# Procesamiento del Lenguaje Natural mediante Redes Neuronales

Día 3: Redes Neuronales Recurrentes

Germán Kruszewski
Facebook AI Research

# Subscripción a la lista de mails

- Si no te llegan los mails, inscribite escribiendo a eci2019nlp-alu-request@dc.uba.ar

# Modelación del lenguaje (LM)

- El objetivo es predecir qué palabra sigue a un determinado contexto:

El niño juega en el _____

| Vocabulario | P |
|---|---|
| a | 1e-10 |
| barco | 2e-3 |
| … | |
| parque | 0.15 |
| patio | 0.1 |
| … | … |

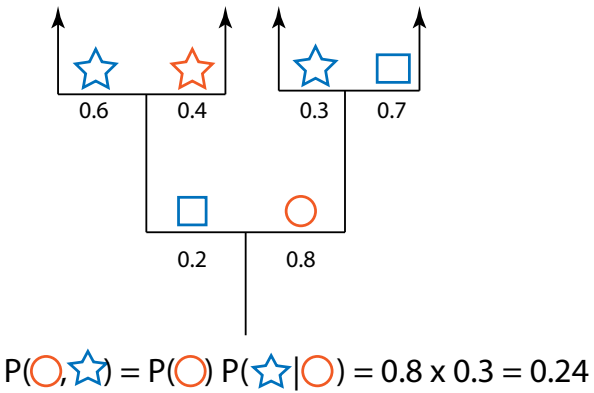
- Formalmente, dado el contexto $x_1, \dots, x_t$ queremos predecir la probabilidad de ver la siguiente palabra $x_{t+1}$

$$P(x_{t+1}|x_t, \dots, x_1)$$

donde $x_i$ es una palabra perteneciente a un vocabulario $V$ prefijado con antelación.

# Modelación del lenguaje (LM)

- Equivalentemente, un modelo del lenguaje puede asignar una probabilidad a cualquier secuencia de palabras.
- P([el, niño, juega]) = P(el)P(niño | el)P(juega | niño, el)



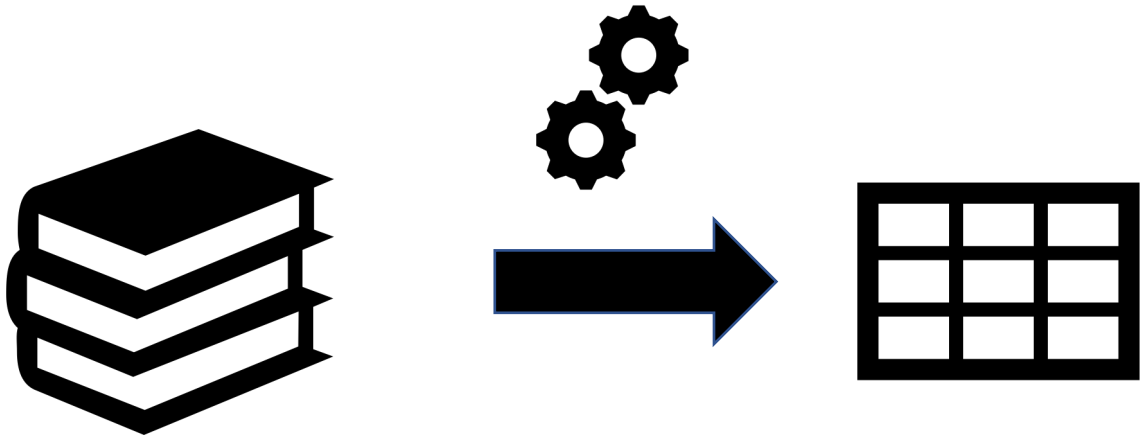
# Modelos del lenguaje en la vida

# Modelos del lenguaje en la vida

# Otras Aplicaciones

- Reconocimiento de voz (“Recognize speech” / “Wreck a nice beach”)
- Reconocimiento de escritura manuscrita
- Corrección gramatical
- Identificación de autor
- Traducción automática
- Etc.

# Construcción de un modelo del lenguaje



Grandes cantidades de texto

Estadísticas

# Modelos del lenguaje basados en n-grams

- Idea: Usamos cuántas veces aparece cada secuencia de no más de $n$ palabras en un gran corpus de texto:
-
$$P(x_n|x_{n-1}, \dots, x_1) = \frac{\text{frec}(x_n, x_{n-1}, \dots, x_1)}{\text{frec}(x_{n-1}, \dots, x_1)}$$
- Por ejemplo, un modelo 4-gram:

$$P(\boldsymbol{w} \in \boldsymbol{V}|\text{juega, en, el}) = \frac{\text{frec}(\text{juega, en, el, } \boldsymbol{w})}{\text{frec}(\text{juega, en, el})}$$

Calcular estas estadísticas es muy rápido. ✓

# Generación de texto

- Podemos usar las probabilidades de la palabra siguiente para generar texto:

condicionamos condicionamos

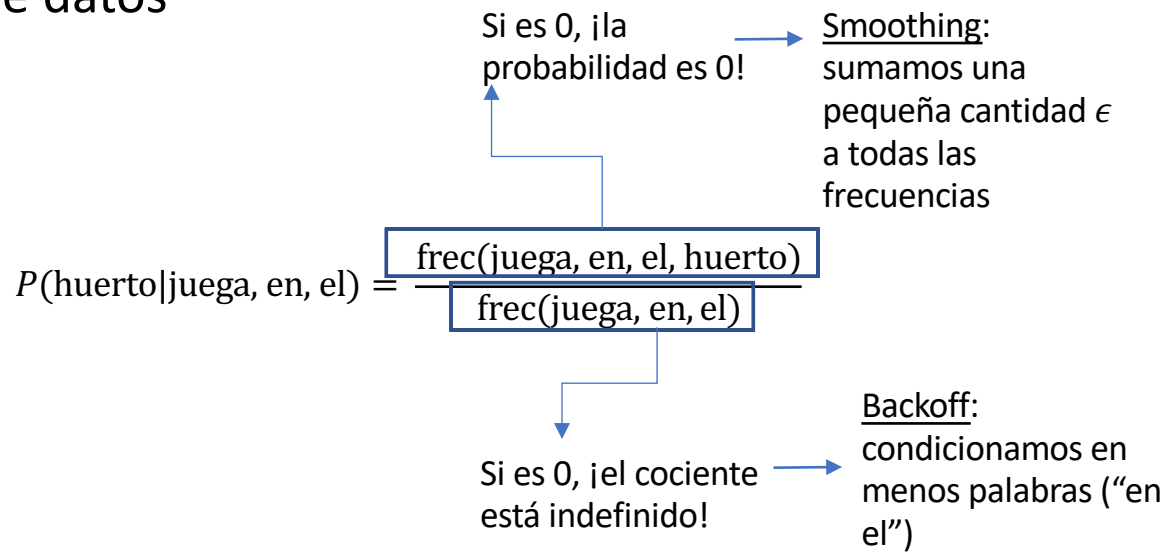
| Vocabulario | P |
|---|---|
| a | 1e-10 |
| Dulcinea | 2e-3 |
| Dulcinea | 1e-3 |
| … | … |
| mas | 0.1 |
| más | 1e-9 |
| pato | 1e-1 |
| patio | 1e-4 |
| … | … |

Cuanto más que a don Quijote y halló que montaban setenta y tres noches estuvo don Quijote , adonde podía vivir con más facilidad rapasen vuestras aventuras , y duermen , por gozar dél como de marca de fardo , que no fuese entendida : - ¡ Si , señor , si no sigo tu parecer y éste dijo : - Sancho amigo , que yo procuraré no apartarme destos contornos - dijo don Quijote - , ¿ por qué quieres poner esta borrica en mi muerte .

Hecho siguiendo https://nlpforhackers.io/language-models/

# Problemas con los modelos basados en n-grams

- Escasez de datos



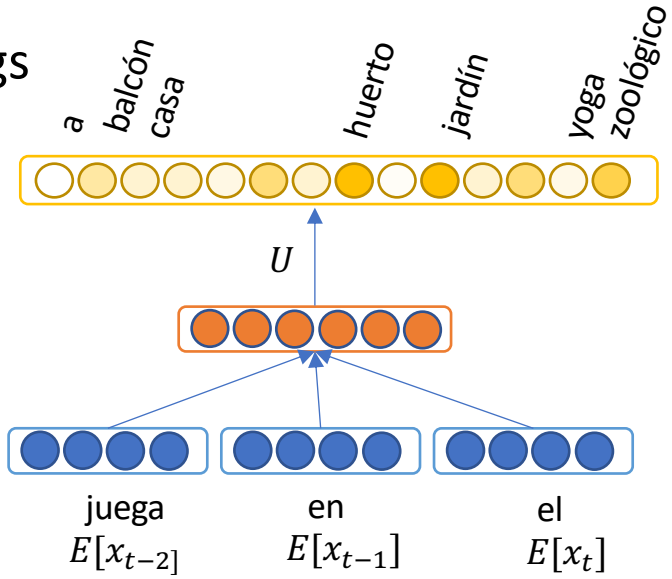

- Espacio en memoria

# Usando la similitud entre las palabras para solventar la escasez de datos

- Quizás no vimos “juega en el huerto” pero vimos “juega en el jardín”.
- Jardín y huerto posiblemente aparecen en contextos similares.

# Modelo neuronal del lenguaje

- Representamos las $n$ palabras anteriores por sus word embeddings.
- Idea inicial: Bolsa de Word Embeddings
- Problema:
  “juega en el” = “el juega en”



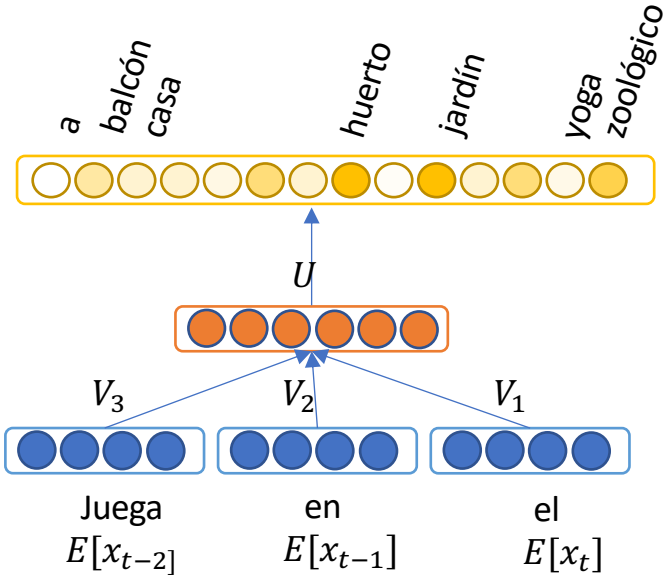
# Modelo neuronal del lenguaje

$$h_t = \tanh\left(\sum_{i=0}^{n-1} V_i E[x_{t-i}] + b_h\right)$$

$$\hat{y_t} = \text{softmax}(Uh_t + b_y)$$

Donde $x_{t-i}$ es el índice de la palabra en la posición $t - i$ y $E$ es una matriz de word embeddings.

- ✓ Tamaño del modelo independiente de la cantidad de datos
- ✓ Extrapola basado en similitudes




Bengio et al. (2003)

# Más problemas

- Seguimos pudiendo condicionar únicamente en las últimas $n$ palabras. ¿Qué pasa con la siguiente oración?
  "Messi, el famoso delantero argentino, juega en el _____"
  ¿Qué valor de $n$ es suficiente?
- Número de palabras hacia atrás no es una buena heurística para el lenguaje:
  Messi, el famoso delantero argentino, juega en el _
  Messi, el delantero argentino, juega en el _
  Messi, el delantero, juega en el _

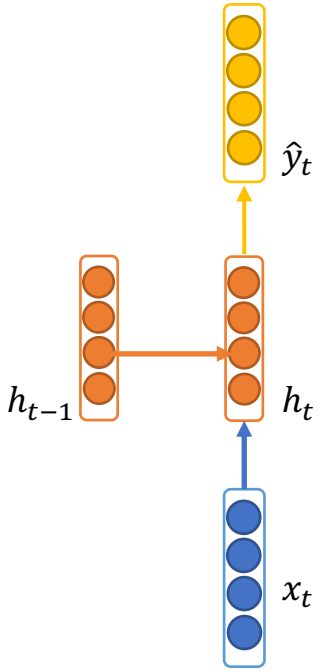
# Redes neuronales recurrentes (RNNs)

- Vamos a escribir una red con una memoria ($h_t$).
- Cada vez que vemos una palabra $x_t$ actualizamos la memoria $h_{t-1}$ y obtenemos un nuevo estado de memoria $h_t$:

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

- Luego predecimos la palabra siguiente usando el estado de la memoria:

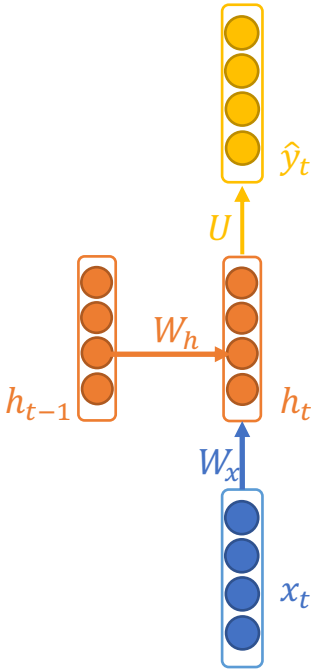
$$\hat{y} = g(h_t)$$



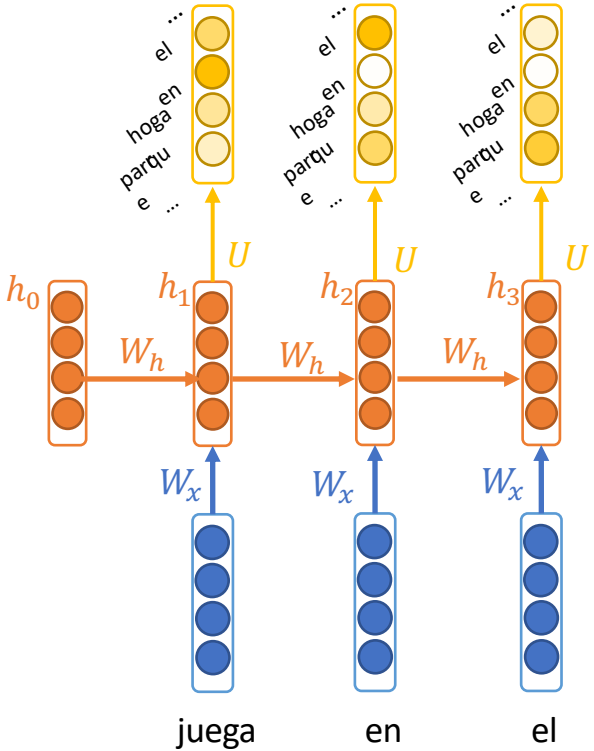

(Elman, 1990)

# Simple RNNs

- $h_t = RNN(h_{t-1}, x_t) = \sigma(W_h h_{t-1} + W_x E[x_t] + b_h)$
- $\hat{y} = \text{softmax}(U h_t + b_y)$



# RNNs vista desdoblada (unfolded)




$$h_t = \sigma(W_h h_{t-1} + W_x E[x_t] + b_h)$$

$$\hat{y} = \text{softmax}(U h_t + b_y)$$

# Aplicación recurrente de una función

$$h_1 = \sigma(W_h h_0 + W_x E[x_0] + b_h)$$

$$h_2 = \sigma(W_h \sigma(W_h h_0 + W_x E[x_0] + b_h) + W_x E[x_1] + b_h)$$

$$h_3 = \sigma(W_h \sigma(W_h \sigma(W_h h_0 + W_x E[x_0] + b_h) + W_x E[x_1] + b_h) + W_x E[x_2] + b_h)$$

# Entrenamiento de una RNN

- En un gran corpus de texto, nos proponemos predecir la siguiente palabra dado el texto anterior

| $x$ | En | un | lugar | de | la | Mancha | , | de | cuyo |
|---|---|---|---|---|---|---|---|---|---|
| $y$ | un | lugar | de | la | Mancha | , | de | cuyo | nombre |

- Función de pérdida (entropía cruzada):

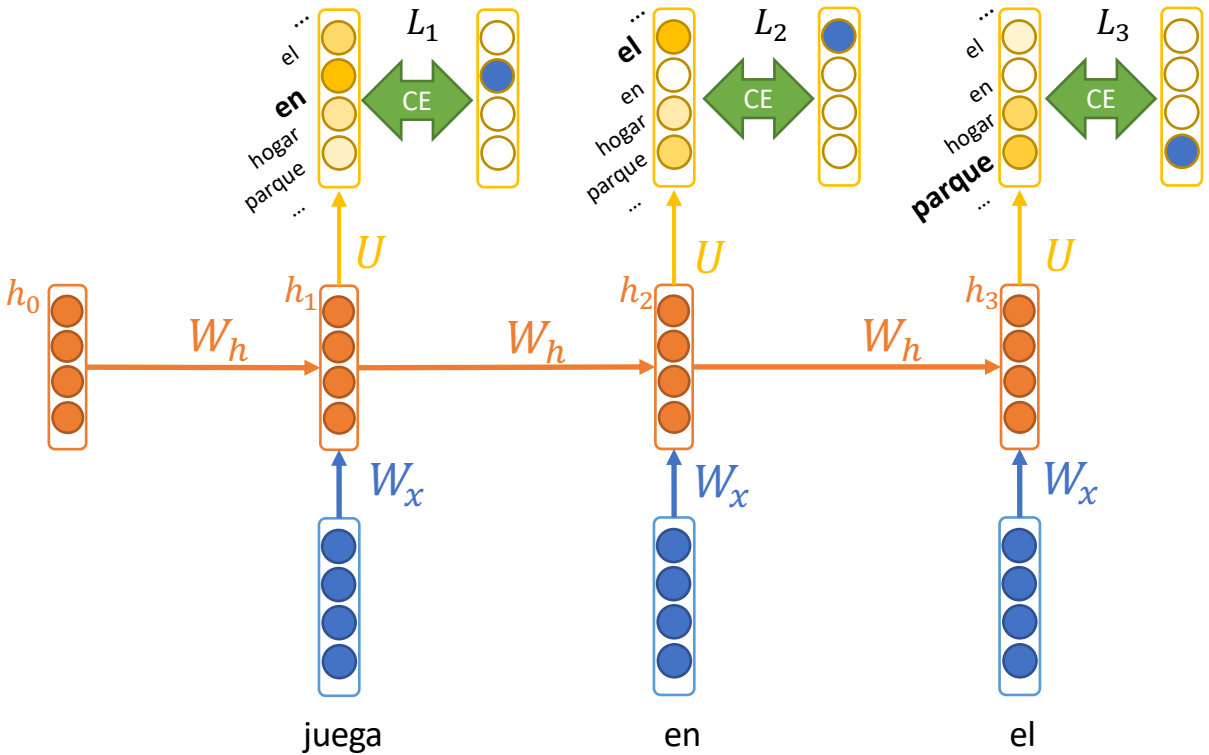
$$L_t(\Theta) = CE(y_t, \hat{y}_t) = -\sum_{w \in V} y_t[w] \log \hat{y}_t[w]$$

$$= -\log \hat{y}_t[x_{t+1}]$$

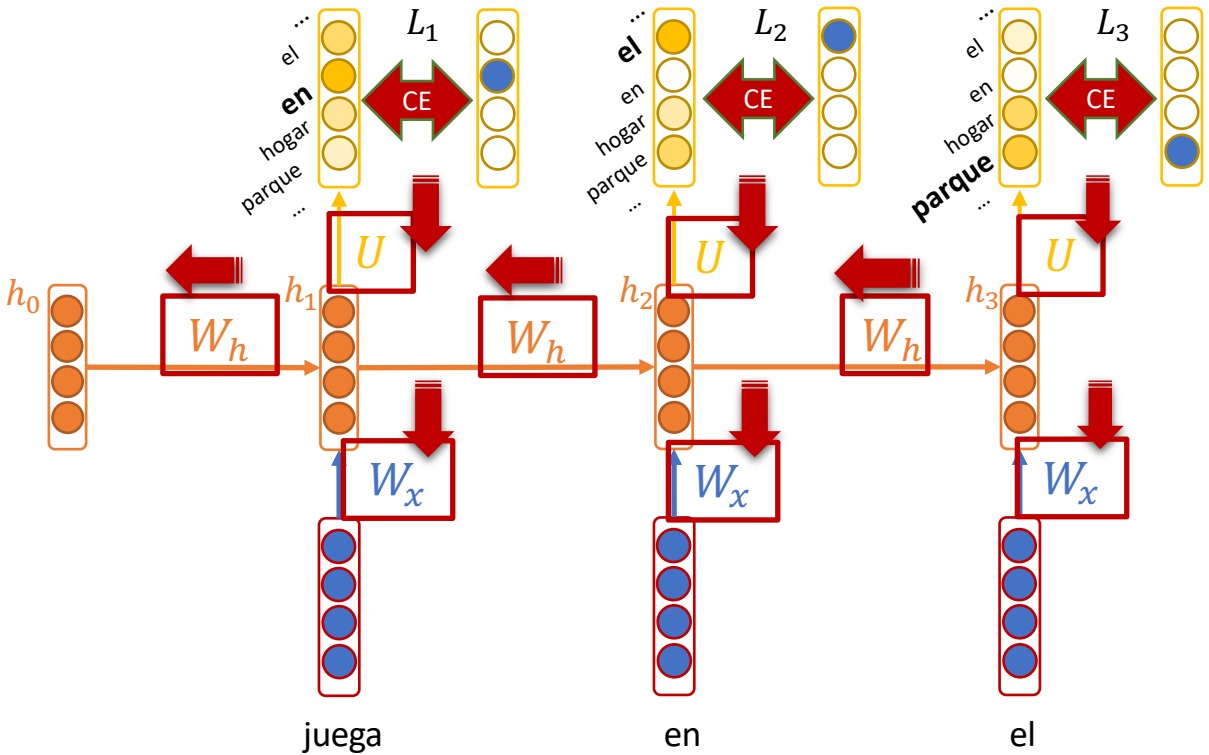
- Tomamos el promedio sobre todos los ejemplos:

$$L(\Theta) = \frac{1}{T}\sum_{t=1}^{T} L_t(\Theta) = \frac{1}{T}\sum_{t}^{T} -\log \hat{y}_t[x_{t+1}]$$

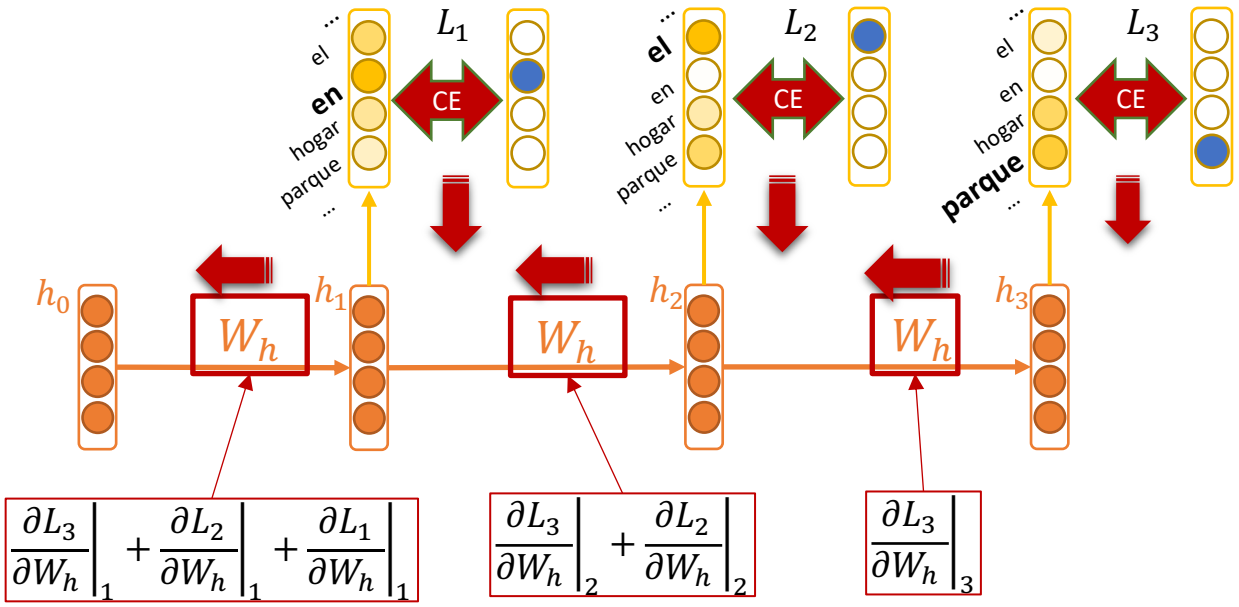
# Entrenamiento de una RNN



# Backpropagation (BPTT)



# Backpropagation (BPTT)



# Implementación práctica

- Supongamos que tenemos 1M de palabras.
- El vector memoria $h_{t=1531}$ depende de $h_0$. Tenemos que hacer backpropagation por los 1531 estados?
- No, hacemos backpropagation cortando el texto en segmentos.
- Idea del algoritmo:
  - Tomamos un segmento de los datos $x = T[i: t+s], y = T[i+1, t+s+1]$
  - Calculamos $\hat{y}, h_{t+s} = RNN(h_i, x)$; $\text{loss} = \text{L}(\hat{y}, y)$
  - Backpropagamos hasta $h_i$
  - Repetimos pasando parar adelante la memoria $h_{t+s}$