.UBAfiuba
FACULTAD DE INGENIERÍA

NLP

Redes neuronales recurrentes (RNNs)

Dr. Rodrigo Cardenas Szigety
rodrigo.cardenas.sz@gmail.com

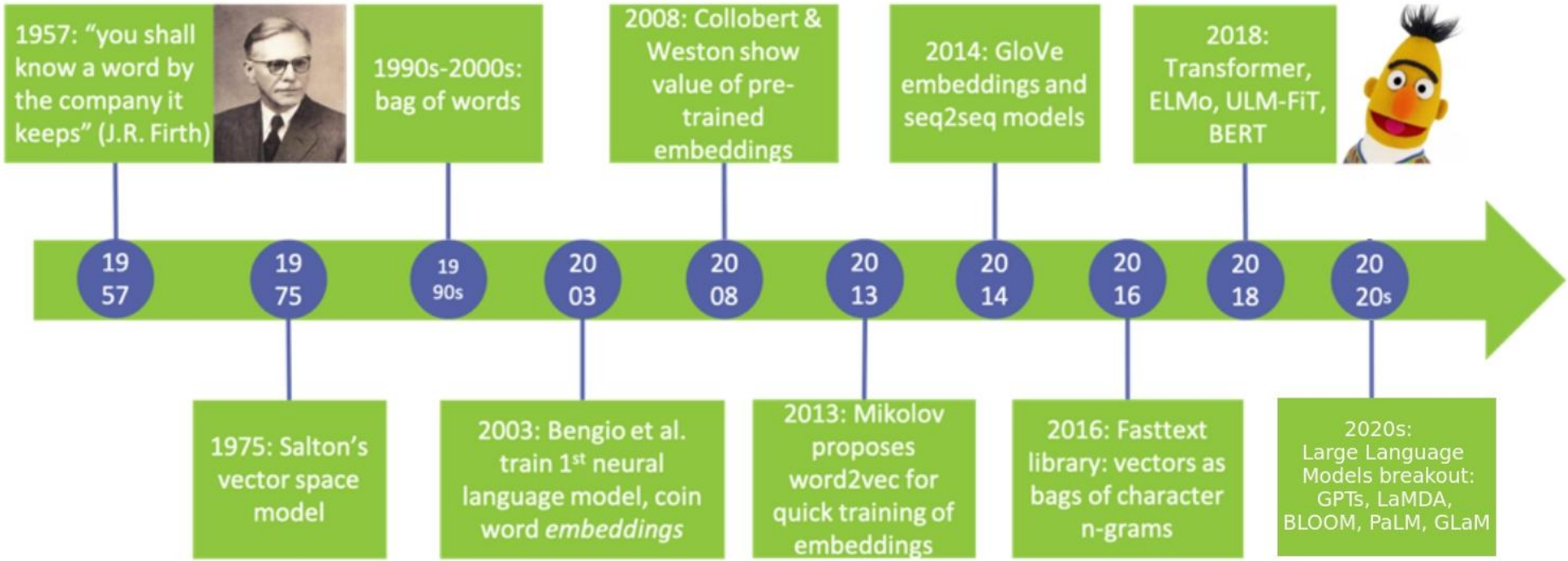
Programa de la materia

Clase 1: Introducción a NLP, Vectorización de documentos.
Clase 2: Preprocesamiento de texto, librerías de NLP, bots de información.
Clase 3: Word Embeddings, CBOW y SkipGRAM, entrenamiento de embeddings.
Clase 4: Redes recurrentes (RNN), problemas de secuencia y estimación de próxima palabra.
Clase 5: Redes LSTM, análisis de sentimientos.
Clase 6: Modelos Seq2Seq, traductores y bots conversacionales.
Clase 7: Celdas con Attention. Transformers, BERT & ELMo, fine tuning.
Clase 8: Cierre del curso, NLP hoy y futuro, deploy.

*Unidades con desafíos a presentar al finalizar el curso.
*Último desafío y cierre del contenido práctico del curso.

Timeline

1990: Celda RNN básica (Elman)
1997: LSTM



2016: LSTMs se convierten en SotA para traducción automática

Redes Neuronales Recurrentes (RNNs)

Es un tipo de neurona con un estado interno (o memoria) de manera que la información del pasado influye en los resultados futuros.

Se utiliza principalmente para resolver problemas de secuencia, en donde el valor anterior está relacionado con el valor futuro.

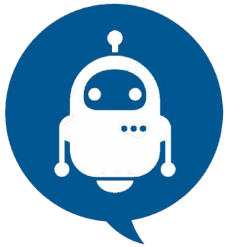
Permite construir modelos cuyos vectores de entrada o salida no posean una dimensión fija.

Implementa modelos de lenguaje de la forma:

$$\prod_{i=1}^{i=m} P(w_i | w_{i-(n-1)}, \ldots, w_{i-1})$$

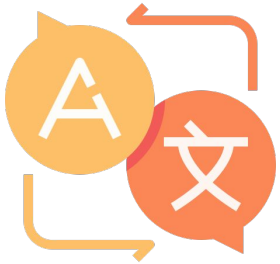
"Hoy el día está hermoso y despejado, se puede ver un hermoso cielo... azul"

Algunos problemas de secuencia



Bots Conversacionales

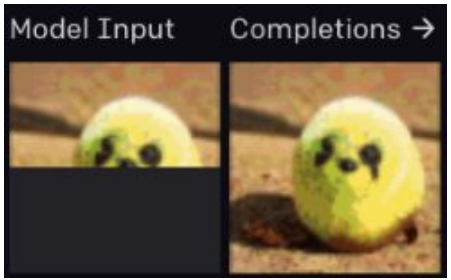
Name entity recognition



Traducción de idiomas

Speech to text

Generar música



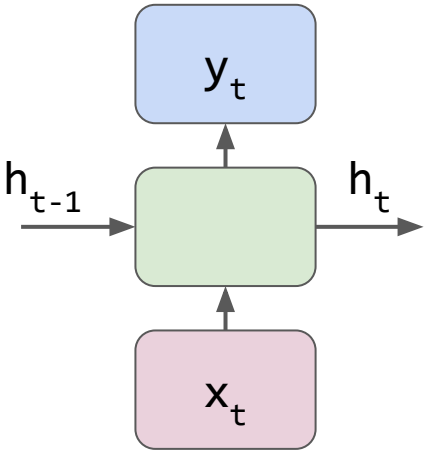
Completar una imagen

Celda RNN básica (Elman)

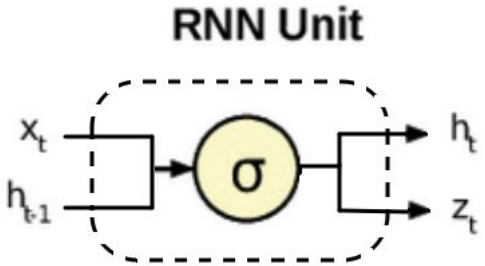
LINK API KERAS

Forward (implementación de TF SimpleRNN)

$$h_t = \sigma(W_{hh} * h_{t-1} + W_{hx} * x + b_h)$$
$$z_t = h_t$$

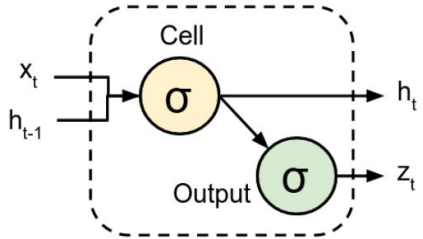


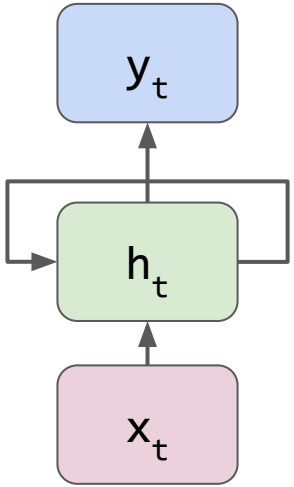
Unidad básica



Salida general

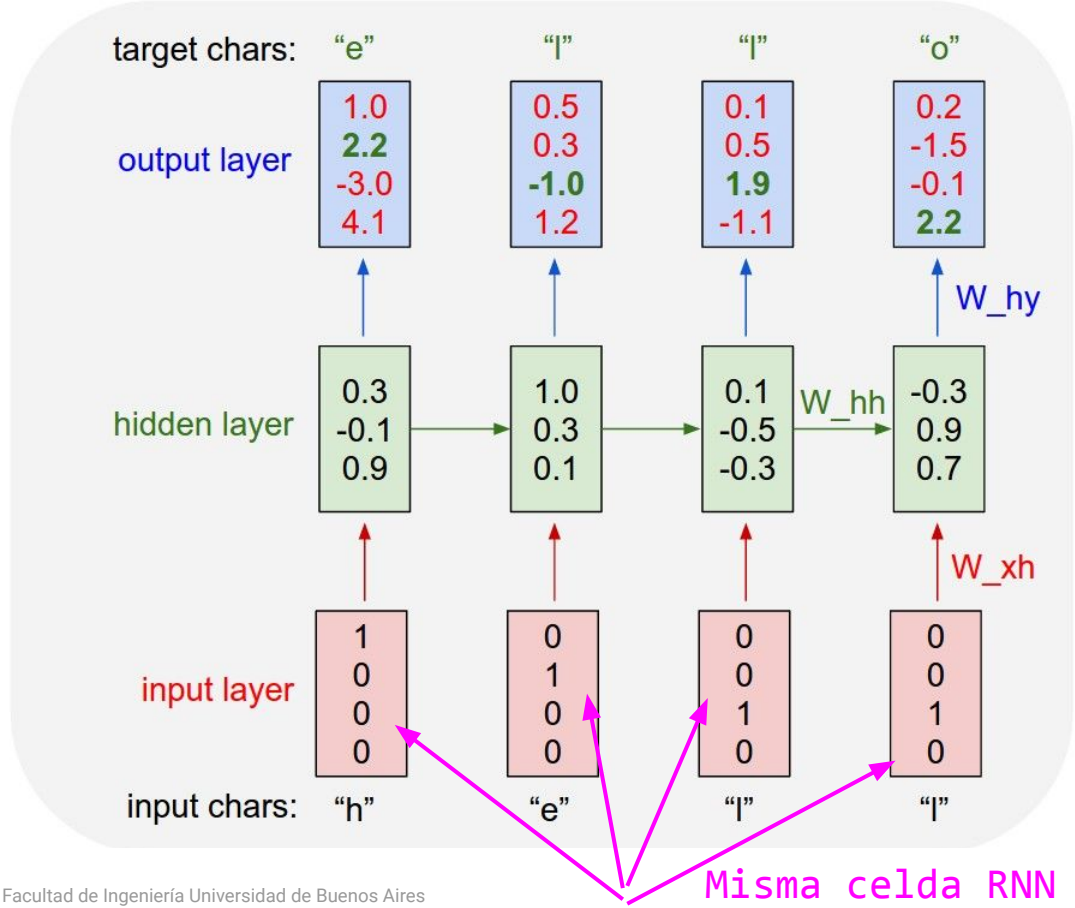
$$z_t = \sigma(W_{hy} * h_t + b_z)$$





Representación compacta

Propagación (ejemplo)



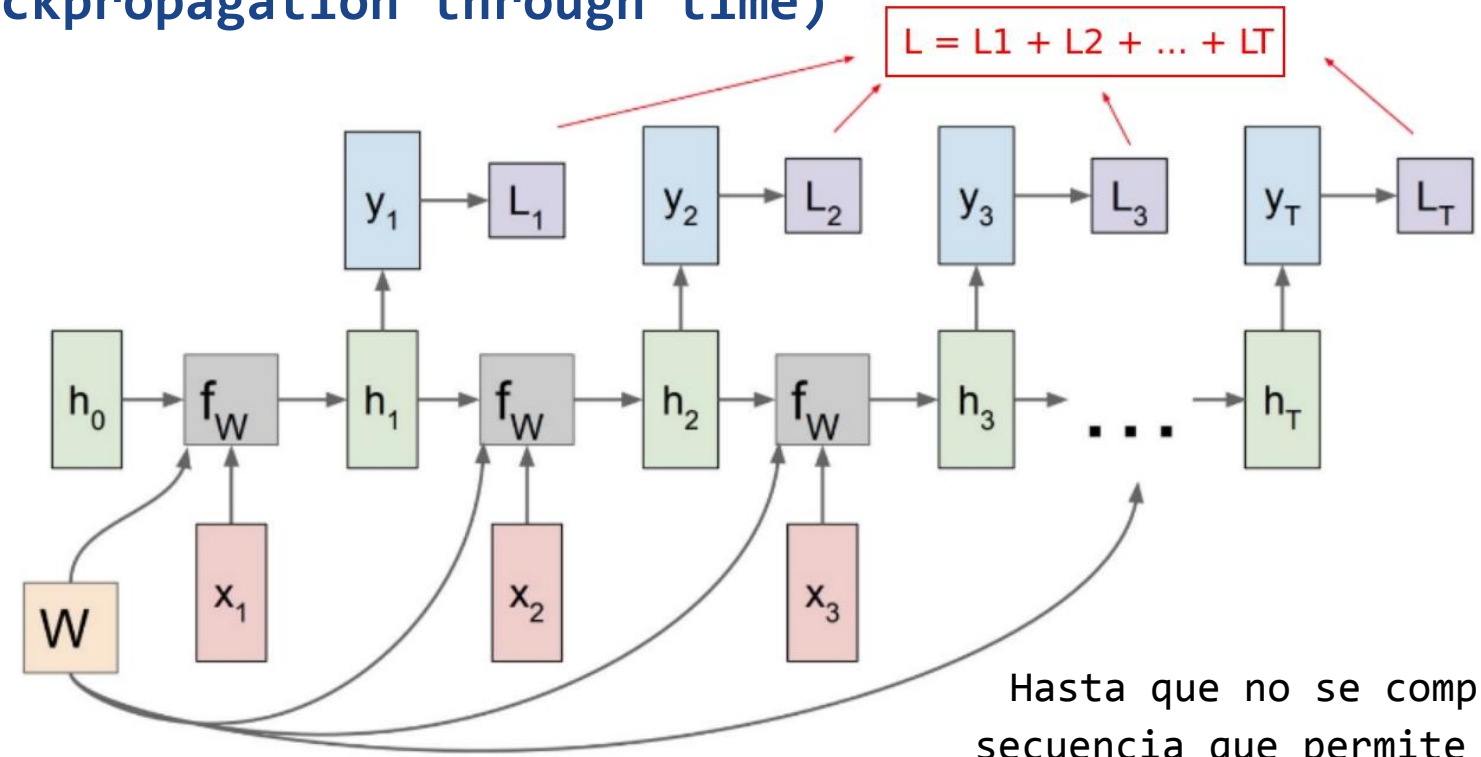
En este ejemplo conceptual entra una palabra/letra y sale otra

En estas redes de secuencia su grafo de cómputo es en serie, no es posible paralelizar, ya que el estado futuro depende del estado anterior.

Con cada salida se actualizan los pesos Whh, Wxh y Why para el próximo cómputo

Misma celda RNN en diferentes instantes

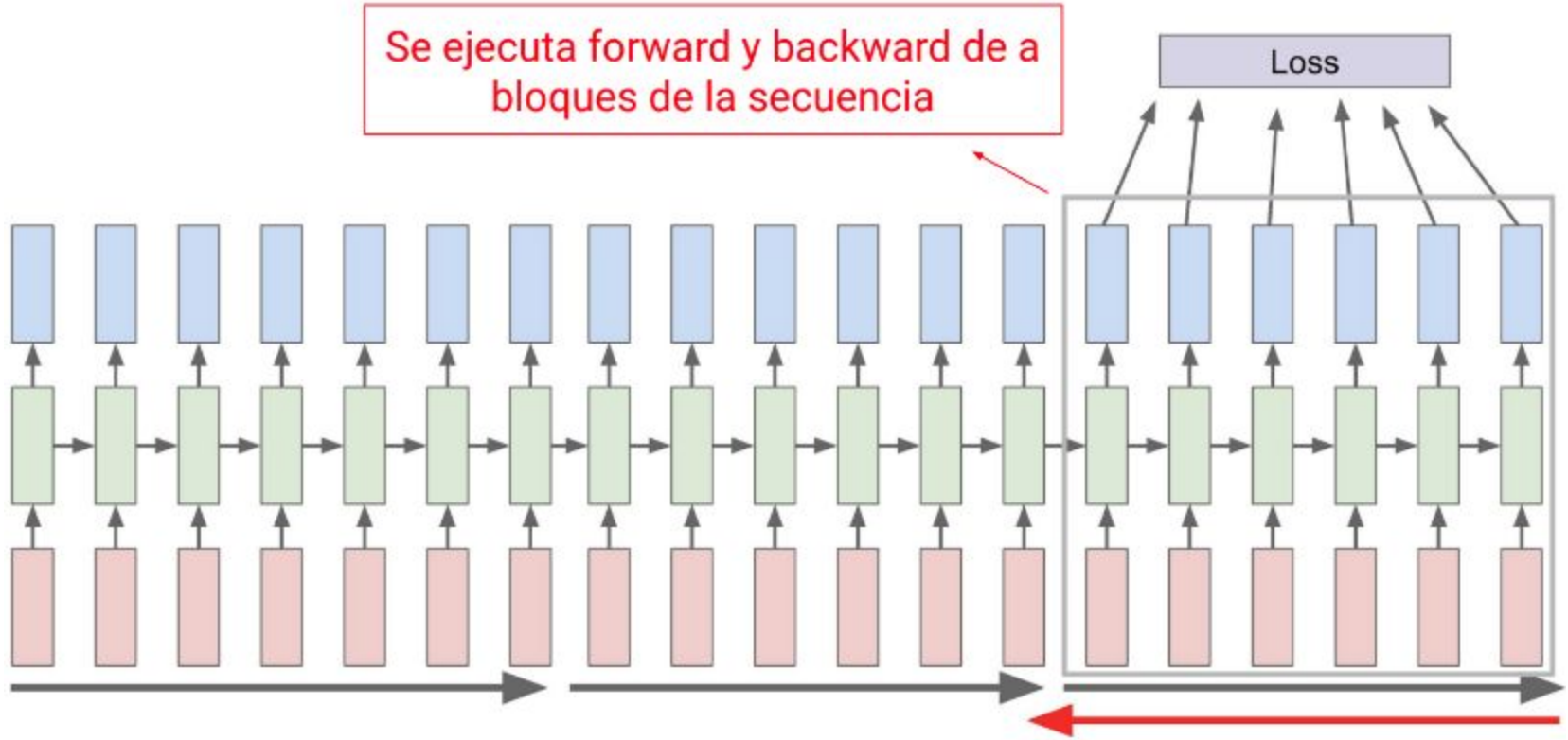
Grafo de cómputo de una RNN y BPTT (Backpropagation through time)



Hasta que no se complete la secuencia que permite calcular el loss no se actualizan los pesos (W) de la/s celda/s

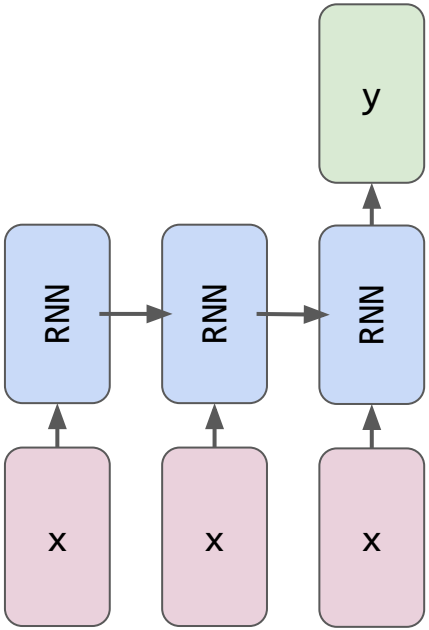
Forward & backward

LINK BACKPROPAGATION NUMPY

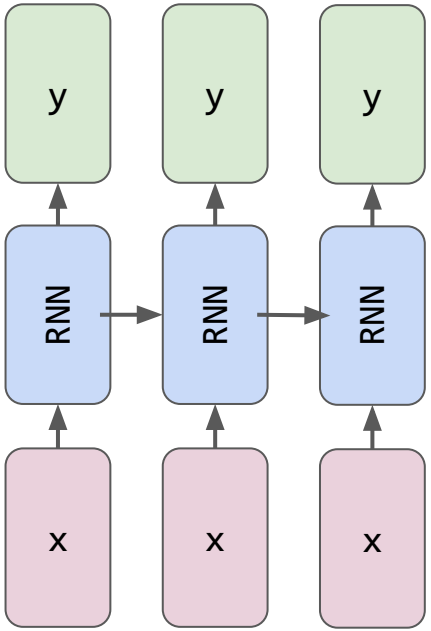


Arquitecturas

LINK



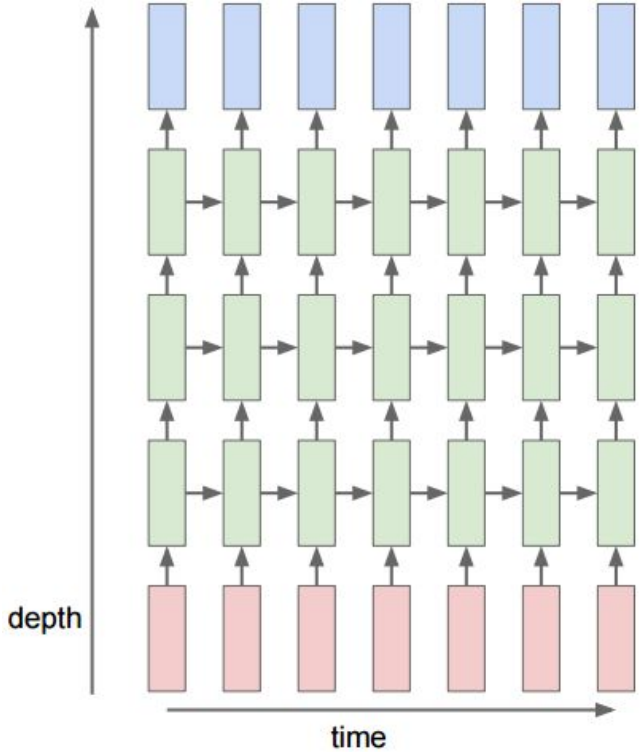
many-to-one



many-to-many

Multi-layer RNN

Tal como se vio en los ejemplos se trata de apilar layers RNN en donde la salida de una se traslada a la entrada de la siguiente



Problema de una RNN tradicional

LINK

"Una RNN tradicional solo usa información del pasado y no de las futuras palabras para predecir"

Ejemplo: Data una sentencia determinar si existe una entidad que represente al nombre de una persona utilizando (name entity recognition)

Ejemplo 1:
“Hoy escuche que Victoria terminó su bot para NLP” → persona

Ejemplo 2:
“Hoy escuche que Victoria cambió de intendente” → ciudad

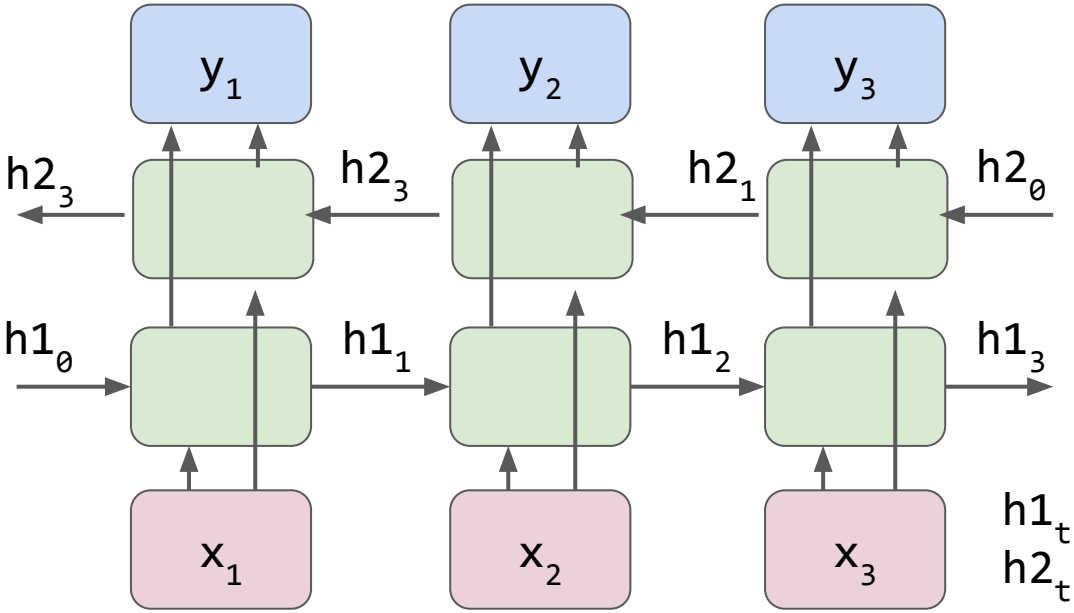
La contextualización de la palabra es a futuro

Bidirectional RNN (BRNN)

BRNN PAPER

API KERAS

"La palabra anterior y la palabra futura tienen impacto en la presente predicción"

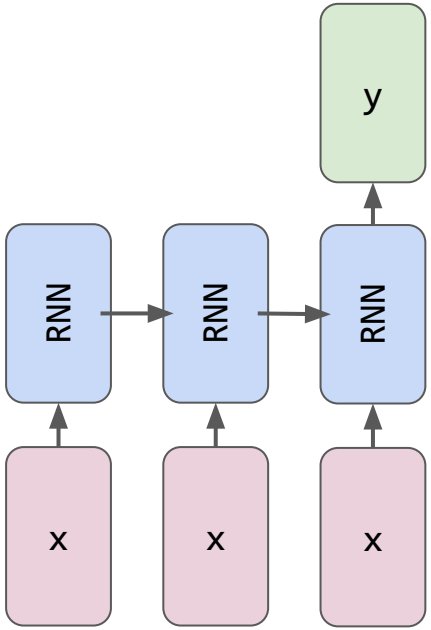


$$h1_t = \sigma(W_{hh1} * h1_{t-1} + W_{hx1} * x + b_1)$$
$$h2_t = \sigma(W_{hh2} * h2_{t+1} + W_{hx2} * x + b_2)$$

Las dos salidas pueden concatenarse, sumarse o promediarse

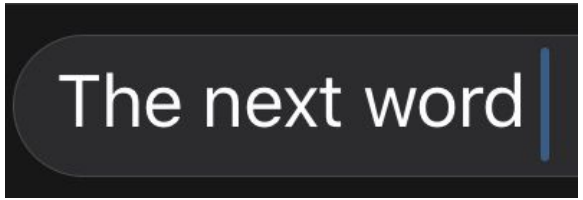
many-to-one

"Dada una sentencia o oración de entrada de tamaño fijo, el sistema arroja un único resultado que la representa".



many-to-one

Este tipo de estructuras se utilizan para determinar cuál es la siguiente palabra o elemento en la secuencia o para clasificación (sentiment analysis).



Predicción de próxima palabra

Análisis de sentimientos

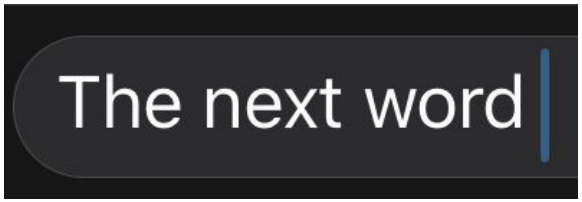
many-to-one

Link al Colab

LINK

Predicción de texto/modelos de lenguaje

Se utilizará many-to-one, por lo que hay que seleccionar la dimensión de la sentencia de entrada y dividir el texto en grupos:



Predicción de próxima palabra

Sentencia

```
'Yesterday, all my troubles seemed so far away'
```

Tokens

```
['yesterday', 'all', 'my', 'troubles', 'seemed', 'so', 'far', 'away']
```

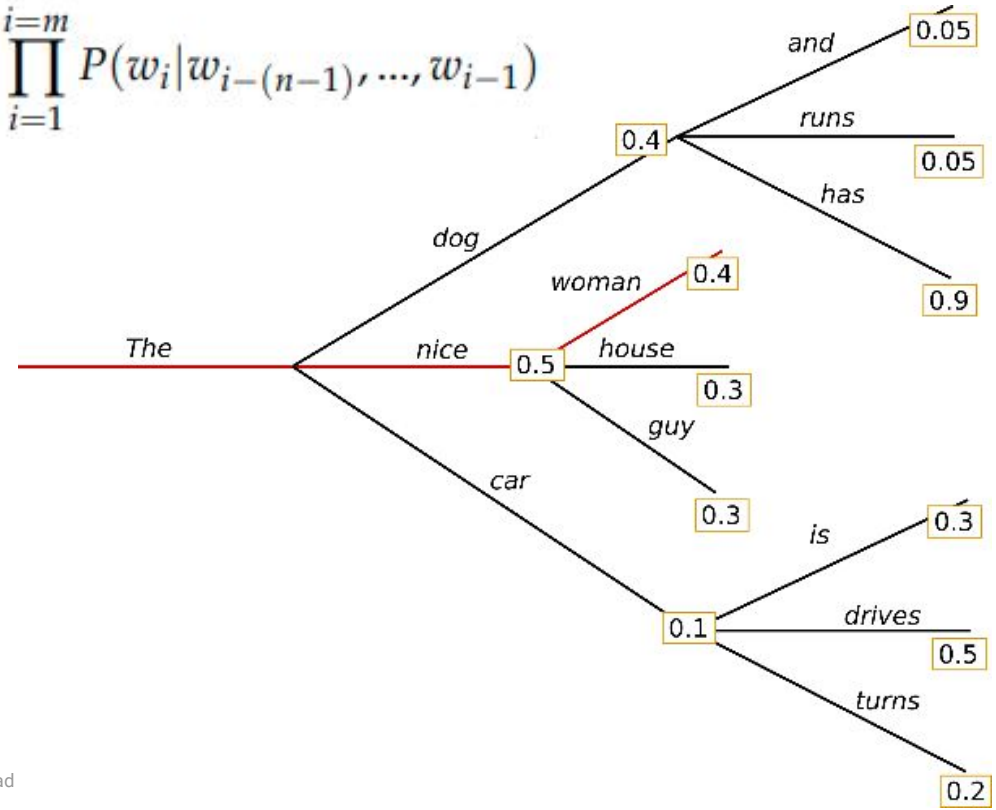
$$\prod_{i=1}^{i=m} P(w_i | w_{i-(n-1)}, \ldots, w_{i-1})$$

Vectores de entrada de 4 tokens

```
[['yesterday', 'all', 'my', 'troubles'],
 ['all', 'my', 'troubles', 'seemed'],
 ['my', 'troubles', 'seemed', 'so'],
 ['troubles', 'seemed', 'so', 'far']]
```

Generación de texto en modelos de lenguaje: Beam search

$$\prod_{i=1}^{i=m} P(w_i|w_{i-(n-1)},...,w_{i-1})$$



`num_beams=2`

Paso 1:
"The nice" → 0.5
"The dog" → 0.4

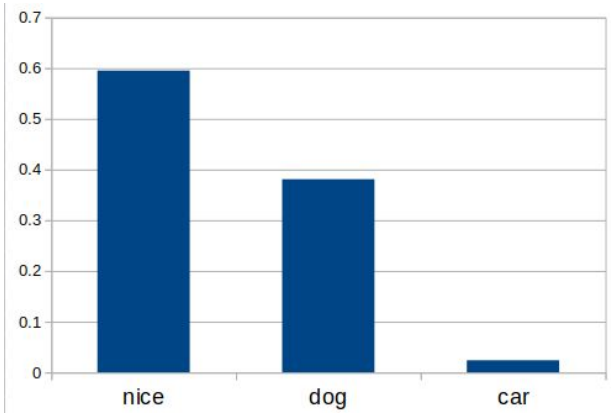
Paso 2:
"The nice woman" → 0.5x0.4 = 0.2
"The dog has" → 0.4x0.9 = 0.36

Siempre encuentra una secuencia con prob >= que Greedy Search

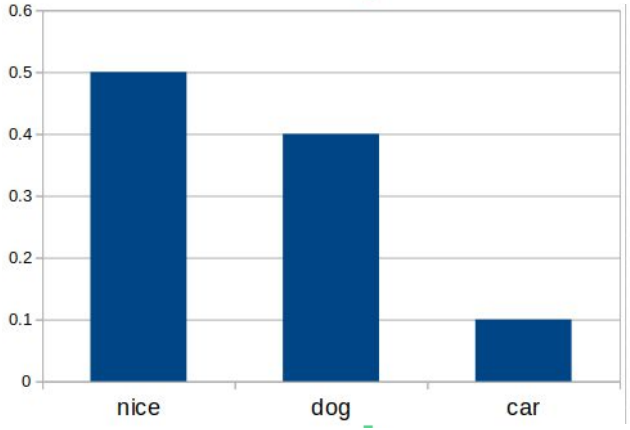
Es un método heurístico. No asegura que encontrar la más probable. ("The car drives")

Generación de texto en modelos de lenguaje: Muestreo con temperatura

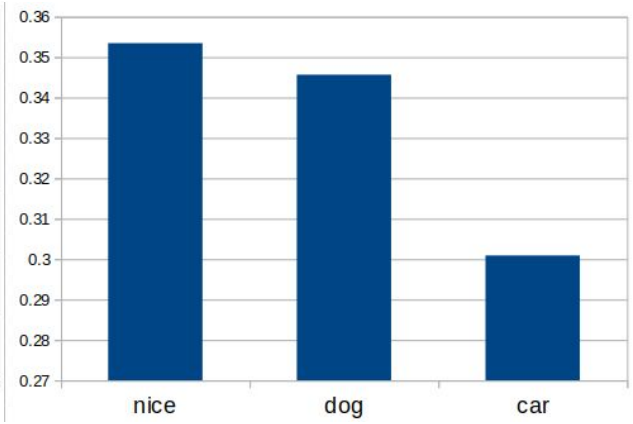
$$P_i = \frac{e^{\frac{y_i}{T}}}{\sum_{k=1}^{n} e^{\frac{y_k}{T}}}$$



Temperatura = 0.5



Temperatura = 1



Temperatura = 10

Text prediction

Link al Colab

LINK

Desafio

Utilizar otro dataset y poner en práctica la generación de secuencias con las estrategias presentadas.