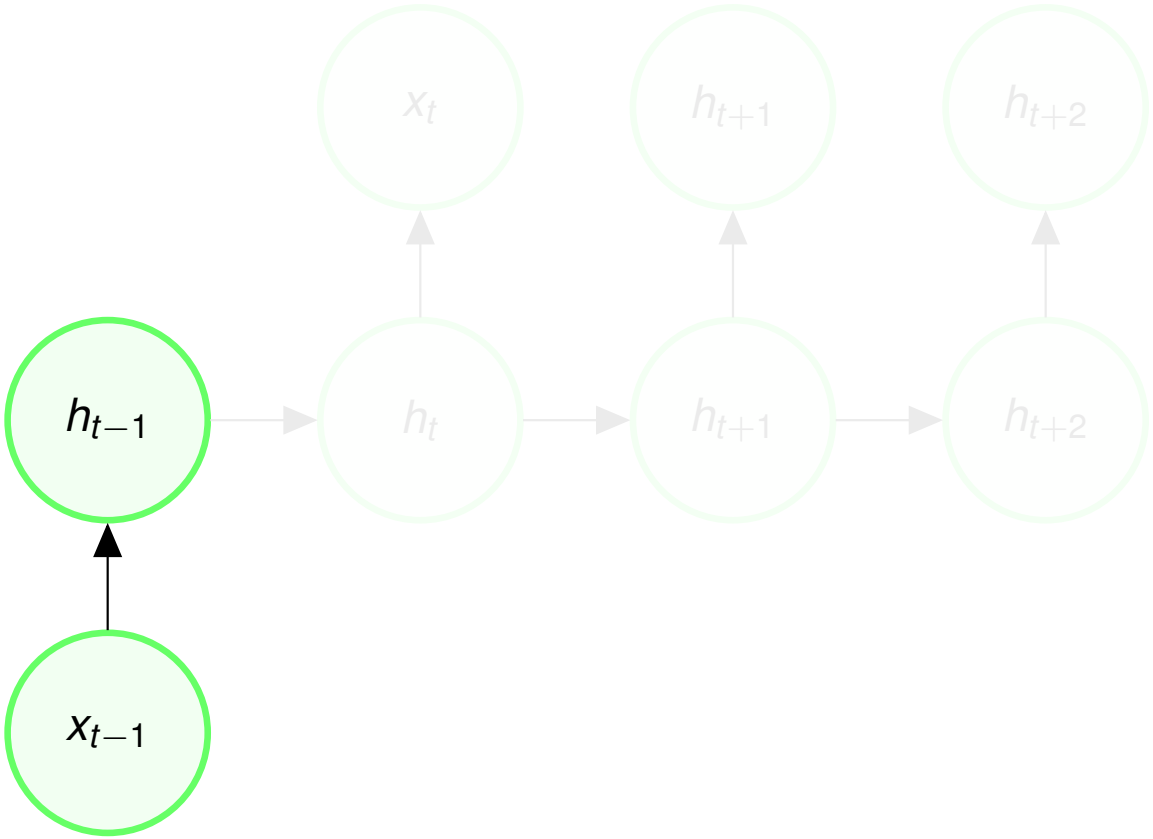


Varios a uno

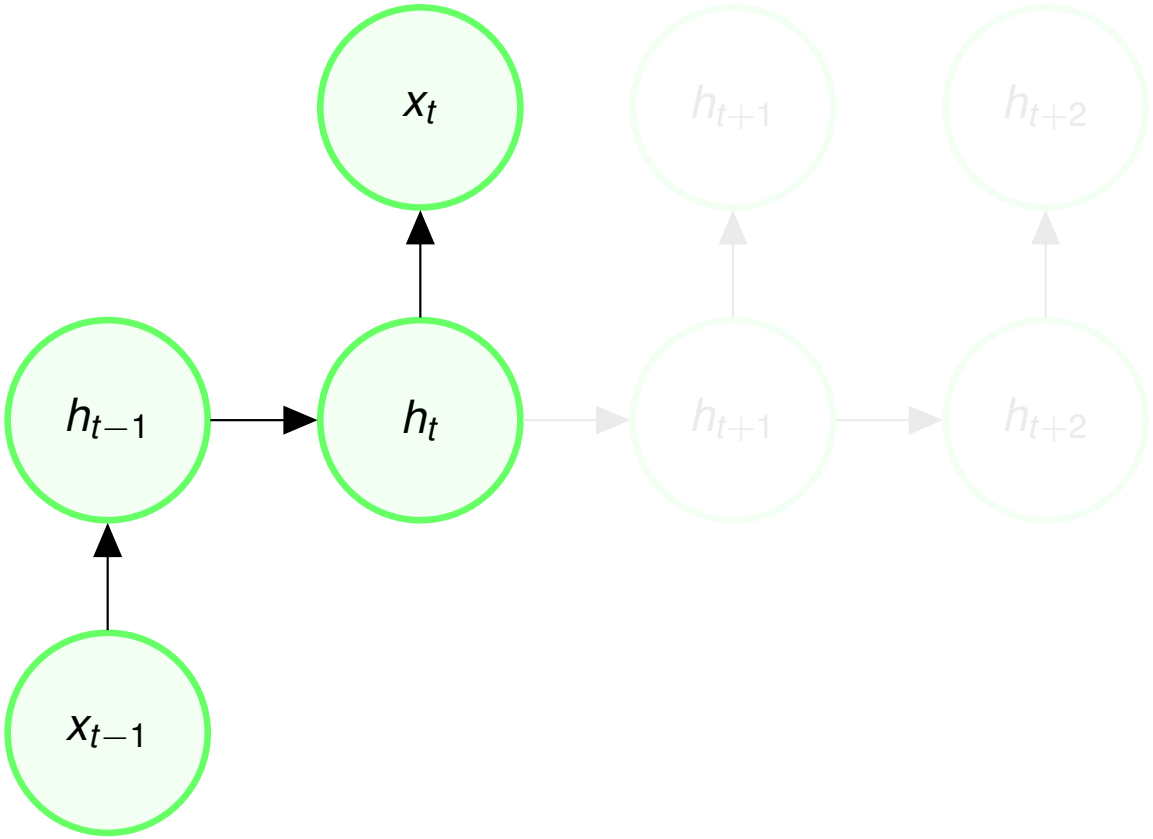


Uno a varios

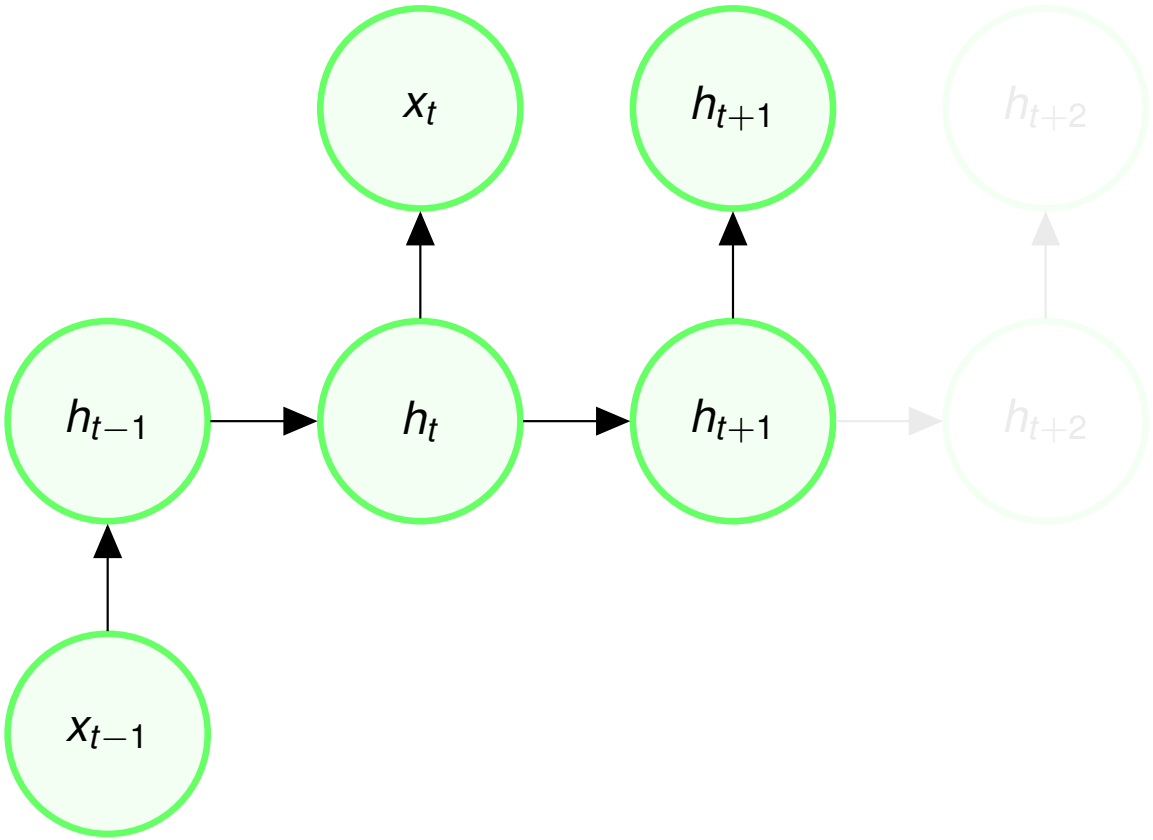
Uno a varios



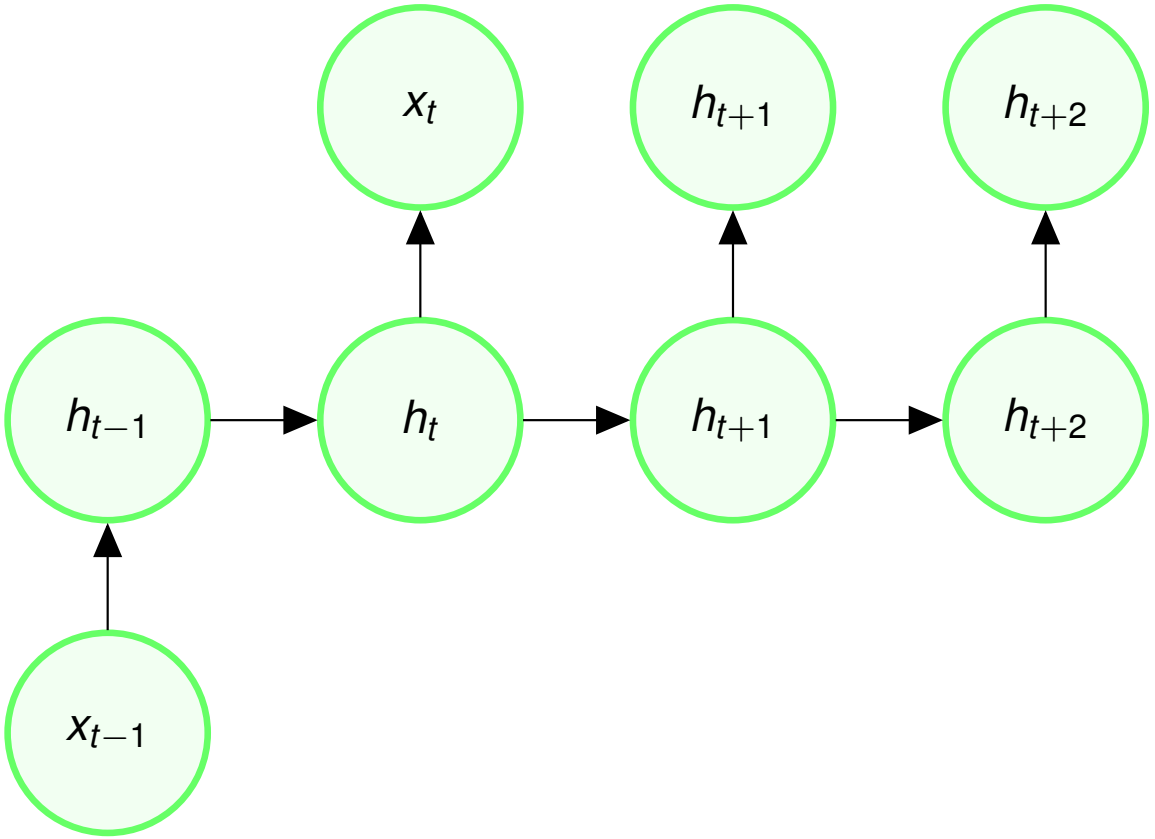
Uno a varios



Uno a varios

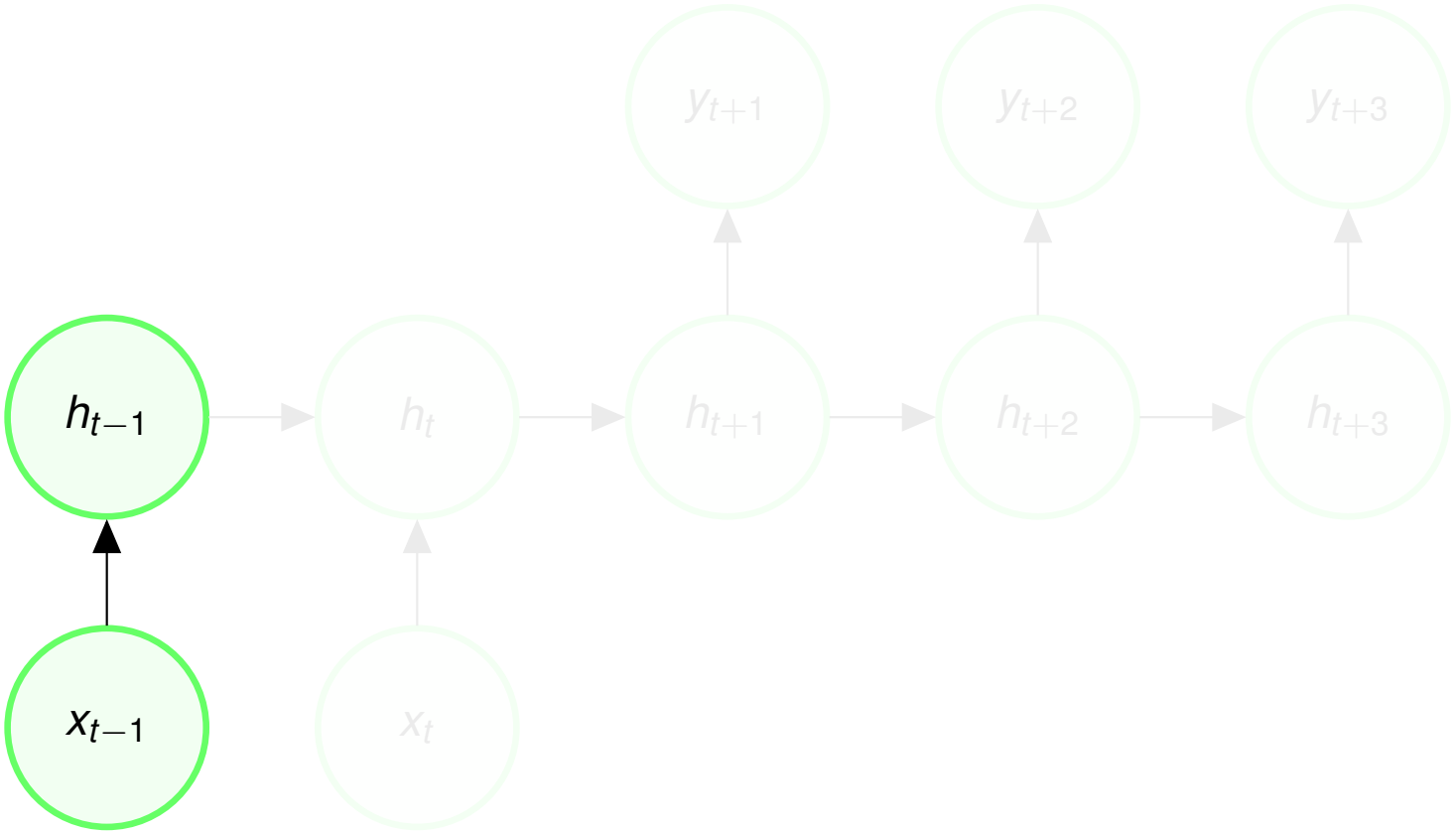


Uno a varios

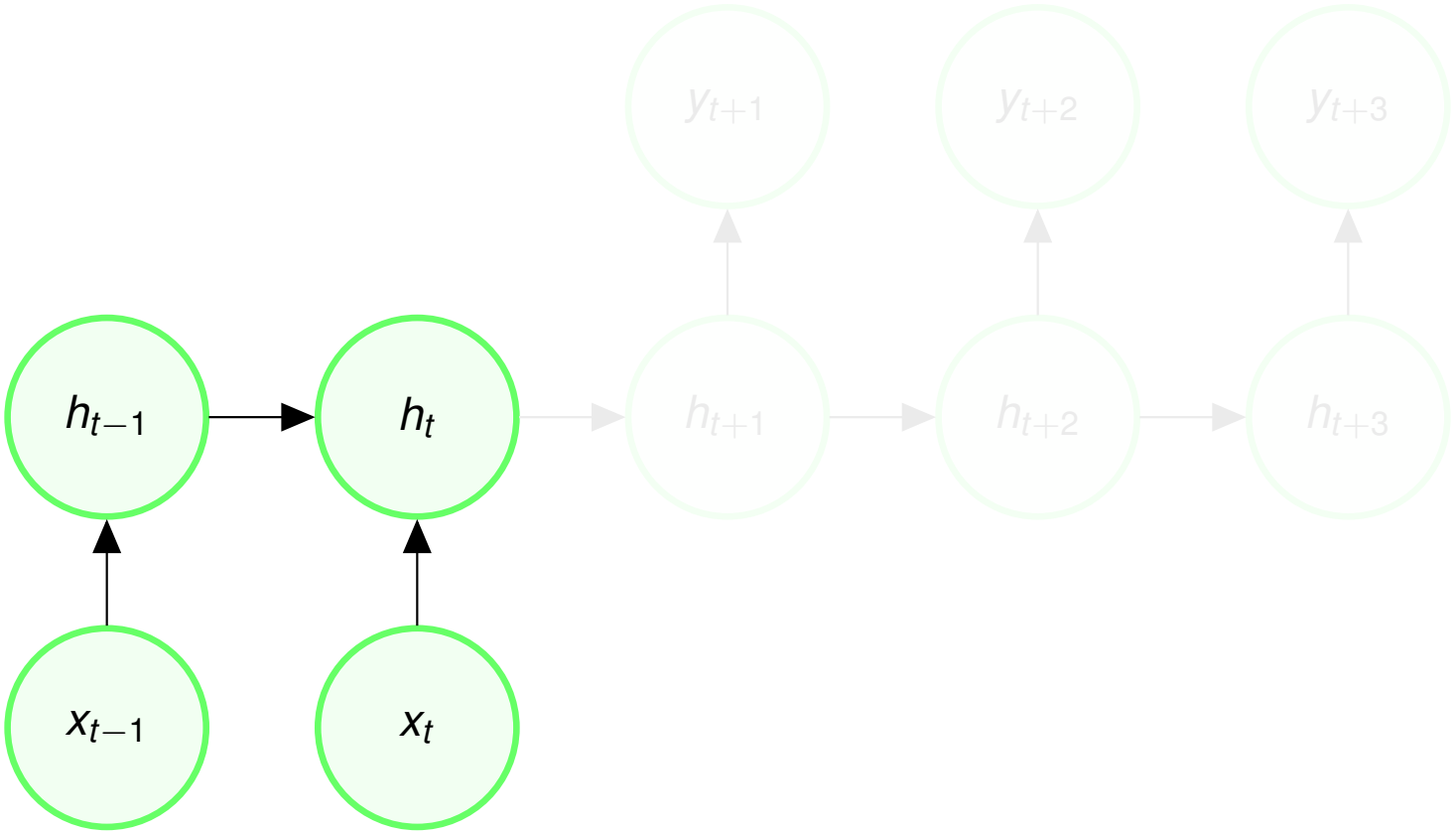


Varios a varios

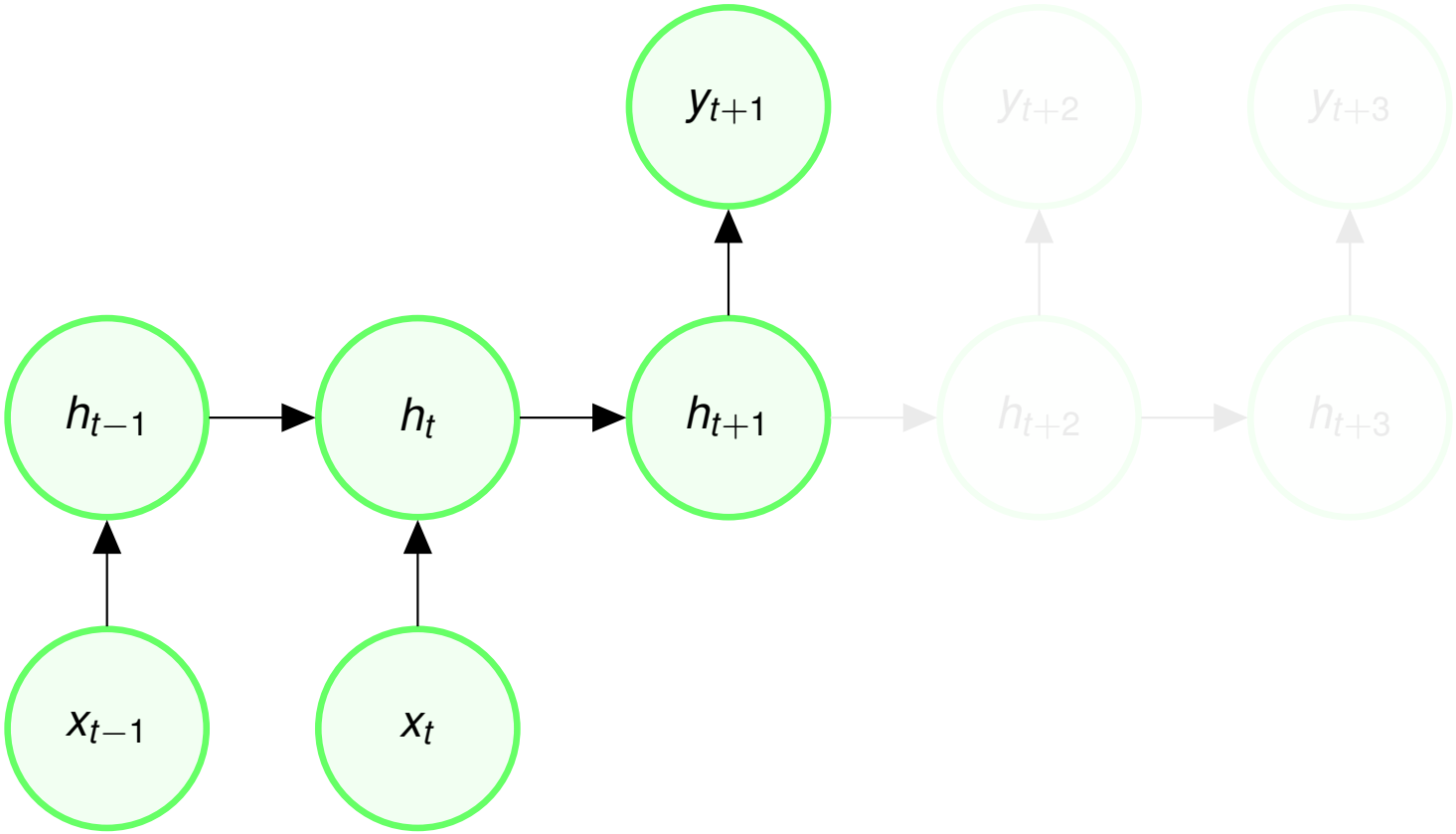
Varios a varios



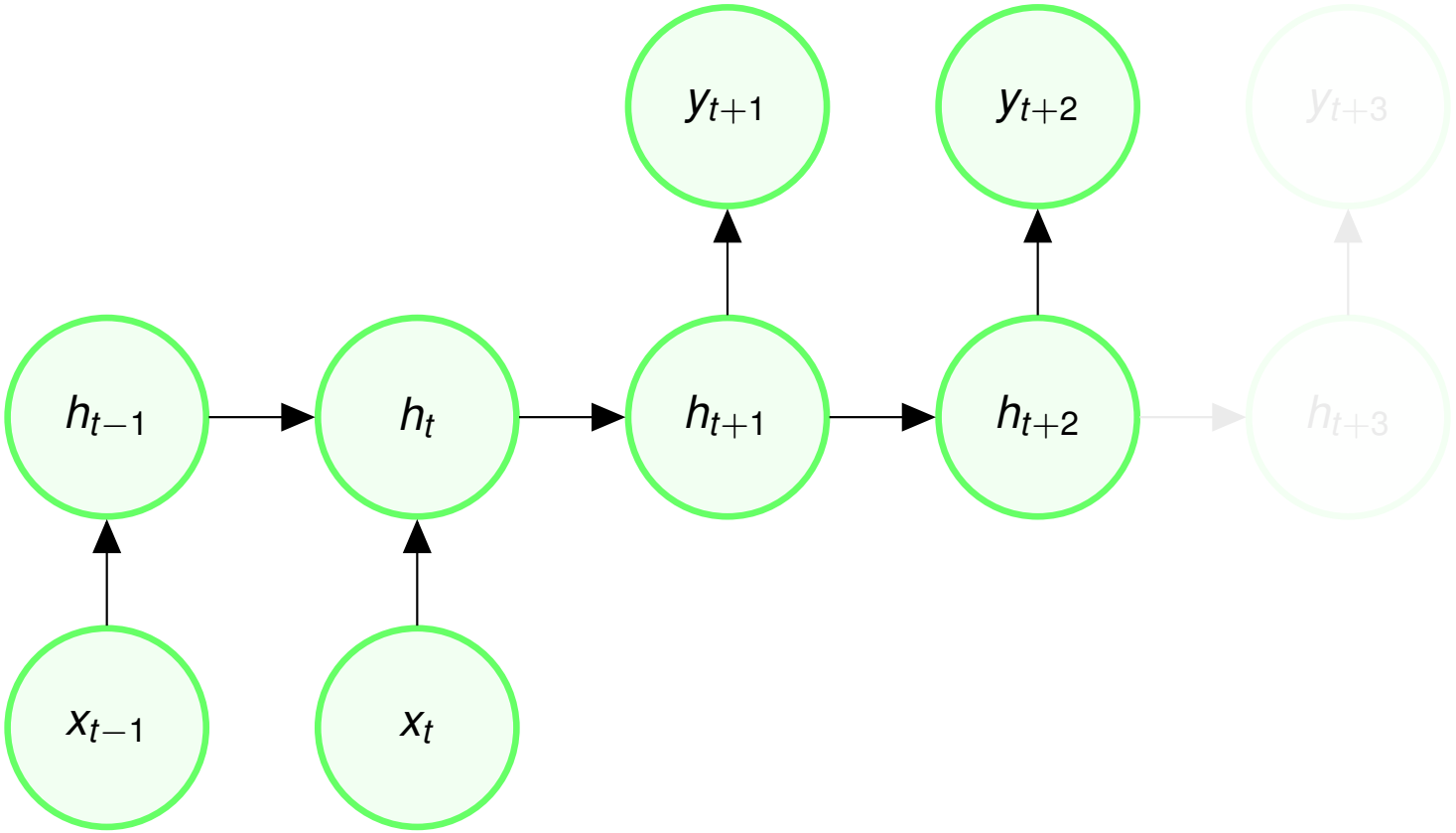
Varios a varios



Varios a varios



Varios a varios



Varios a varios