Tema 4: Arquitecturas y aplicaciones de las redes neuronales profundas

Aprendizaje
Profundo

Grado en Ingeniería y Ciencia de datos (Universidad de Oviedo)

Pablo González, Pablo Pérez
{gonzalezgpablo, pabloperez}@uniovi.es
Centro de Inteligencia Artificial, Gijón

Introducción

En la actualidad existen numerosas arquitecturas DNN creadas en función del problema que pretenden resolver.

Veremos en detalle las siguientes:

- Redes recurrentes
- Transformers
- Redes convolucionales
- Redes generativas adversarias
- Autoencoders
- Otras

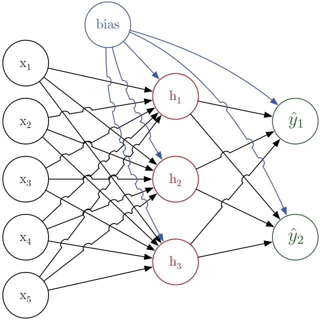
Redes neuronales recurrentes (RNN)

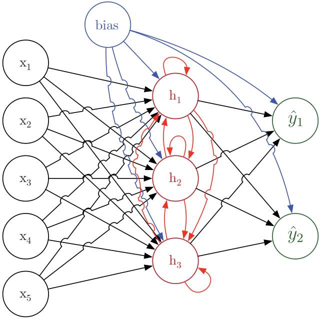
Introducción

¿Qué son?

Redes neuronales profundas que incorporan capas con conexiones recurrentes.

Las conexiones recurrentes sirven para sacar partido de lo aprendido en instantes anteriores.





Introducción

¿Para qué sirven?

Son útiles en el tratamiento de información secuencial (texto, audio, video,...).

Ejemplos:

- Reconocimiento del habla
- Análisis de sentimientos a partir de textos
- Traducción automática
- Reconocimiento de entidades
- Generación de música
- Análisis de secuencias de ADN
- etc.

RNN vs Feed Forward

Ejemplo: Predecir la palabra siguiente a partir del comienzo de una frase

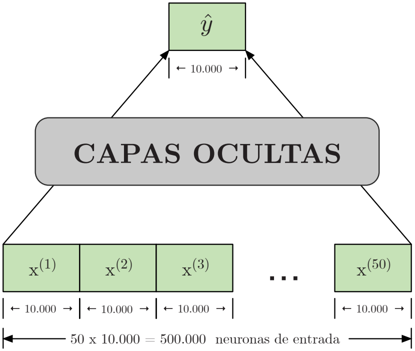
Entrada: Secuencia de palabras de longitud variable.
Salida: Palabra.

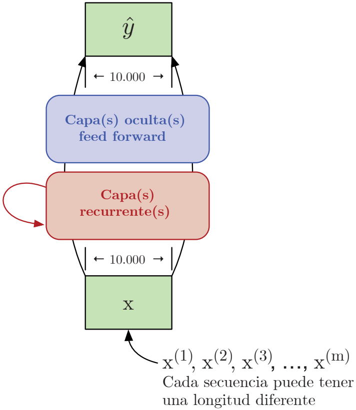
Otros datos:

- Cada palabra se codifica utilizando one-hot.
- Una frase es una secuencia con número variable de vectores one-hot.
- Supongamos:
  - Vocabulario de 10000 palabras.
  - Longitud máxima de frase de 50 palabras.

Feed Forward vs RNN

Mismo problema, arquitecturas diferentes:





Notación

Matrices de pesos:

- $\mathbf{U}$: Conexiones entre capa de entrada y recurrente.
- $\mathbf{V}$: Conexiones de la capa recurrente con la siguiente capa.
- $\mathbf{W}$: Conexiones de la capa recurrente con ella misma.
- $\mathbf{b}_h, \mathbf{b}_y$: Bias de la capa recurrente y de la de salida.

Funciones de activación:

- g_1: Suele ser Tangente Hiperbólica o ReLU.
- g_2: Sigmoide (generalmente).

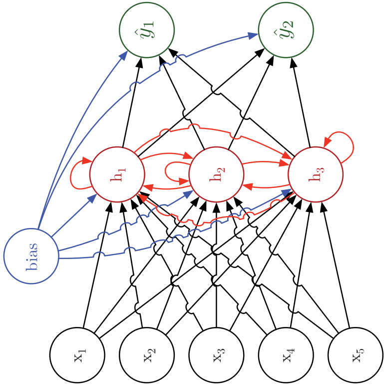
Notación

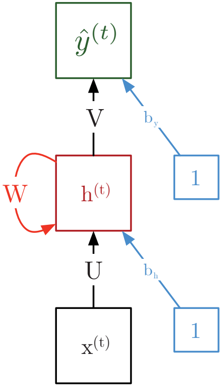
Dimensiones:

- $\mathbf{U}$: $3x5$
- $\mathbf{V}$: $2x3$
- $\mathbf{W}$: $3x3$
- $\mathbf{b}_h$: $3x1$
- $\mathbf{b}_y$: $2x1$

Ecuaciones:

- $\mathbf{h}^{(t)} = g_1(\mathbf{W}\mathbf{h}^{(t-1)} + \mathbf{U}\mathbf{x}^{(t)} + \mathbf{b}_h)$
- $\hat{\mathbf{y}}^{(t)} = g_2(\mathbf{V}\mathbf{h}^{(t)} + \mathbf{b}_y)$





Partiendo de:

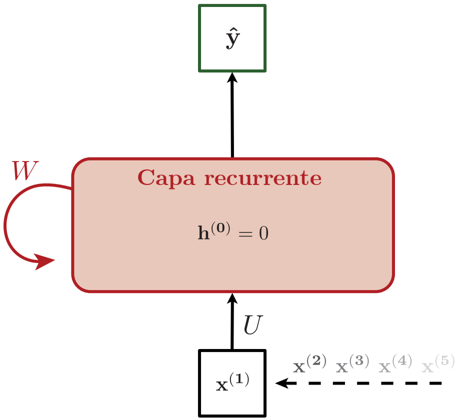
- $\mathbf{h}^{(t)} = g_1(\mathbf{W}\mathbf{h}^{(t-1)} + \mathbf{U}\mathbf{x}^{(t)} + \mathbf{b}_h)$

Para simplificar, eliminamos activación y bias:

- $\mathbf{h}^{(t)} = \mathbf{W}\mathbf{h}^{(t-1)} + \mathbf{U}\mathbf{x}^{(t)}$

Notación:

- $\mathbf{x}^{(i)}$: Vector del i-ésimo elemento de la secuencia de entrada.
- $\mathbf{h}^{(i)}$: Salida de la capa recurrente debida a la entrada $\mathbf{x}^{(i)}$, es decir, en la etapa i-ésima.



Memoria en las RNN

Partiendo de:

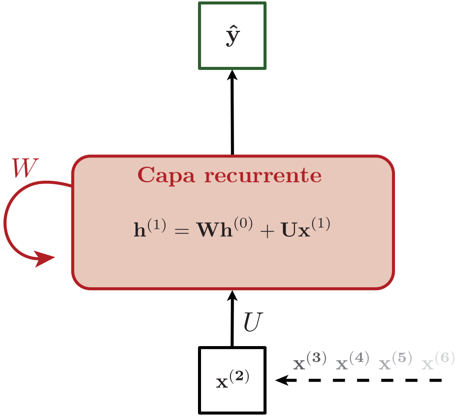
- $\mathbf{h}^{(t)} = g_1(\mathbf{W}\mathbf{h}^{(t-1)} + \mathbf{U}\mathbf{x}^{(t)} + \mathbf{b}_h)$

Para simplificar, eliminamos activación y bias:

- $\mathbf{h}^{(t)} = \mathbf{W}\mathbf{h}^{(t-1)} + \mathbf{U}\mathbf{x}^{(t)}$

Notación:

- $\mathbf{x}^{(i)}$: Vector del i-ésimo elemento de la secuencia de entrada.
- $\mathbf{h}^{(i)}$: Salida de la capa recurrente debida a la entrada $\mathbf{x}^{(i)}$, es decir, en la etapa i-ésima.



Partiendo de:

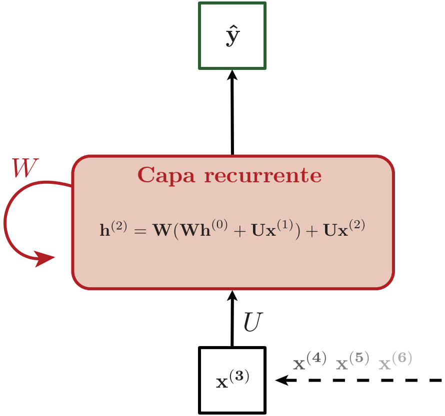
- $\mathbf{h}^{(t)} = g_1(\mathbf{W}\mathbf{h}^{(t-1)} + \mathbf{U}\mathbf{x}^{(t)} + \mathbf{b}_h)$

Para simplificar, eliminamos activación y bias:

- $\mathbf{h}^{(t)} = \mathbf{W}\mathbf{h}^{(t-1)} + \mathbf{U}\mathbf{x}^{(t)}$

Notación:

- $\mathbf{x}^{(i)}$: Vector del i-ésimo elemento de la secuencia de entrada.
- $\mathbf{h}^{(i)}$: Salida de la capa recurrente debida a la entrada $\mathbf{x}^{(i)}$, es decir, en la etapa i-ésima.



Memoria en las RNN

Partiendo de:

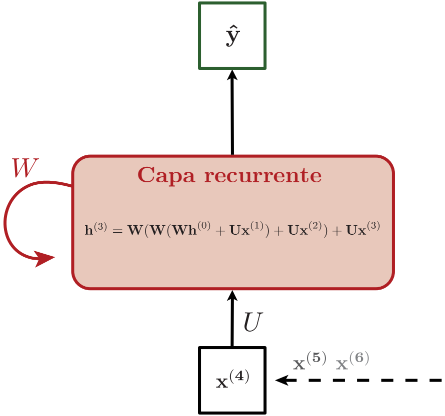
- $\mathbf{h}^{(t)} = g_1(\mathbf{W}\mathbf{h}^{(t-1)} + \mathbf{U}\mathbf{x}^{(t)} + \mathbf{b}_h)$

Para simplificar, eliminamos activación y bias:

- $\mathbf{h}^{(t)} = \mathbf{W}\mathbf{h}^{(t-1)} + \mathbf{U}\mathbf{x}^{(t)}$

Notación:

- $\mathbf{x}^{(i)}$: Vector del i-ésimo elemento de la secuencia de entrada.
- $\mathbf{h}^{(i)}$: Salida de la capa recurrente debida a la entrada $\mathbf{x}^{(i)}$, es decir, en la etapa i-ésima.



Memoria en las RNN

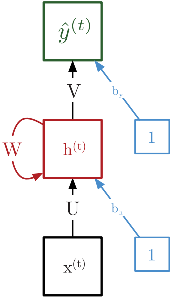
$$\mathbf{h}^{(1)} = g_1\left(\mathbf{W}\mathbf{h}^{(0)} + \mathbf{U}\mathbf{x}^{(1)} + \mathbf{b}_h\right)$$

$$\mathbf{h}^{(2)} = g_1\left(\mathbf{W}\left(g_1\left(\mathbf{W}\mathbf{h}^{(0)} + \mathbf{U}\mathbf{x}^{(1)} + \mathbf{b}_h\right)\right) + \mathbf{U}\mathbf{x}^{(2)} + \mathbf{b}_h\right)$$

$$\mathbf{h}^{(3)} = g_1\left(\mathbf{W}\left(g_1\left(\mathbf{W}\left(g_1\left(\mathbf{W}\mathbf{h}^{(0)} + \mathbf{U}\mathbf{x}^{(1)} + \mathbf{b}_h\right)\right) + \mathbf{U}\mathbf{x}^{(2)} + \mathbf{b}_h\right)\right) + \mathbf{U}\mathbf{x}^{(3)} + \mathbf{b}_h\right)$$

$$\vdots$$

$$\mathbf{h}^{(t)} = g_1\left(\mathbf{W}\mathbf{h}^{(t-1)} + \mathbf{U}\mathbf{x}^{(t)} + \mathbf{b}_h\right) \Leftarrow \text{Salida de la capa recurrente.}$$



$$\mathbf{y}^{(1)} = g_2\left(\mathbf{V}\mathbf{h}^{(1)} + \mathbf{b}_y\right) = g_2\left(\mathbf{V}\cdot g_1\left(\mathbf{W}\mathbf{h}^{(0)} + \mathbf{U}\mathbf{x}^{(1)} + \mathbf{b}_h\right) + \mathbf{b}_y\right)$$

$$\mathbf{y}^{(2)} = g_2\left(\mathbf{V}\cdot g_1\left(\mathbf{W}\left(g_1\left(\mathbf{W}\mathbf{h}^{(0)} + \mathbf{U}\mathbf{x}^{(1)} + \mathbf{b}_h\right)\right) + \mathbf{U}\mathbf{x}^{(2)} + \mathbf{b}_h\right) + \mathbf{b}_y\right)$$

$$\mathbf{y}^{(3)} = g_2\left(\mathbf{V}\cdot g_1\left(\mathbf{W}\left(g_1\left(\mathbf{W}\left(g_1\left(\mathbf{W}\mathbf{h}^{(0)} + \mathbf{U}\mathbf{x}^{(1)} + \mathbf{b}_h\right)\right) + \mathbf{U}\mathbf{x}^{(2)} + \mathbf{b}_h\right)\right) + \mathbf{U}\mathbf{x}^{(3)} + \mathbf{b}_h\right) + \mathbf{b}_y\right)$$

$$\vdots$$

$$\mathbf{y}^{(t)} = g_2\left(\mathbf{V}\mathbf{h}^{(t)} + \mathbf{b}_y\right) \Leftarrow \text{Salida de la red.}$$

Memoria en las RNN

$$\mathbf{h}^{(1)} = g_1\left(\mathbf{W}\mathbf{h}^{(0)} + \mathbf{U}\mathbf{x}^{(1)} + \mathbf{b}_h\right)$$

$$\mathbf{h}^{(2)} = g_1\left(\mathbf{W}\left(g_1\left(\mathbf{W}\mathbf{h}^{(0)} + \mathbf{U}\mathbf{x}^{(1)} + \mathbf{b}_h\right)\right) + \mathbf{U}\mathbf{x}^{(2)} + \mathbf{b}_h\right)$$

$$\vdots$$

$$\mathbf{h}^{(t)} = g_1\left(\mathbf{W}\mathbf{h}^{(t-1)} + \mathbf{U}\mathbf{x}^{(t)} + \mathbf{b}_h\right)$$

$$\mathbf{M} = \left(\begin{array}{ccc|ccc} w_{1,1} & \cdots & w_{1,n_h} & u_{1,1} & \cdots & u_{1,n_i} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ w_{n_h,1} & \cdots & w_{n_h,n_h} & u_{n_h,1} & \cdots & u_{n_h,n_i} \end{array}\right)$$

$$\mathbf{h}^{(t)} = g_1\left(\mathbf{M}\left[\mathbf{h}^{(t-1)}, \mathbf{x}^{(t)}\right] + \mathbf{b}_h\right) \Leftarrow \text{Salida de la capa recurrente.}$$

$$\mathbf{y}^{(1)} = g_2\left(\mathbf{V}\mathbf{h}^{(1)} + \mathbf{b}_y\right) = g_2\left(\mathbf{V} \cdot g_1\left(\mathbf{W}\mathbf{h}^{(0)} + \mathbf{U}\mathbf{x}^{(1)} + \mathbf{b}_h\right) + \mathbf{b}_y\right)$$

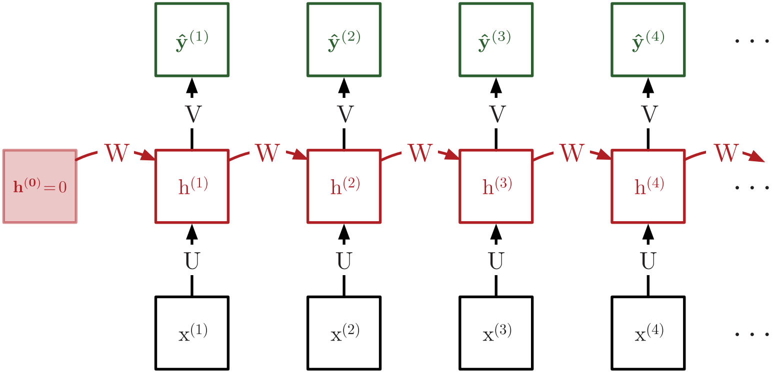
$$\mathbf{y}^{(2)} = g_2\left(\mathbf{V} \cdot g_1\left(\mathbf{W}\left(g_1\left(\mathbf{W}\mathbf{h}^{(0)} + \mathbf{U}\mathbf{x}^{(1)} + \mathbf{b}_h\right)\right) + \mathbf{U}\mathbf{x}^{(2)} + \mathbf{b}_h\right) + \mathbf{b}_y\right)$$

$$\vdots$$

$$\mathbf{y}^{(t)} = g_2\left(\mathbf{V}\mathbf{h}^{(t)} + \mathbf{b}_y\right) \Leftarrow \text{Salida de la red.}$$

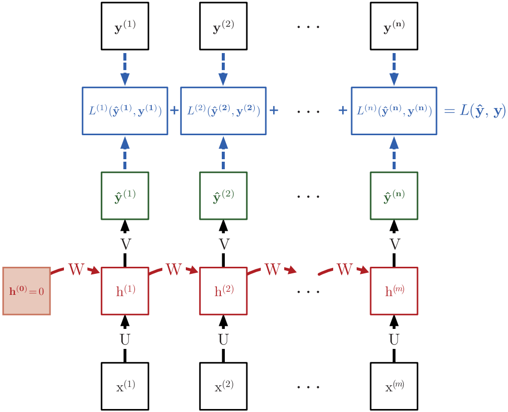
Despliegue en una RNN

Las mismas matrices **U**, **V** y **W** se utilizan en distintas etapas de la secuencia de entrada:



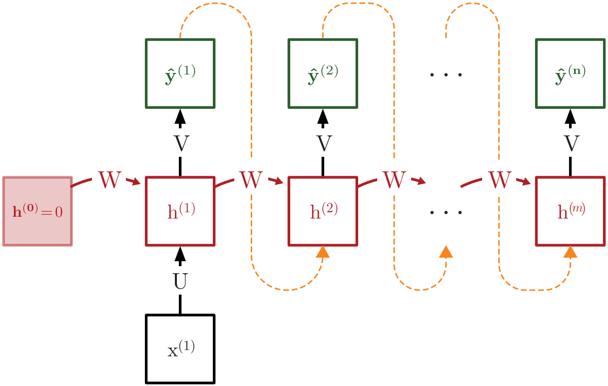
Backpropagation through time (BPTT):

1. Se presenta una secuencia de etapas con su correspondiente entrada y salida deseada a la red.
2. Se desarrolla la red y se aplica la propagación hacia delante (etapa a etapa), luego se calculan y acumulan los errores de cada etapa.
3. Se actualizan los pesos en el sentido contrario al desarrollo de la red (como si se enrollase la red de nuevo).



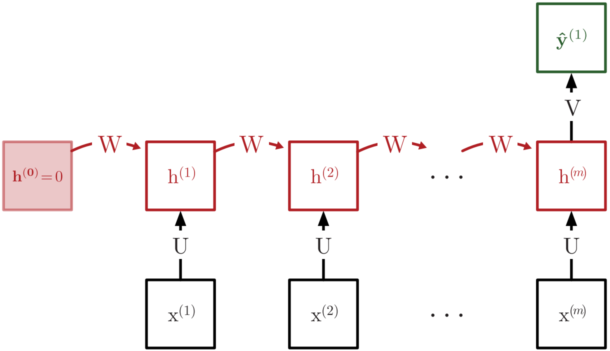
Arquitecturas de ejemplo: 1 a N

Generación automática de música: A partir de una nota de partida se genera una pieza musical, es decir, una secuencia más o menos larga de notas.

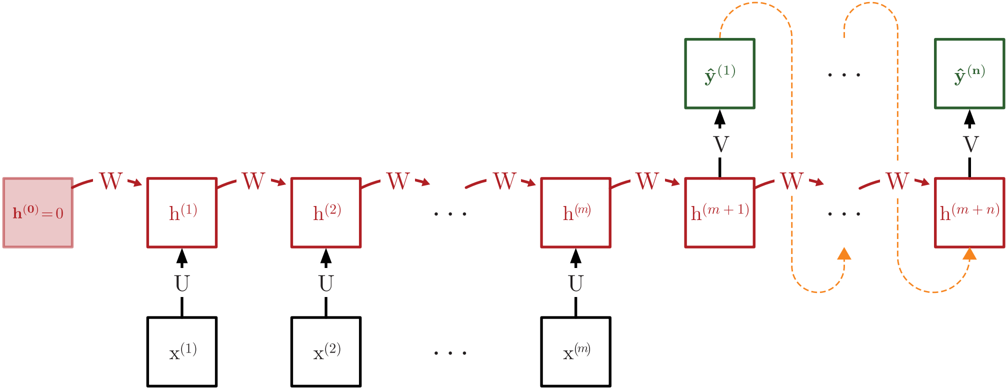


Arquitecturas de ejemplo: M a 1

Análisis de sentimientos: A partir de un texto, predecir si la opinión que refleja es favorable (positiva) o desfavorable (negativa) con respecto a un ítem que se está evaluando.



Traducción automática: A partir de un texto en un idioma, generar la versión traducida a otro idioma.



Exploding/Vanishing Gradients

En redes profundas, debido al gran número de productos encadenados, puede producirse los ya descritos:

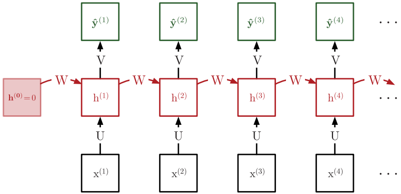
- **Exploding gradients:** Crecimiento de gradientes tal que las oscilaciones durante el descenso de gradiente provocan que la red no converja.
- **Vanishing gradients:** Los gradientes se hacen tan pequeños que el descenso apenas es apreciable y la red no aprende.

¿Puede ser un problema en una red recurrente con una sola capa? $\rightarrow$ **SI**

Exploding/Vanishing Gradients

Una secuencia de n elementos hace que la red se comporte como si tuviese n capas

$$\mathbf{h}^{(10)} = g_1\left(\mathbf{M}\left[g_1\left(\mathbf{M}\left[g_1\left(\mathbf{M}\left[g_1\left(\mathbf{M}\left[g_1\left(\mathbf{M}\left[g_1\left(\mathbf{M}\left[g_1\left(\mathbf{M}\left[g_1\left(\mathbf{M}\left[g_1\left(\mathbf{M}\left[g_1\left(\mathbf{M}\left[h^{(0)},\mathbf{x}^{(1)}\right]+\mathbf{b}_h\right),\mathbf{x}^{(2)}\right]+\mathbf{b}_h\right),\mathbf{x}^{(3)}\right]+\mathbf{b}_h\right),\mathbf{x}^{(4)}\right]+\mathbf{b}_h\right),\mathbf{x}^{(5)}\right]+\mathbf{b}_h\right),\mathbf{x}^{(6)}\right]+\mathbf{b}_h\right),\mathbf{x}^{(7)}\right]+\mathbf{b}_h\right),\mathbf{x}^{(8)}\right]+\mathbf{b}_h\right),\mathbf{x}^{(9)}\right]+\mathbf{b}_h\right),\mathbf{x}^{(10)}\right]+\mathbf{b}_h\right)$$



Solución:

- **Exploding:** Estrategias de clipping.
- **Vanishing:** Diseño de nuevos tipos de red:
  - Redes GRU.
  - Redes LSTM.

Vanishing Gradients

- En redes **feed forward** este efecto provoca variaciones mínimas en las primeras capas.
- En las **redes recurrentes** implica que la influencia de los elementos de la secuencia de entrada se concentrará en el final de la secuencia.
  - *"La bicicleta, siendo la alternativa de transporte urbano más ecológica, es poco usada"*
  - *"Las bicicletas, siendo la alternativa de transporte urbano más ecológica, son poco usadas"*

Sería bueno que la red 'recordase' que estamos hablando en singular o en plural.

Redes neuronales recurrentes (RNN)

Gate Recurrent Unit (GRU)

Gate Recurrent Unit (GRU)

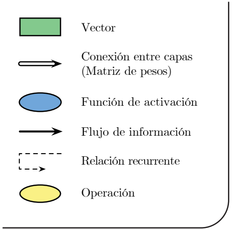
Definición

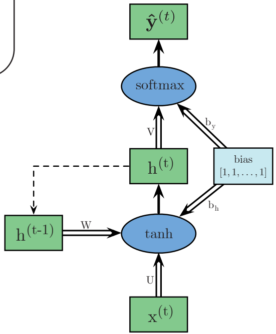
Redes recurrentes con 'gates' (puertas) que regulan el flujo de información que pasa de una etapa a la siguiente.

- **Regulación:** Transmitir desde cada neurona una salida ponderada entre su activación y la entrada recurrente.
    - El factor de ponderación se aprende y suele ser en muchos casos próximo a 0 o a 1.
- **Efecto memoria:** Puede ocurrir que entre la etapa j y la etapa k fluya la información de la etapa anterior, es decir, $\mathbf{h_i}^{(j)} \simeq \mathbf{h_i}^{(j+1)} \simeq \ldots \simeq \mathbf{h_i}^{(k-2)} \simeq \mathbf{h_i}^{(k-1)}$. Esto provoca que la salida en la etapa k dependa en gran medida de la etapa j, que puede ser muy anterior.

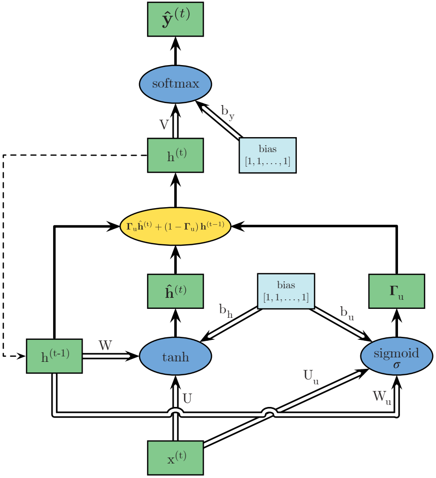
Intuitivamente, una red GRU puede permitir que las neuronas se especialicen en determinadas propiedades de la secuencia que están tratando para mejorar la predicción.

RNN vs GRU



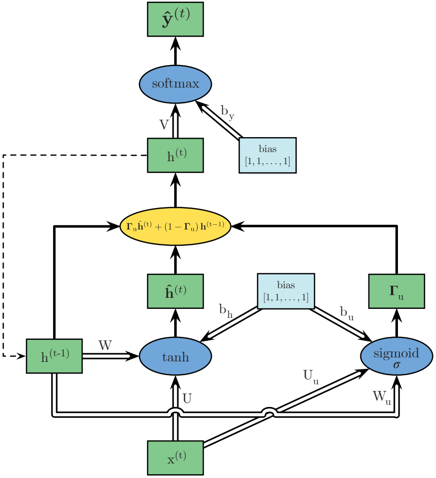


RNN simple

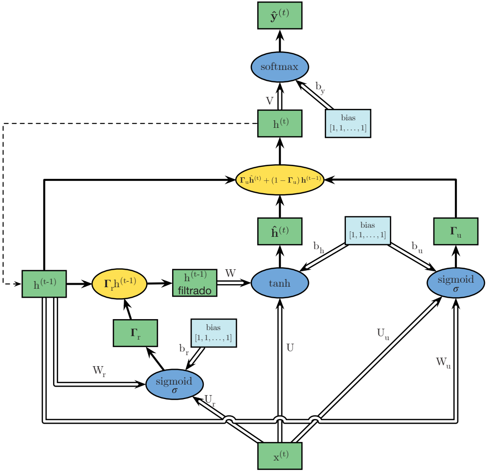


GRU

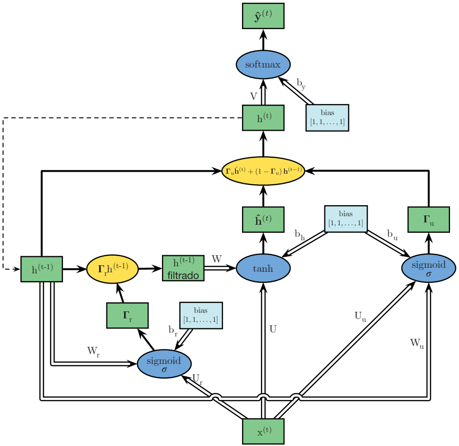
GRU con relevancia



GRU



GRU con puerta de relevancia



GRU con puerta de relevancia

$$\mathbf{\Gamma}_{\mathrm{u}} = \sigma(\mathbf{M}_u[\mathbf{h}^{(t-1),\mathbf{x}^{(t)}}] + \mathbf{b}_u)$$

$$\mathbf{\Gamma}_{\mathrm{r}} = \sigma(\mathbf{M}_r[\mathbf{h}^{(t-1),\mathbf{x}^{(t)}}] + \mathbf{b}_r)$$

$$\hat{\boldsymbol{h}}^{(t)} = \tanh(\mathbf{M}[\mathbf{\Gamma}_{\mathrm{r}} \odot \mathbf{h}^{(t-1)}, \mathbf{x}^{(t)}] + \mathbf{b}_h)$$

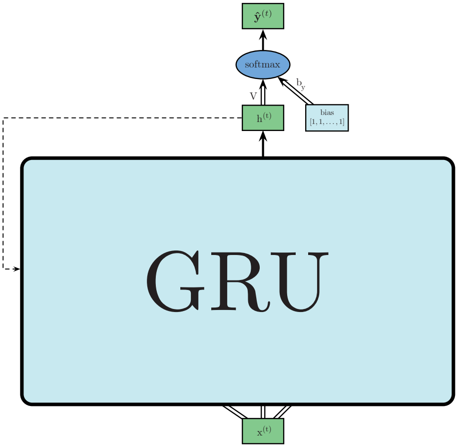
$$\boldsymbol{h}^{(t)} = \mathbf{\Gamma}_{\mathrm{r}} \odot \hat{\mathbf{h}}^{(t)} + (1 - \mathbf{\Gamma}_{\mathrm{u}}) \odot \mathbf{h}^{(t-1)}$$

donde:

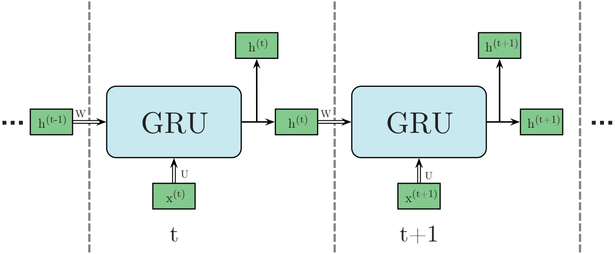
$$\mathbf{M} = [\mathbf{W}, \mathbf{U}]$$

$$\mathbf{M}_r = [\mathbf{W}_r, \mathbf{U}_r]$$

$$\mathbf{M}_u = [\mathbf{W}_u, \mathbf{U}_u]$$



GRU con puerta de relevancia



GRU con puerta de relevancia desplegado

Redes neuronales recurrentes (RNN)

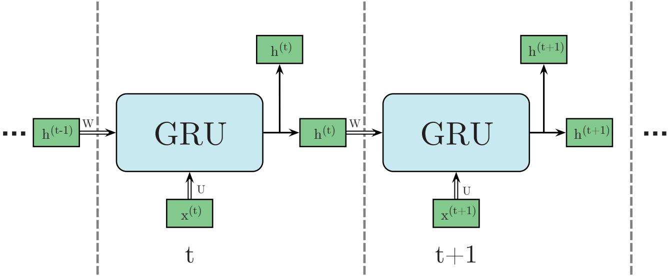
Long Short Term Memory (LSTM)

Descripción

Se presentaron antes que las GRU (Hochreiter y Schmidhuber, 1997). Más complejas y flexibles que las redes GRU ya que cuentan con varias 'gates'.

En las redes LSTM hay dos vectores de información que se pasan de una etapa a otra:

- $\hat{\mathbf{h}}^{(t)}$: Salida de la capa LSTM.
- $\hat{\mathbf{c}}^{(t)}$: *Estado* de la capa LSTM.



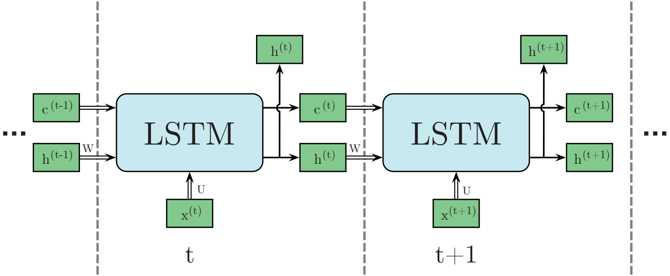
Long Short Term Memory (LSTM)

Descripción

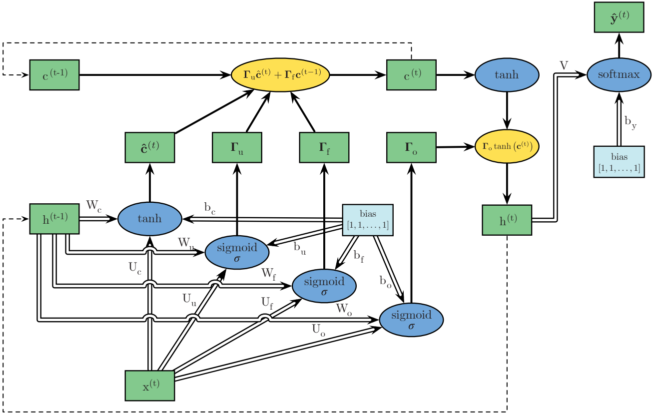
Se presentaron antes que las GRU (Hochreiter y Schmidhuber, 1997). Más complejas y flexibles que las redes GRU ya que cuentan con varias 'gates'.

En las redes LSTM hay dos vectores de información que se pasan de una etapa a otra:

- $\hat{\mathbf{h}}^{(t)}$: Salida de la capa LSTM.
- $\hat{\mathbf{c}}^{(t)}$: *Estado* de la capa LSTM.



Notación

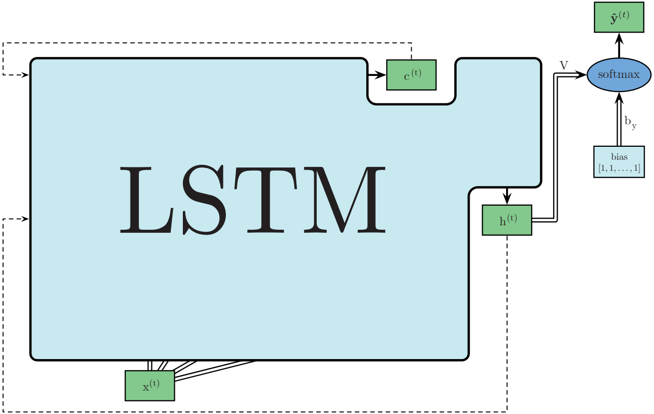


$$\Gamma_u = \sigma(\mathbf{M}_u[\mathbf{h}^{(t-1)}, \mathbf{x}^{(t)}] + \mathbf{b}_u)$$
$$\Gamma_f = \sigma(\mathbf{M}_f[\mathbf{h}^{(t-1)}, \mathbf{x}^{(t)}] + \mathbf{b}_f)$$
$$\Gamma_o = \sigma(\mathbf{M}_o[\mathbf{h}^{(t-1)}, \mathbf{x}^{(t)}] + \mathbf{b}_o)$$
$$\hat{\boldsymbol{c}}^{(t)} = \tanh(\mathbf{M}_c[\mathbf{h}^{(t-1)}, \mathbf{x}^{(t)}] + \mathbf{b}_c)$$
$$\boldsymbol{c}^{(t)} = \Gamma_u \odot \hat{\boldsymbol{c}}^{(t)} + \Gamma_f \odot \boldsymbol{c}^{(t-1)}$$
$$\boldsymbol{h}^{(t)} = \Gamma_o \odot \tanh(\boldsymbol{c}^{(t)})$$

donde:

$$\mathbf{M}_u = [\mathbf{W}_u, \mathbf{U}_u] \quad \mathbf{M}_f = [\mathbf{W}_f, \mathbf{U}_f]$$
$$\mathbf{M}_o = [\mathbf{W}_o, \mathbf{U}_o] \quad \mathbf{M}_c = [\mathbf{W}_c, \mathbf{U}_c]$$

Notación



$$\boldsymbol{\Gamma}_{\mathrm{u}} = \sigma(\mathbf{M}_u[\mathbf{h}^{(t-1)}, \mathbf{x}^{(t)}] + \mathbf{b}_u)$$
$$\boldsymbol{\Gamma}_{\mathrm{f}} = \sigma(\mathbf{M}_f[\mathbf{h}^{(t-1)}, \mathbf{x}^{(t)}] + \mathbf{b}_f)$$
$$\boldsymbol{\Gamma}_{\mathrm{o}} = \sigma(\mathbf{M}_o[\mathbf{h}^{(t-1)}, \mathbf{x}^{(t)}] + \mathbf{b}_o)$$
$$\hat{\boldsymbol{c}}^{(t)} = \tanh(\mathbf{M}_c[\mathbf{h}^{(t-1)}, \mathbf{x}^{(t)}] + \mathbf{b}_c)$$
$$\boldsymbol{c}^{(t)} = \boldsymbol{\Gamma}_{\mathrm{u}} \odot \hat{\boldsymbol{c}}^{(t)} + \boldsymbol{\Gamma}_{\mathrm{f}} \odot \boldsymbol{c}^{(t-1)}$$
$$\boldsymbol{h}^{(t)} = \boldsymbol{\Gamma}_{\mathrm{o}} \odot \tanh(\boldsymbol{c}^{(t)})$$

donde:

$$\mathbf{M}_u = [\mathbf{W}_u, \mathbf{U}_u] \quad \mathbf{M}_f = [\mathbf{W}_f, \mathbf{U}_f]$$
$$\mathbf{M}_o = [\mathbf{W}_o, \mathbf{U}_o] \quad \mathbf{M}_c = [\mathbf{W}_c, \mathbf{U}_c]$$

Variantes

LSTM con peepholes:

- El cálculo de los valores de las gates también depende del estado de la capa recurrente, **c**.
- Esta variante permite aprender con mayor precisión a partir de secuencias con diferencias temporales muy sutiles.

LSTM:

$$\boldsymbol{\Gamma}_{\mathrm{u}} = \sigma(\mathbf{M}_u[\mathbf{h}^{(t-1)}, \mathbf{x}^{(t)}] + \mathbf{b}_u)$$
$$\boldsymbol{\Gamma}_{\mathrm{f}} = \sigma(\mathbf{M}_f[\mathbf{h}^{(t-1)}, \mathbf{x}^{(t)}] + \mathbf{b}_f)$$
$$\boldsymbol{\Gamma}_{\mathrm{o}} = \sigma(\mathbf{M}_o[\mathbf{h}^{(t-1)}, \mathbf{x}^{(t)}] + \mathbf{b}_o)$$

LSTM con peepholes:

$$\boldsymbol{\Gamma}_{\mathrm{u}} = \sigma(\mathbf{M}_u[\mathbf{c}^{(t-1)}, \mathbf{h}^{(t-1)}, \mathbf{x}^{(t)}] + \mathbf{b}_u)$$
$$\boldsymbol{\Gamma}_{\mathrm{f}} = \sigma(\mathbf{M}_f[\mathbf{c}^{(t-1)}, \mathbf{h}^{(t-1)}, \mathbf{x}^{(t)}] + \mathbf{b}_f)$$
$$\boldsymbol{\Gamma}_{\mathrm{o}} = \sigma(\mathbf{M}_o[\mathbf{c}^{(t)}, \mathbf{h}^{(t-1)}, \mathbf{x}^{(t)}] + \mathbf{b}_o)$$

Redes neuronales recurrentes (RNN)

Bidireccionales

RNN Bidireccionales

Descripción

Las redes bidireccionales (Schuster y Paliwal, 1997) tienen en cuenta el contexto **posterior** a un elemento en una secuencia.

El contexto puede ser relevante:

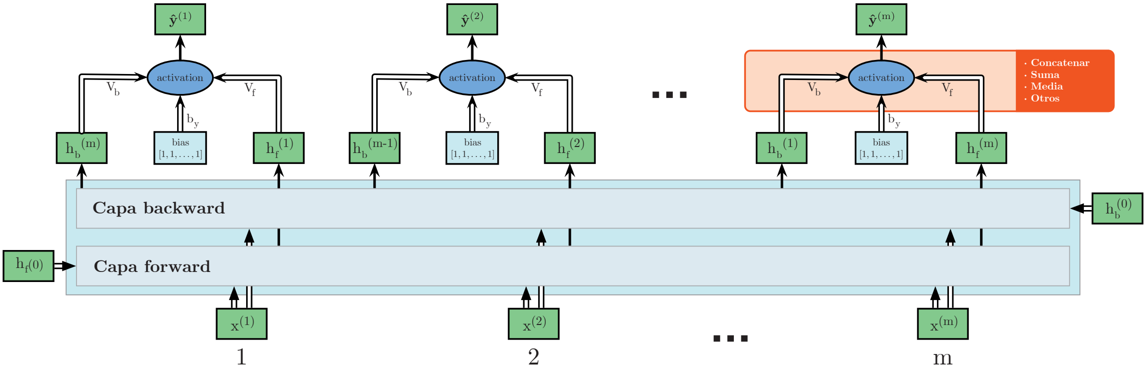
- *"Deberías ver Roma, es una ciudad con gran cantidad de monumentos"*
- *"Deberías ver Roma, es una película de Alfonso Cuarón que te va a encantar"*

En este caso es necesario utilizar elementos posteriores a la palabra Roma para determinar de que estamos hablando.

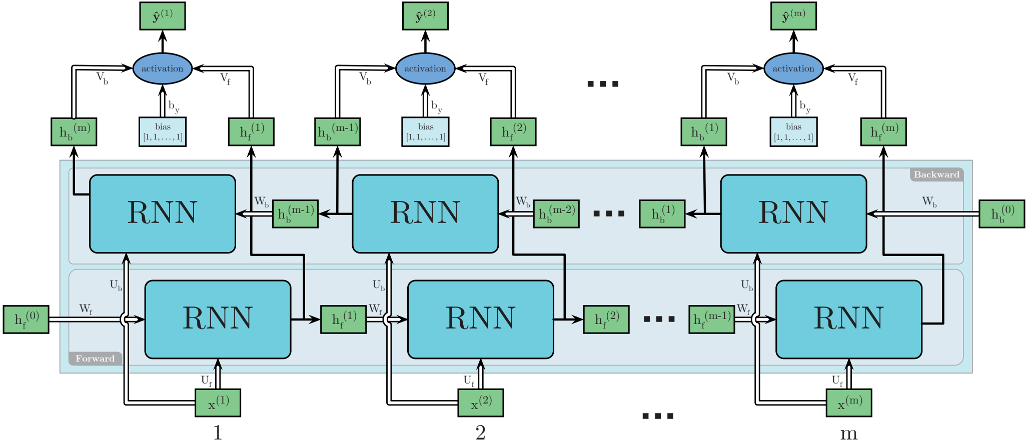
RNN Bidireccional (BRNN)

Constituidas por:

- **Capa forward**: Celdas RNN que procesan la secuencia de inicio a fin.
- **Capa backward**: Celdas que la procesan en orden inverso.



Capas forward y backward en detalle:



Referencias

Referencias

1. **Redes recurrentes (GRU, LSTM y Bidireccionales), Oscar Luaces**