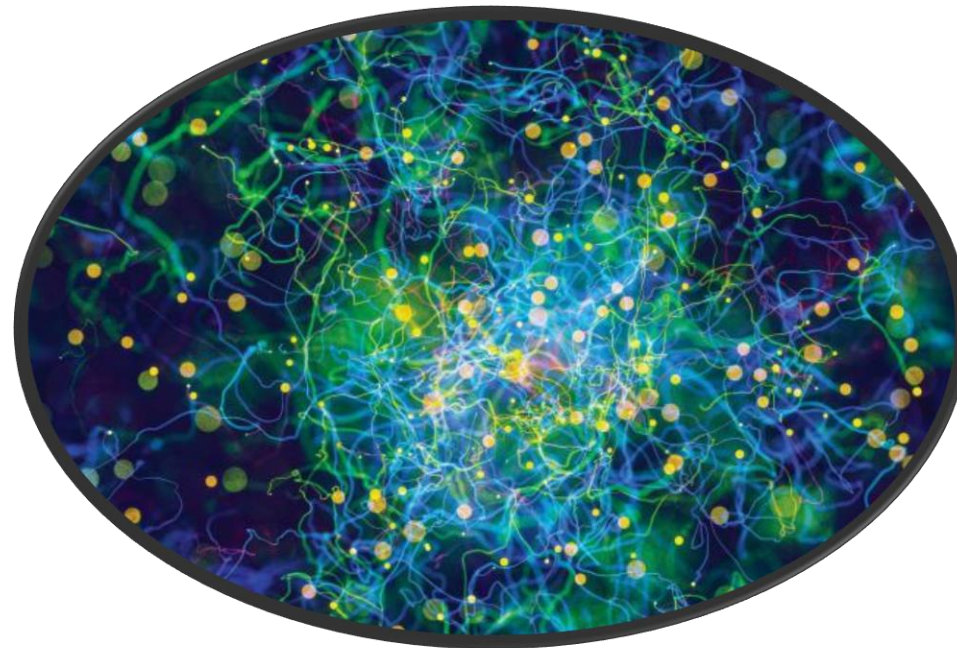


Deep Learning



Día 6


EXPOSITOR: Ing. Giorgio Morales Luna

Junio 2018



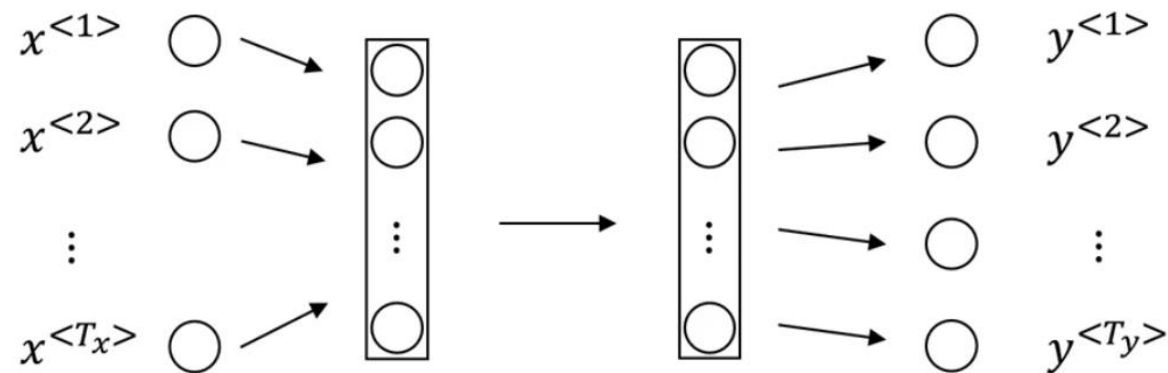
6. Modelos Secuenciales

6.1. Aplicaciones

Reconocimiento de voz		→	<i>"The quick brown fox jumps over the lazy dog"</i>
Generación de música		→	
Análisis de sentimiento	"La película fue malísima"	→	
Traducción	"I am hungry"	→	"Tengo hambre"
Reconocimiento de acciones en video		→	"Corriendo"
Reconocimiento de nombres	"Ayer, Harry Potter conoció a Hermione Granger"	→	"Ayer, Harry Potter conoció a Hermione Granger "

6.2. Recurrent Neural Network (RNN)

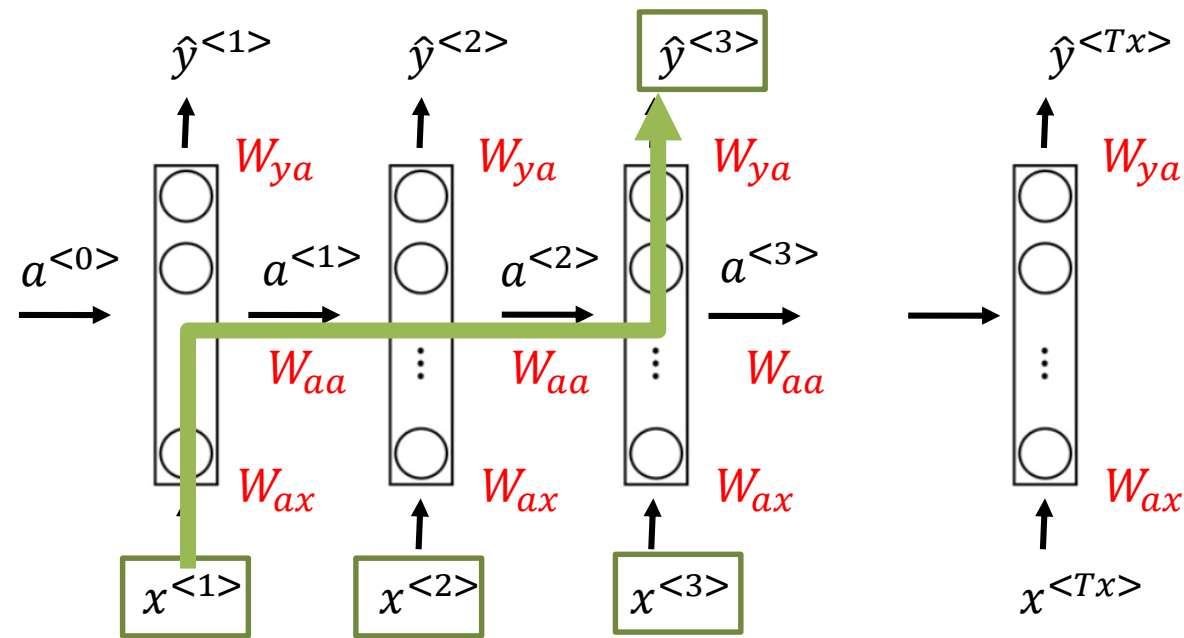
¿Por qué no usar una red neuronal estándar?



Problemas:

- Las entradas y salidas pueden tener distintas dimensiones en distintos ejemplos.
- Las características aprendidas por la red no se comparten con otras posiciones del texto.

6.2. Recurrent Neural Network (RNN)

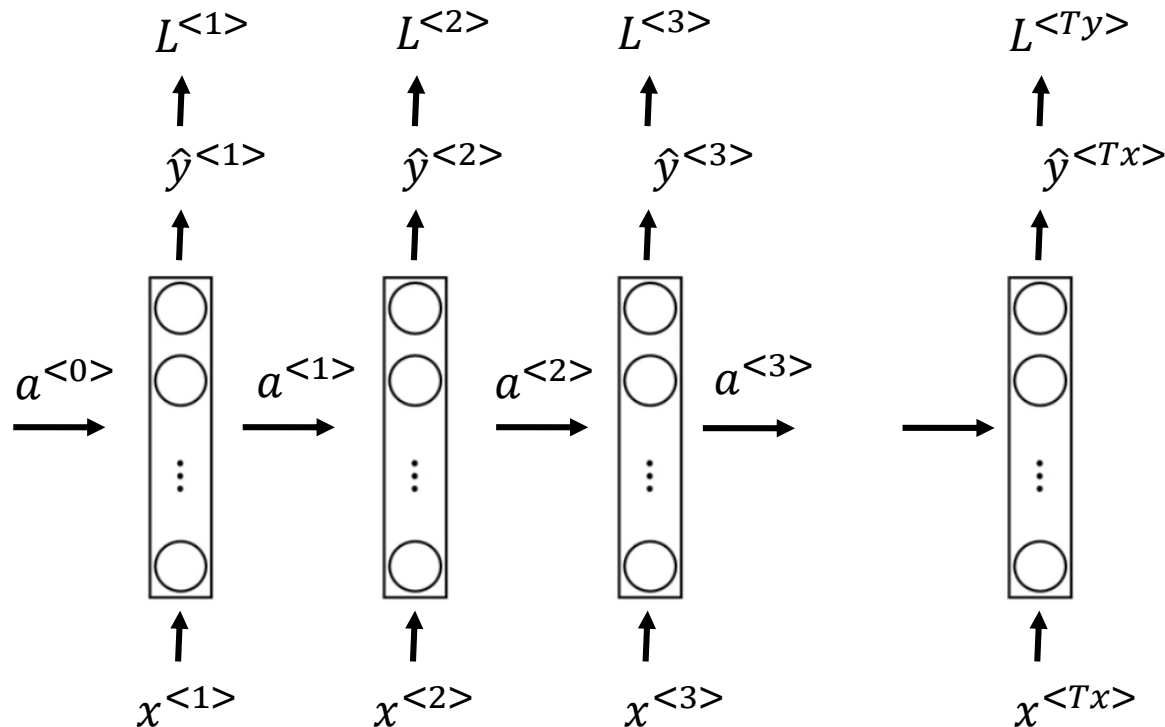


He said “Teddy Roosevelt was a great President”

He said “Teddy bears are on sale!”

6.2. Recurrent Neural Network (RNN)

Backpropagation through time

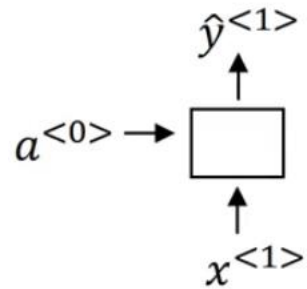


$$L^{<t>}(\hat{y}^{<t>}, y^{<t>}) = -y^{<t>} \log(\hat{y}^{<t>}) - (1 - y^{<t>}) \log(1 - \hat{y}^{<t>})$$

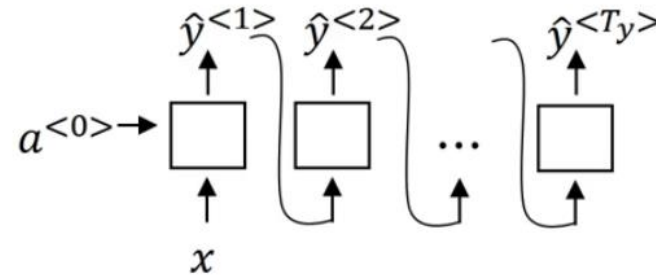
$$L(\hat{y}, y) = \sum_{t=1}^{Ty} L^{<t>}(\hat{y}^{<t>}, y^{<t>})$$

6.2. Recurrent Neural Network (RNN)

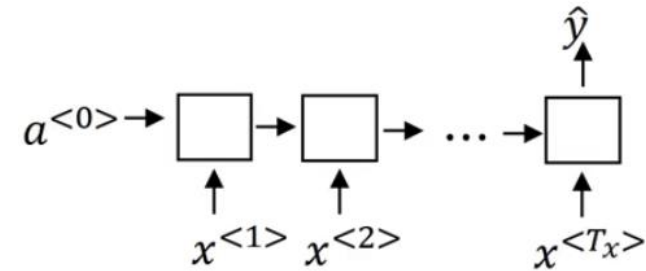
Tipos de RNNs



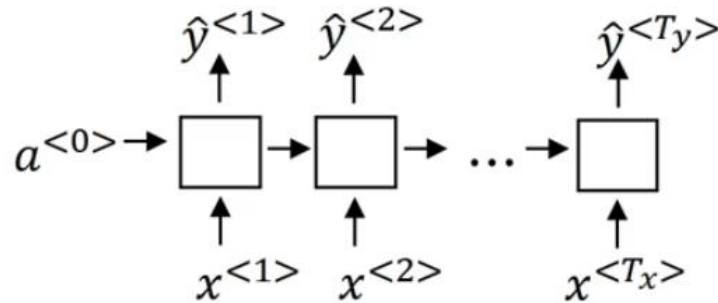
One to one



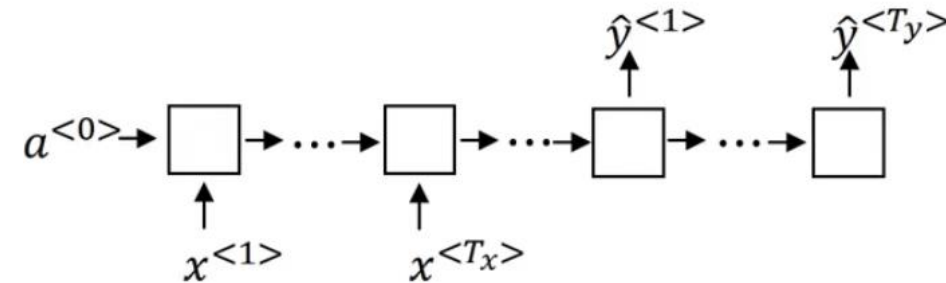
One to many



Many to one



Many to many

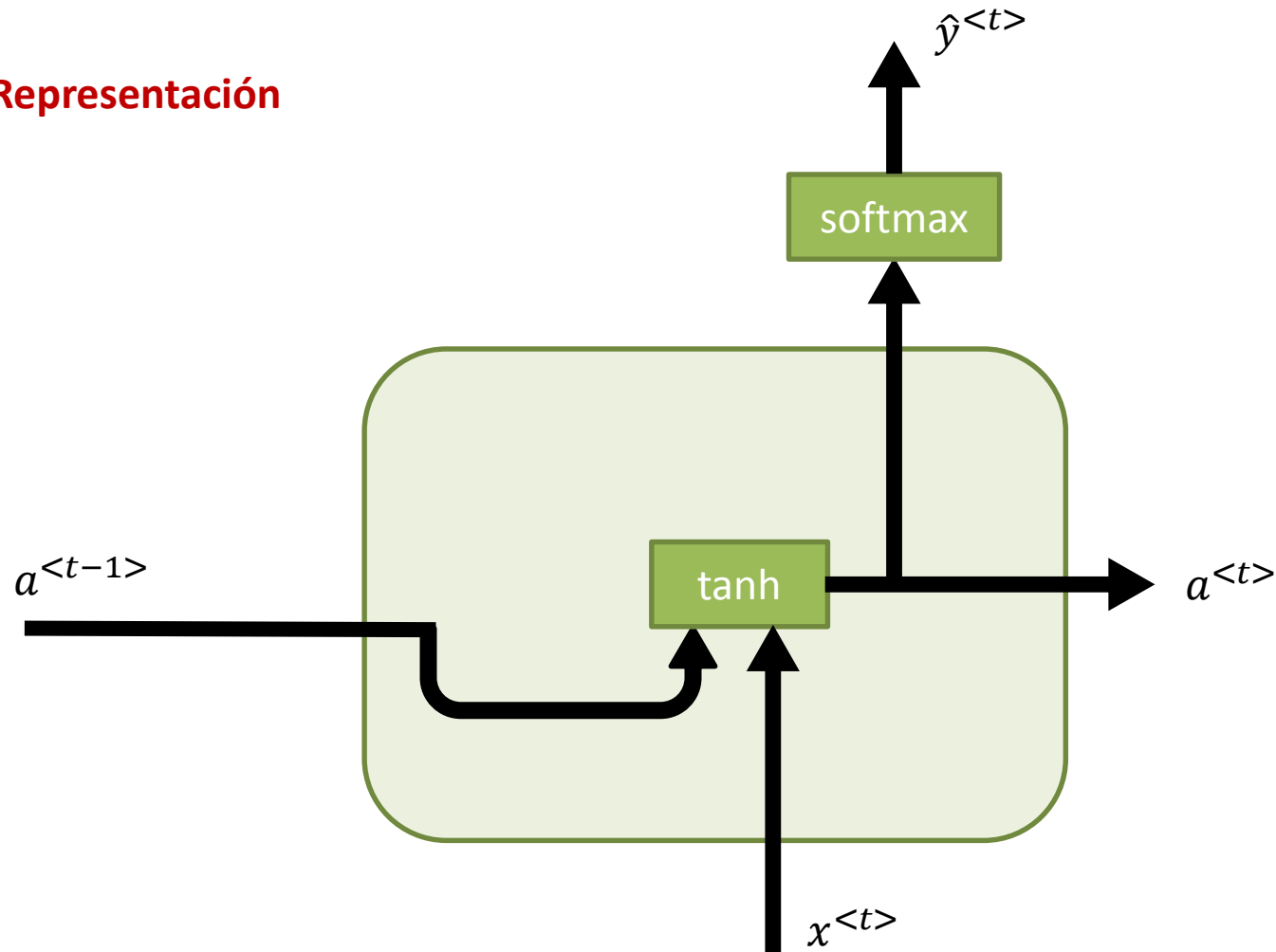


Many to many

<http://karpathy.github.io/2015/05/21/rnn-effectiveness/>

6.2. Recurrent Neural Network (RNN)

Representación



$$a^{<t>} = g(W_a[a^{<t-1>}, x^{<t>}] + b_a)$$

6.3. Gated Recurrent Unit (GRU)

$$c^{<t>} = a^{<t>}$$

$$\tilde{c}^{<t>} = \tanh(W_c[c^{<t-1>}, x^{<t>}] + b_c)$$

$$\Gamma_u = \sigma(W_u[c^{<t-1>}, x^{<t>}] + b_u)$$

$$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + (1 - \Gamma_u) * c^{<t-1>}$$

6.3. Gated Recurrent Unit (GRU)

$$c^{<t>} = a^{<t>}$$

$$\tilde{c}^{<t>} = \tanh(W_c[\Gamma_r * c^{<t-1>}, x^{<t>}] + b_c)$$

$$\Gamma_u = \sigma(W_u[c^{<t-1>}, x^{<t>}] + b_u)$$

$$\Gamma_r = \sigma(W_r[c^{<t-1>}, x^{<t>}] + b_r)$$

$$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + (1 - \Gamma_u) * c^{<t-1>}$$

6.4. Long-Short Term Memory (LSTM)

GRU

$$\tilde{c}^{<t>} = \tanh(W_c[\Gamma_r * c^{<t-1>}, x^{<t>}] + b_c)$$

$$\Gamma_u = \sigma(W_u[c^{<t-1>}, x^{<t>}] + b_u)$$

$$\Gamma_r = \sigma(W_r[c^{<t-1>}, x^{<t>}] + b_r)$$

$$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + (1 - \Gamma_u)c^{<t-1>}$$

LSTM

$$\tilde{c}^{<t>} = \tanh(W_c[\Gamma_r * c^{<t-1>}, x^{<t>}] + b_c)$$

$$\Gamma_u = \sigma(W_u[c^{<t-1>}, x^{<t>}] + b_u)$$

$$\Gamma_f = \sigma(W_f[c^{<t-1>}, x^{<t>}] + b_f)$$

$$\Gamma_o = \sigma(W_o[c^{<t-1>}, x^{<t>}] + b_o)$$

$$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + \Gamma_f * c^{<t-1>}$$

$$a^{<t>} = \Gamma_o * \tanh(c^{<t>})$$

6.4. Long-Short Term Memory (LSTM)

LSTM

$$\tilde{c}^{<t>} = \tanh(W_c[\Gamma_r * c^{<t-1>}, x^{<t>}] + b_c)$$

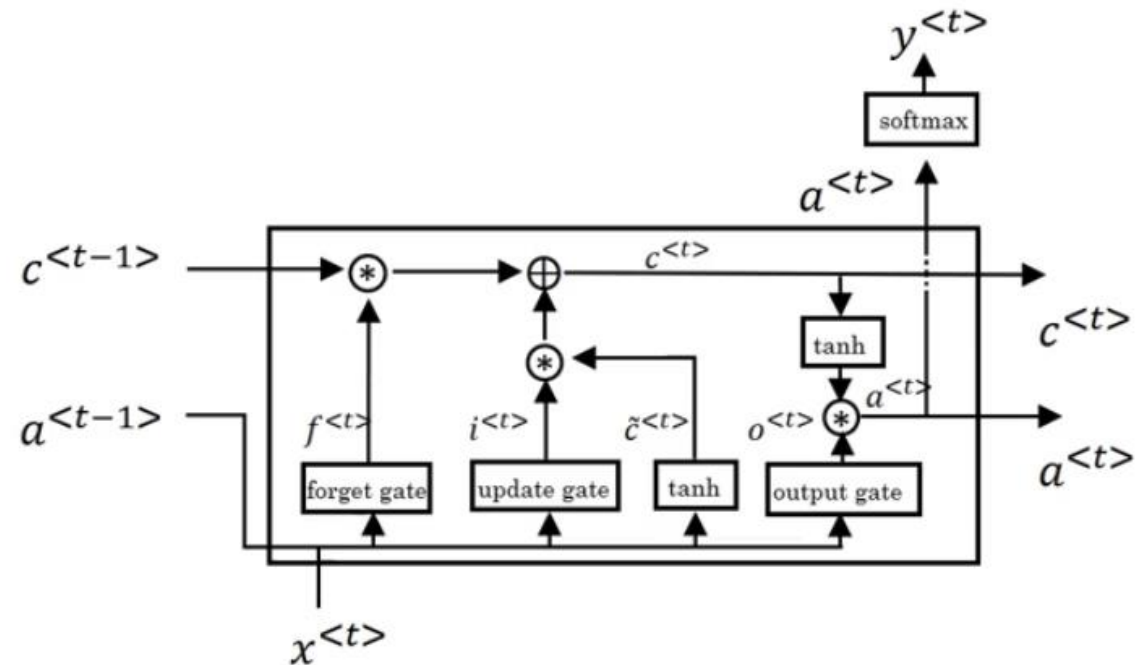
$$\Gamma_u = \sigma(W_u[c^{<t-1>}, x^{<t>}] + b_u)$$

$$\Gamma_f = \sigma(W_f[c^{<t-1>}, x^{<t>}] + b_f)$$

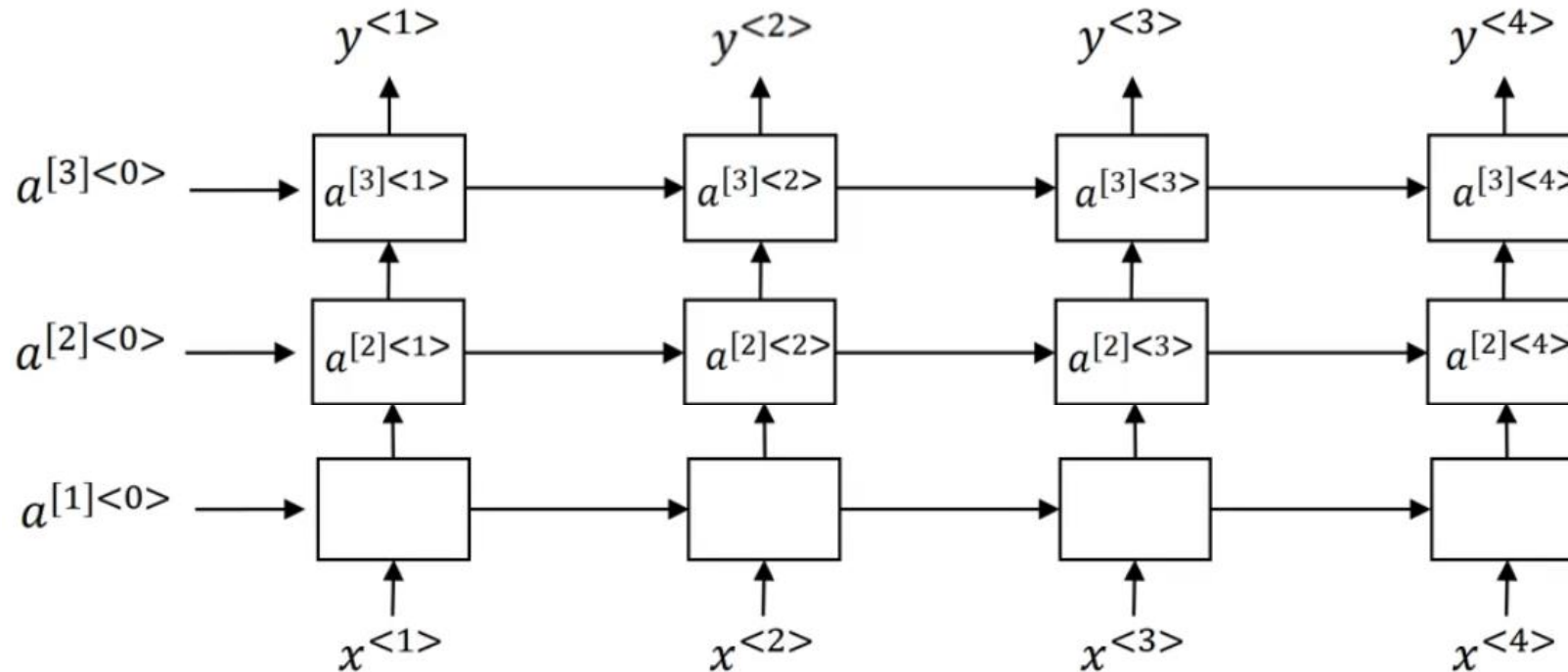
$$\Gamma_o = \sigma(W_o[c^{<t-1>}, x^{<t>}] + b_o)$$

$$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + \Gamma_f * c^{<t-1>}$$

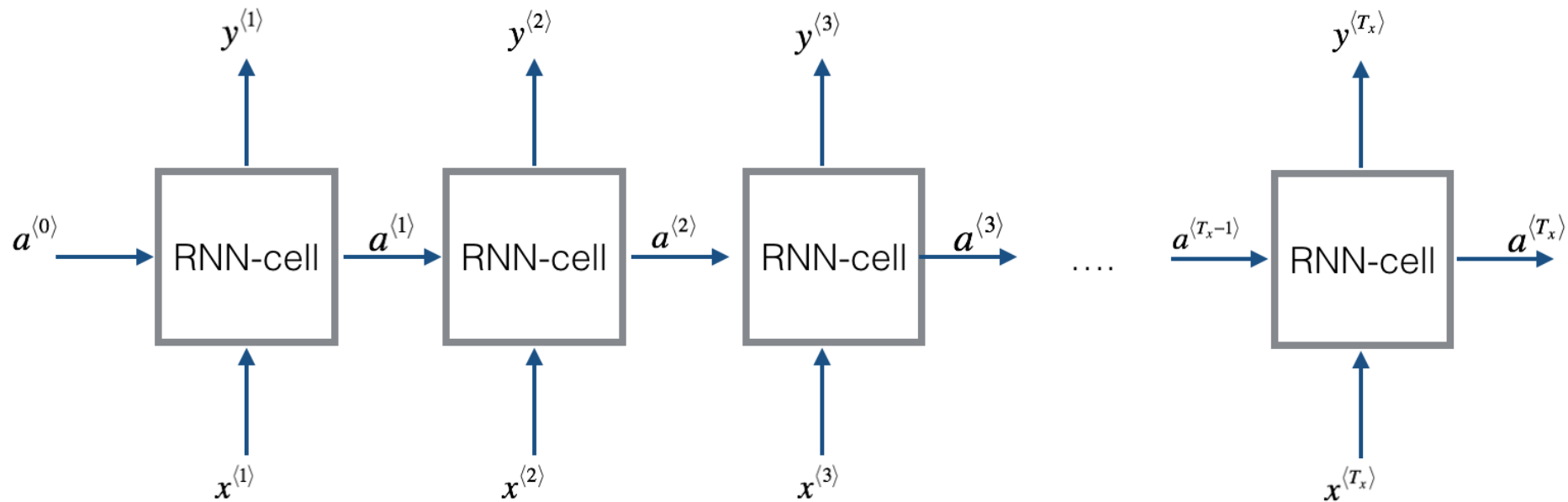
$$a^{<t>} = \Gamma_o * \tanh(c^{<t>})$$



6.5. Deep RNN



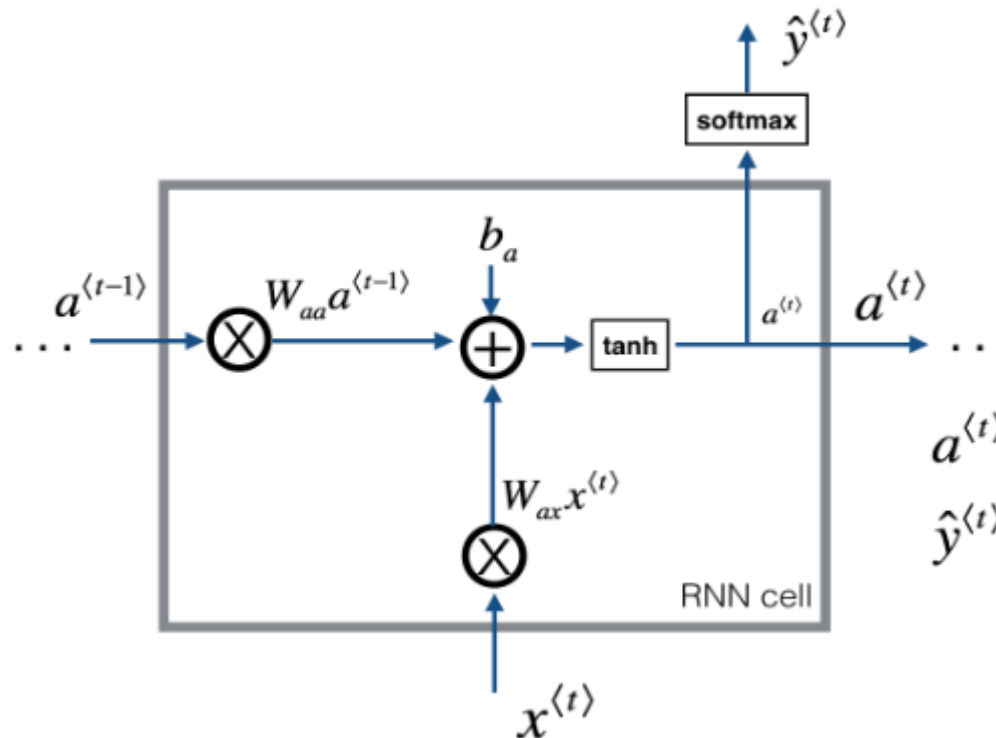
6.6. Recurrent Neural Network – Paso a Paso



1. Implementar los cálculos necesarios para una sola celda.
2. Implementar un loop sobre todos los T_x tiempos para procesar todos los inputs.

6.6. Recurrent Neural Network – Paso a Paso

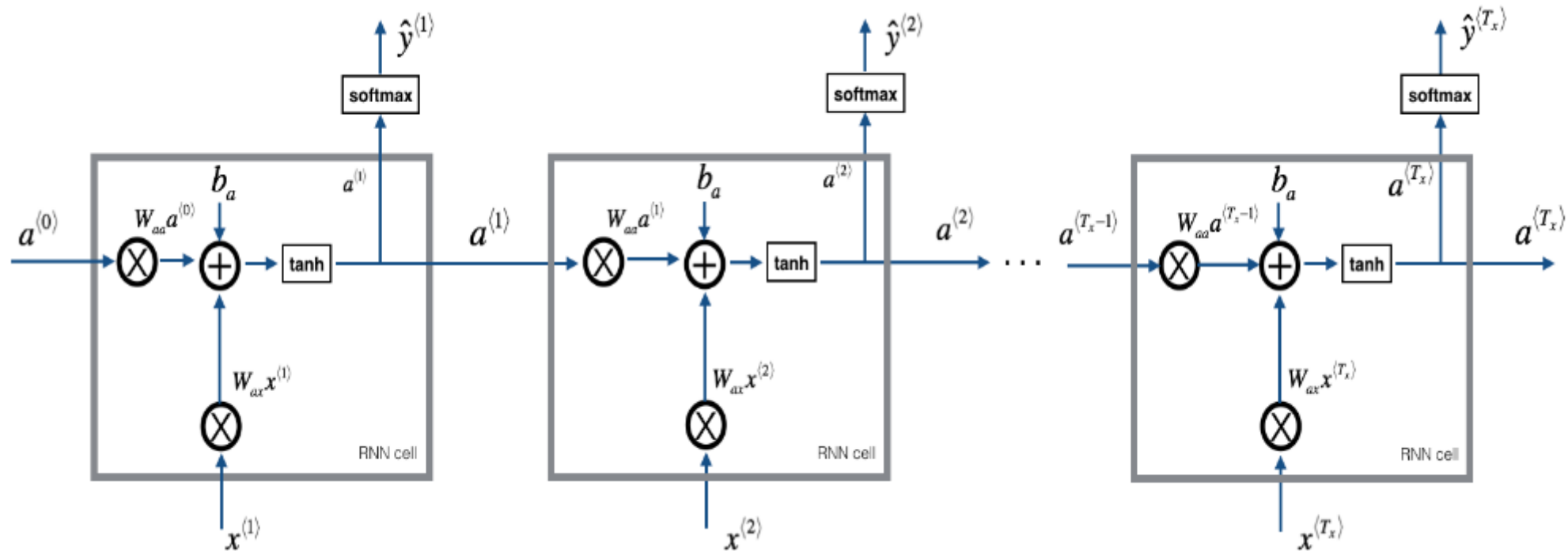
6.6.1. Celda RNN



$$a^{(t)} = \tanh(W_{ax}x^{(t)} + W_{aa}a^{(t-1)} + b_a)$$
$$\hat{y}^{(t)} = \text{softmax}(W_{ya}a^{(t)} + b_y)$$

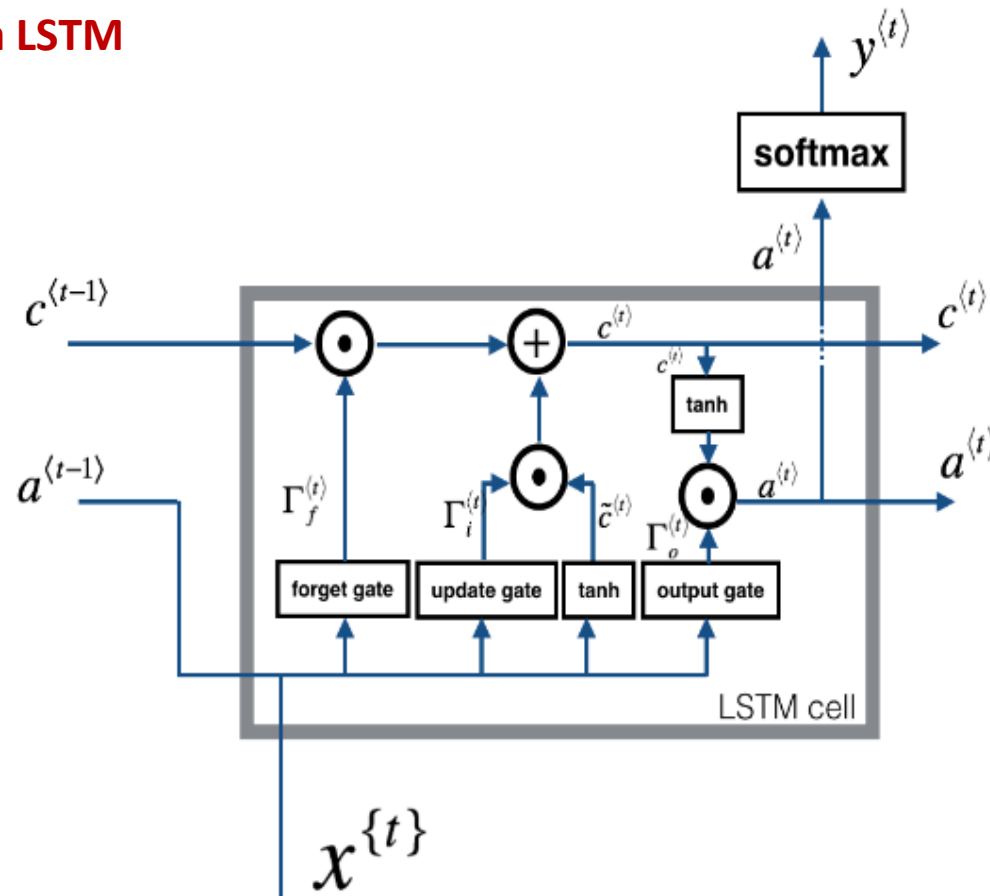
6.6. Recurrent Neural Network – Paso a Paso

6.6.2. Forward Propagation



6.6. Recurrent Neural Network – Paso a Paso

6.6.3. Celda LSTM



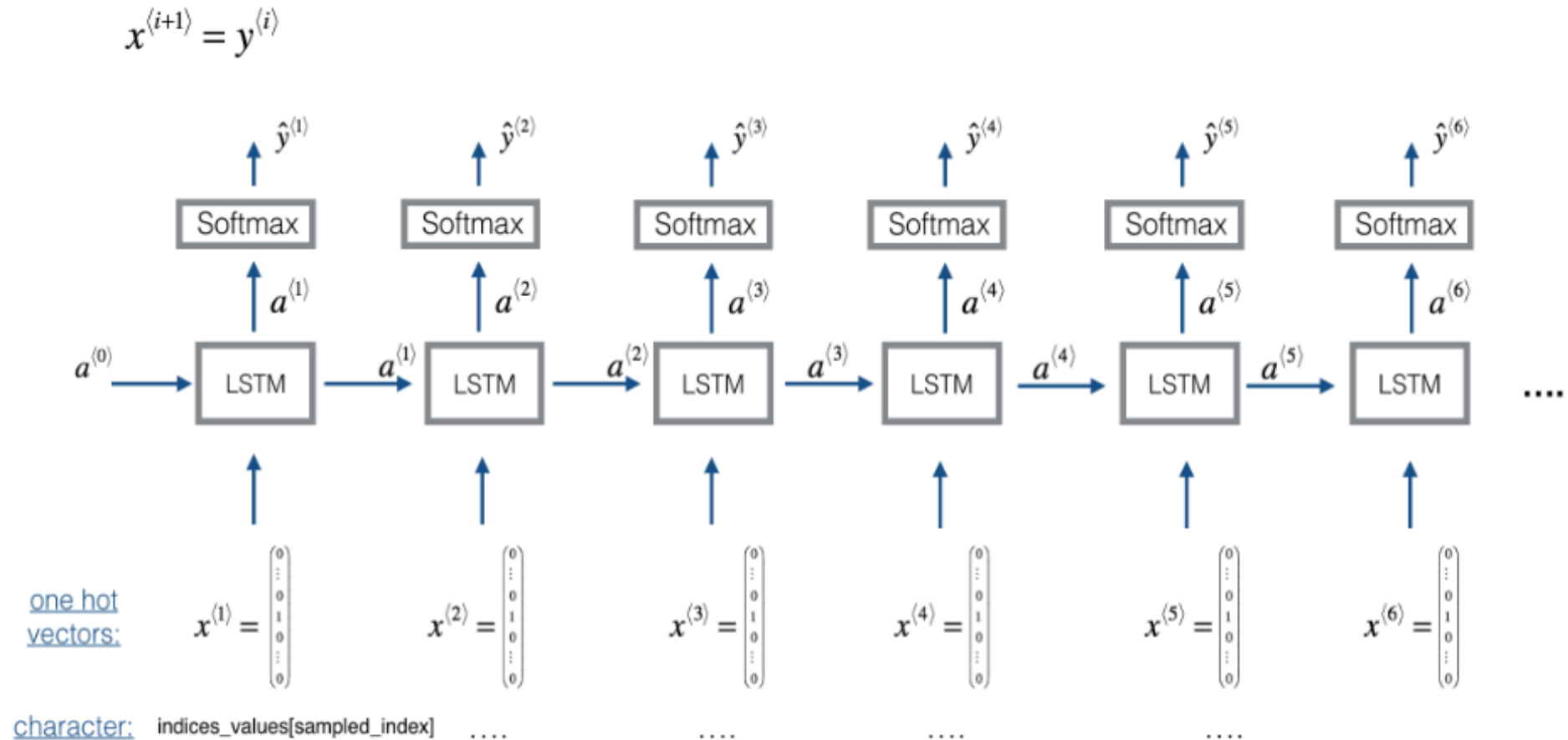
$$\begin{aligned}\Gamma_f^{\langle t \rangle} &= \sigma(W_f[a^{\langle t-1 \rangle}, x^{\langle t \rangle}] + b_f) \\ \Gamma_u^{\langle t \rangle} &= \sigma(W_u[a^{\langle t-1 \rangle}, x^{\langle t \rangle}] + b_u) \\ \tilde{c}^{\langle t \rangle} &= \tanh(W_c[a^{\langle t-1 \rangle}, x^{\langle t \rangle}] + b_c) \\ c^{\langle t \rangle} &= \Gamma_f^{\langle t \rangle} \circ c^{\langle t-1 \rangle} + \Gamma_u^{\langle t \rangle} \circ \tilde{c}^{\langle t \rangle} \\ \Gamma_o^{\langle t \rangle} &= \sigma(W_o[a^{\langle t-1 \rangle}, x^{\langle t \rangle}] + b_o) \\ a^{\langle t \rangle} &= \Gamma_o^{\langle t \rangle} \circ \tanh(c^{\langle t \rangle})\end{aligned}$$

6.7. Jazz Generation Example



<https://soundcloud.com/deepjazz-ai>

6.7. Jazz Generation Example



6.7. Jazz Generation Example

Dataset

X: Es un array de dimension (m, $T_x T_x$, 78). Tenemos “m” datos de entrenamiento examples, cada uno con una secuencia de “ $T_x=30$ ” valores musicales. En cada momento de la secuencia, el input tiene uno de los 78 diferentes posibles “valores”, representado como un vector “one-hot”. Así, por ejemplo, $X[i,t,:]$ es un vector one-hot que representa el “valor” de la muestra “i” en el tiempo “t”.

Y: En esencia es idéntico a X, pero rotado un paso a la izquierda (al pasado). Esto es porque nos interesa que la red use valores pasados para predecir el siguiente valor, así, nuestro modelo secuencial tratará de predecir $y^{<t>}$ dados $x^{<1>}$, $x^{<2>}$..., $x^{<t>}$. Sin embargo, la data de Y está reordenada para que tenga las dimensiones ($T_y, m, 78$), donde $T_y=T_x$.

6.7. Jazz Generation Example

Build the model

X: Es un array de **dimension** (m , $T \times T_x$, 78). Tenemos “ m ” **datos de entrenamiento** examples, cada uno con una **secuencia de “ $T_x=30$ ”** valores musicales. En cada momento de la secuencia, el input tiene uno de los **78 diferentes posibles “valores”**, representado como un vector “one-hot”. Así, por ejemplo, $X[i,t,:]$ es un vector one-hot que representa el “valor” de la muestra “ i ” en el tiempo “ t ”.

Y: En esencia es **idéntico a X, pero rotado un paso a la izquierda** (al pasado). Esto es porque nos interesa que la red use valores pasados para predecir el siguiente valor, así, nuestro modelo secuencial tratará de predecir $y^{<t>}$ dados $x^{<1>}$, $x^{<2>}$..., $x^{<t>}$. Sin embargo, la data de Y está reordenada para que tenga las dimensiones ($T_y, m, 78$), donde $T_y = T_x$.

6.7. Jazz Generation Example

Predicting and Sampling

