

1^a
Emisión

DATA SCIENCE

Módulo 8

Introducción al Deep Learning

Redes recurrentes

Instructor: Gibran Fuentes Pineda



UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

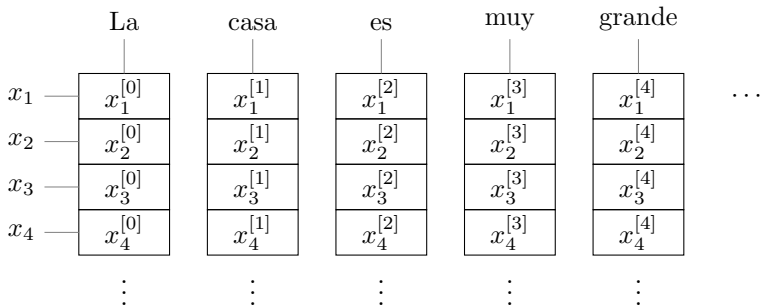
Dirección General de Cómputo y de Tecnologías de información y Comunicación

Dirección de Docencia en TIC



**Educación
Continua**
1971 - 2021

Motivación: secuencias de palabras



Objetivo del tema

- ▶ Conocer los componentes básicos y comprender el funcionamiento de las redes recurrentes, así como las estrategias para entrenarlas.

Sub-temas

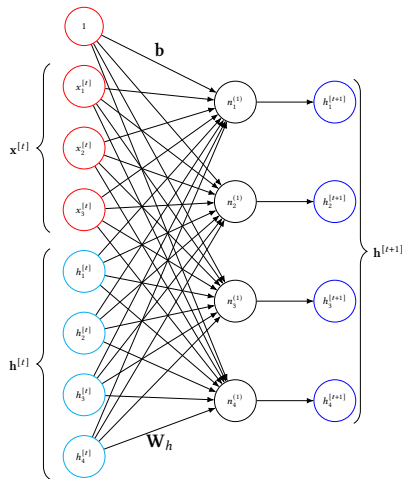
- 7.1 Unidad recurrente básica
- 7.2 Long-short term memory (LSTM)
- 7.3 Gated recurrent unit (GRU)
- 7.4 Arquitecturas de redes recurrentes
- 7.5 Algoritmo de retropropagación de errores para redes recurrentes
- 7.6 Ejemplo práctico: series de tiempo

Unidad recurrente básica

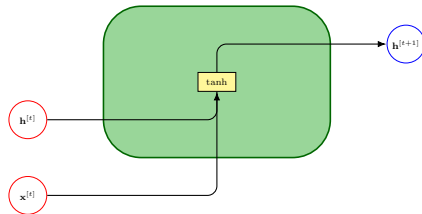
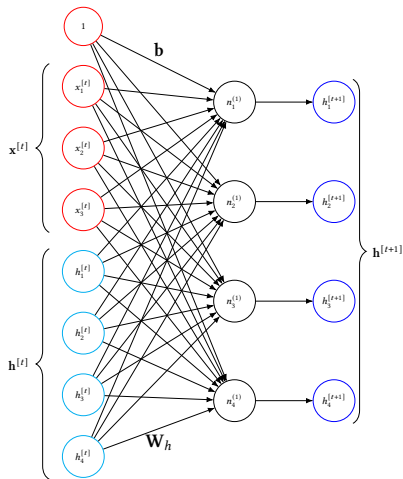
- Capas con retro-alimentación en sus conexiones

1. Entradas en tiempo t ($\mathbf{x}^{[t]}$)
2. Estado en tiempo t ($\mathbf{h}^{[t]}$)

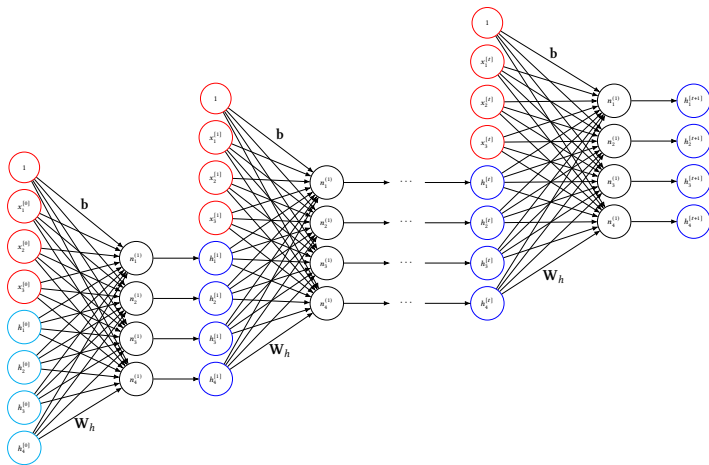
$$\begin{aligned}\mathbf{h}^{[t+1]} &= \phi \left(\mathbf{W}_h \cdot \underbrace{\begin{bmatrix} \mathbf{h}^{[t]} \\ \mathbf{x}^{[t]} \end{bmatrix}}_{\text{Concatenación}} + \mathbf{b}_h \right) \\ &= \phi \left(\mathbf{W}_{hh} \cdot \mathbf{h}^{[t]} + \mathbf{W}_{hx} \cdot \mathbf{x}^{[t]} + \mathbf{b}_h \right)\end{aligned}$$



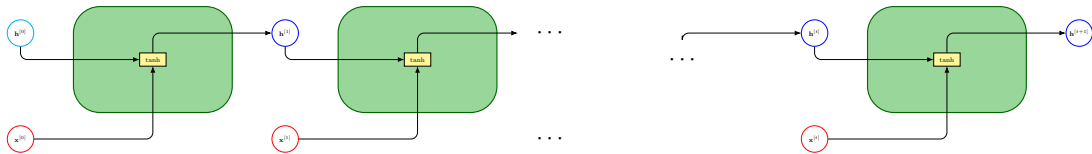
Unidad recurrente básica: diagrama de celda



Unidad recurrente básica: despliegue

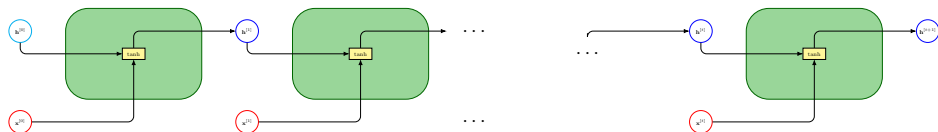


Unidad recurrente básica: despliegue de celdas



Modelando dependencias a corto plazo

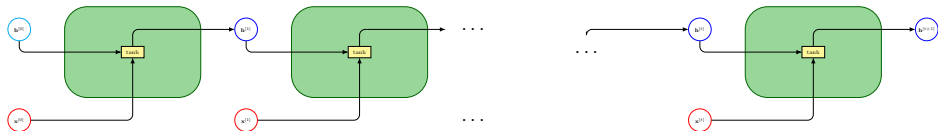
- ▶ En teoría una red recurrente básica puede modelar dependencias a corto y largo plazo
 - ▶ Siegelmann y Sontag mostraron que las redes recurrentes son Turing completas¹



¹Siegelmann and Sontag. On The Computational Power Of Neural Nets, 1995.

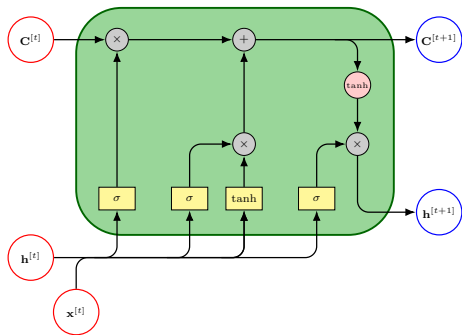
El problema de la memoria a largo plazo

- ▶ En práctica es muy difícil entrenarlas para tareas con dependencias a largo plazo



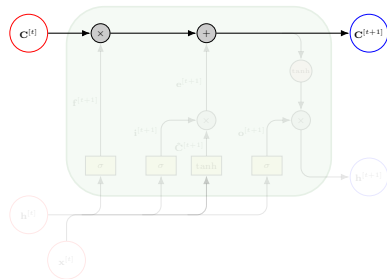
Long-short term memory (LSTM)

- ▶ Agregan elementos internos a la celda básica que permiten capturar dependencias a corto y largo plazo



Long-short term memory (LSTM): salida de la capa anterior

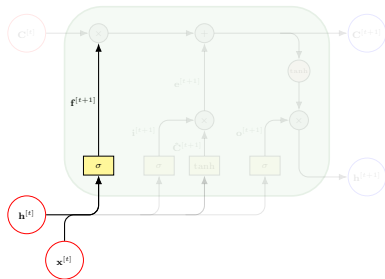
- Agrega o elimina información del estado $\mathbf{C}^{[t]}$ a partir de la entrada actual $\mathbf{x}^{[t+1]}$ y la salida anterior $\mathbf{h}^{[t]}$



Long-short term memory (LSTM): compuerta de olvido

- Determina qué olvidar del estado $\mathbf{C}^{[t]}$ y en qué proporción a partir de la entrada actual $\mathbf{x}^{[t+1]}$ y la salida anterior $\mathbf{h}^{[t]}$

$$\mathbf{f}^{[t+1]} = \sigma \left(\mathbf{w}_f \cdot [\mathbf{h}^{[t]}, \mathbf{x}^{[t+1]}] + \mathbf{b}_f \right)$$



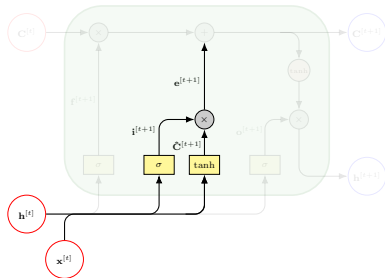
Long-short term memory (LSTM): computerta de entrada

- Determina qué agregar al estado $\mathbf{C}^{[t]}$ y en qué proporción a partir de la entrada actual $\mathbf{x}^{[t+1]}$ y el estado oculto anterior $\mathbf{h}^{[t]}$

$$\mathbf{i}^{[t+1]} = \sigma \left(\mathbf{W}_i \cdot \left[\mathbf{h}^{[t]}, \mathbf{x}^{[t+1]} \right] + \mathbf{b}_i \right)$$

$$\hat{\mathbf{C}}^{[t+1]} = \tanh \left(\mathbf{W}_c \cdot \left[\mathbf{h}^{[t]}, \mathbf{x}^{[t+1]} \right] + \mathbf{b}_c \right)$$

$$\mathbf{e}^{[t+1]} = \mathbf{i}^{[t+1]} \odot \hat{\mathbf{C}}^{[t+1]}$$

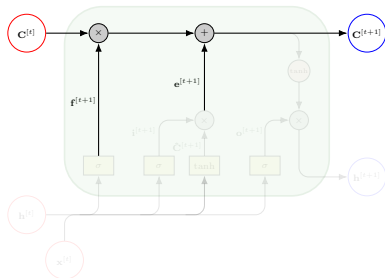


Long-short term memory (LSTM): nuevo estado

- El nuevo estado $\mathbf{C}^{[t+1]}$ se obtiene como una combinación del estado $\mathbf{C}^{[t]}$, la salida $\mathbf{f}^{(t)}$ de la compuerta de olvido y la salida $\mathbf{e}^{[t+1]}$ de la compuerta de entrada

$$\mathbf{C}^{[t+1]} = \mathbf{f}^{[t+1]} \odot \mathbf{C}^{[t]} + \mathbf{e}^{[t+1]}$$

donde \odot denota el producto de Hadamard

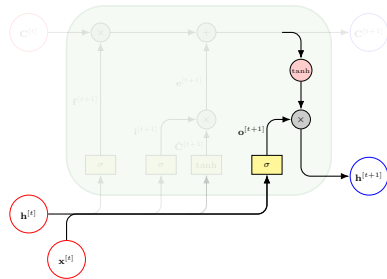


Long-short term memory (LSTM): computerta de salida

- El siguiente estado oculto $\mathbf{h}^{[t+1]}$ se obtiene como una combinación de la entrada actual $\mathbf{x}^{[t+1]}$, el estado oculto anterior $\mathbf{h}^{[t]}$ y el nuevo estado $\mathbf{C}^{[t+1]}$

$$\mathbf{o}^{[t+1]} = \sigma \left(\mathbf{W}_o \cdot \left[\mathbf{h}^{[t]}, \mathbf{x}^{[t+1]} \right] + \mathbf{b}_o \right)$$

$$\mathbf{h}^{[t+1]} = \mathbf{o}^{[t+1]} \odot \tanh \left(\mathbf{C}^{[t+1]} \right)$$



Gated recurrent unit (GRU)

- Combina compuertas de olvido y entrada en una sólo

$$\mathbf{z}^{[t+1]} = \sigma(\mathbf{W}_z \cdot [\mathbf{h}^{[t]}, \mathbf{x}^{[t+1]}] + \mathbf{b}_z)$$

$$\mathbf{r}^{[t+1]} = \sigma(\mathbf{W}_r \cdot [\mathbf{h}^{[t]}, \mathbf{x}^{[t+1]}] + \mathbf{b}_r)$$

$$\tilde{\mathbf{h}}^{[t+1]} = \tanh(\mathbf{W}_h \cdot [\mathbf{r}^{[t+1]} \odot \mathbf{h}^{[t]}, \mathbf{x}^{[t+1]}] + \mathbf{b}_h)$$

$$\mathbf{h}^{[t+1]} = (1 - \mathbf{z}^{[t+1]}) \odot \mathbf{h}^{[t]} + \mathbf{z}^{[t+1]} \odot \tilde{\mathbf{h}}^{[t+1]}$$

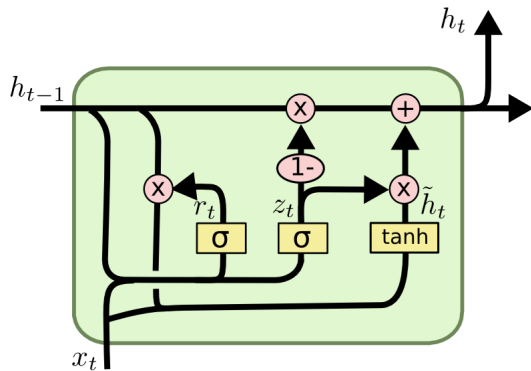


Imagen tomada de C. Olah. Understanding LSTM Networks, 2015.

Arquitecturas de redes recurrentes

- ▶ Contienen celdas recurrentes en conjunto con otras capas
- ▶ La salida de una celda alimenta otras capas u otras celdas
- ▶ Por ejemplo, para predecir el siguiente símbolo en un texto con una celda recurrente básica, a la salida podemos agregar una capa densa con función de activación *softmax*

$$\hat{\mathbf{y}}^{[t+1]} = \text{softmax}(\mathbf{W}_y \cdot \mathbf{h}^{[t+1]} + \mathbf{b}_y)$$

Arquitecturas de redes recurrentes: ejemplo

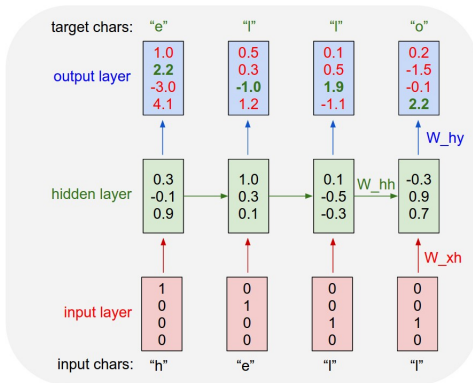


Imagen tomada de Karpathy 2015 (<http://karpathy.github.io/2015/05/21/rnn-effectiveness/>)

Arquitecturas de redes recurrentes: tareas de uno a uno

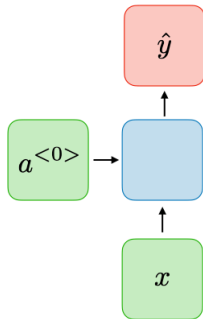


Imagen tomada de Amidi. Recurrent Neural Networks cheatsheet

Arquitecturas de redes recurrentes: tareas de uno a muchos

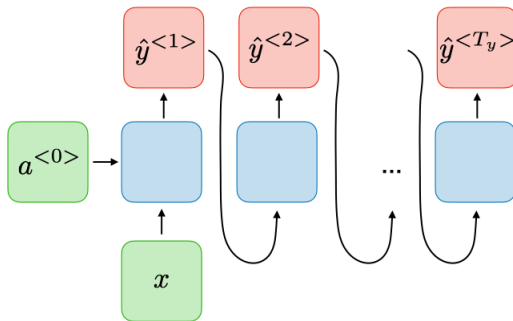


Imagen tomada de Amidi. Recurrent Neural Networks cheatsheet

Arquitecturas de redes recurrentes: tareas de muchos a uno

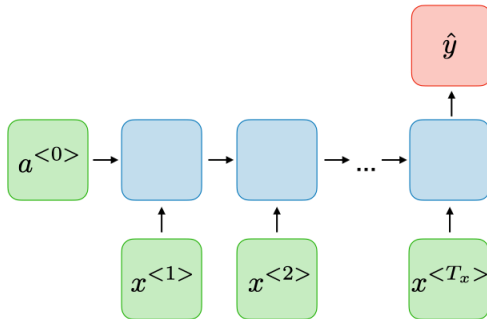


Imagen tomada de Amidi. Recurrent Neural Networks cheatsheet

Arquitecturas de redes recurrentes: tareas de muchos a muchos

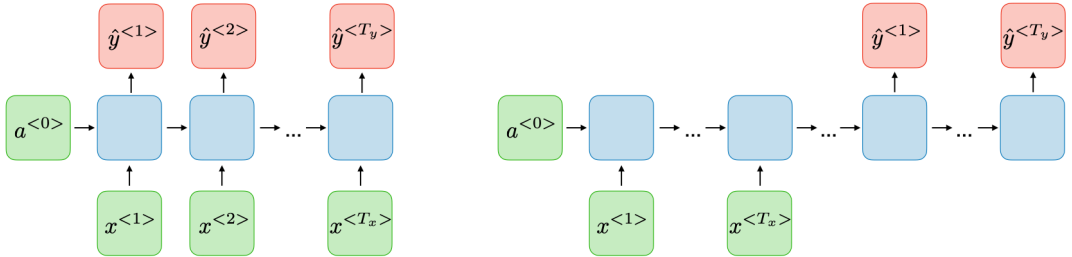


Imagen tomada de Amidi. Recurrent Neural Networks cheatsheet

Arquitecturas de redes recurrentes: LSTM/GRU bidireccional

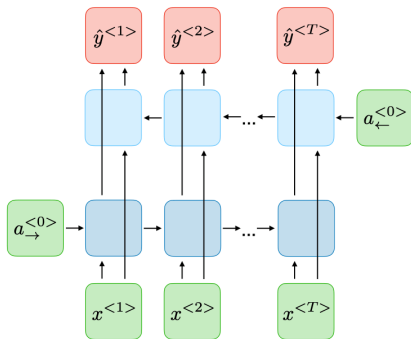


Imagen tomada de Amidi. Recurrent Neural Networks cheatsheet

Arquitecturas de redes recurrentes: celdas apiladas

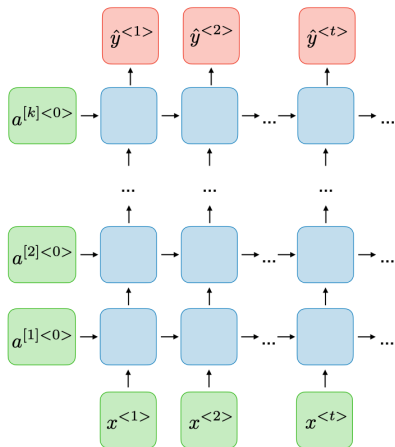


Imagen tomada de Amidi. Recurrent Neural Networks cheatsheet

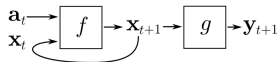
Algoritmo de retropropagación de errores para redes recurrentes

- ▶ Pérdida en el tiempo

$$\mathcal{L}(\hat{\mathbf{y}}, \mathbf{y}) = \sum_{t=1}^T \mathcal{L}(\hat{\mathbf{y}}^{[t]}, \mathbf{y}^{[t]})$$

- ▶ Retropropagación

$$\frac{\partial \mathcal{L}^{[T]}}{\partial \theta} = \sum_{t=1}^T \frac{\partial \mathcal{L}^{[t]}}{\partial \theta}$$



⇓ unfold through time ⇓

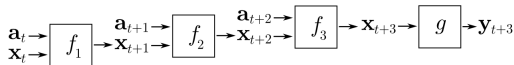


Imagen tomada de Wikipedia (https://en.wikipedia.org/wiki/Backpropagation_through_time)

Referencias

- ▶ Zhang, A., Lipton, Z. C., Li, M., & Smola, A. J. (2021). Dive into deep learning. Capítulos 8 y 9. Disponible en <http://www.d2l.ai/>.
- ▶ Murphy, K. P. (2022). Probabilistic Machine Learning: An introduction. MIT Press. Capítulo 15, sección 15.2. Disponible en <https://probml.github.io/pml-book/book1.html>.
- ▶ Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. MIT press. Capítulo 10. Disponible en <https://www.deeplearningbook.org/>.
- ▶ Amidi, A. & Amidi, S. Recurrent Neural Networks cheatsheet. Disponible en <https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-recurrent-neural-networks>.
- ▶ Olah, C. (2015). Understanding LSTM Networks. Disponible en <https://colah.github.io/posts/2015-08-Understanding-LSTMs/>.
- ▶ Karpathy A. (2015). The Unreasonable Effectiveness of Recurrent Neural Networks. Disponible en <http://karpathy.github.io/2015/05/21/rnn-effectiveness/>.

Contacto

Gibran Fuentes Pineda

Investigador Asociado "C", IIMAS, UNAM

gibranfp@unam.mx

LinkedIn: @gibranfuentespineda