Procesamiento de lenguaje natural (NLP)

Luis R. Furlán

Octubre de 2021

NLP

- Se verá como trabajar con datos textuales en conjunto con “deep learning”
- Esta es una extensión natural de las series de tiempo y las redes neuronales recurrentes

NLP

- Se creará una red neuronal que genere texto nuevo, basado en un cuerpo de datos textuales
- Vale la pena repasar el artículo “The unreasonable effectiveness of RNNs” de Andrej Apathy ya que, en esencia el modelo está basado en un proyecto descrito en ese artículo

NLP

¿Qué vamos a hacer, y cómo funcionará?

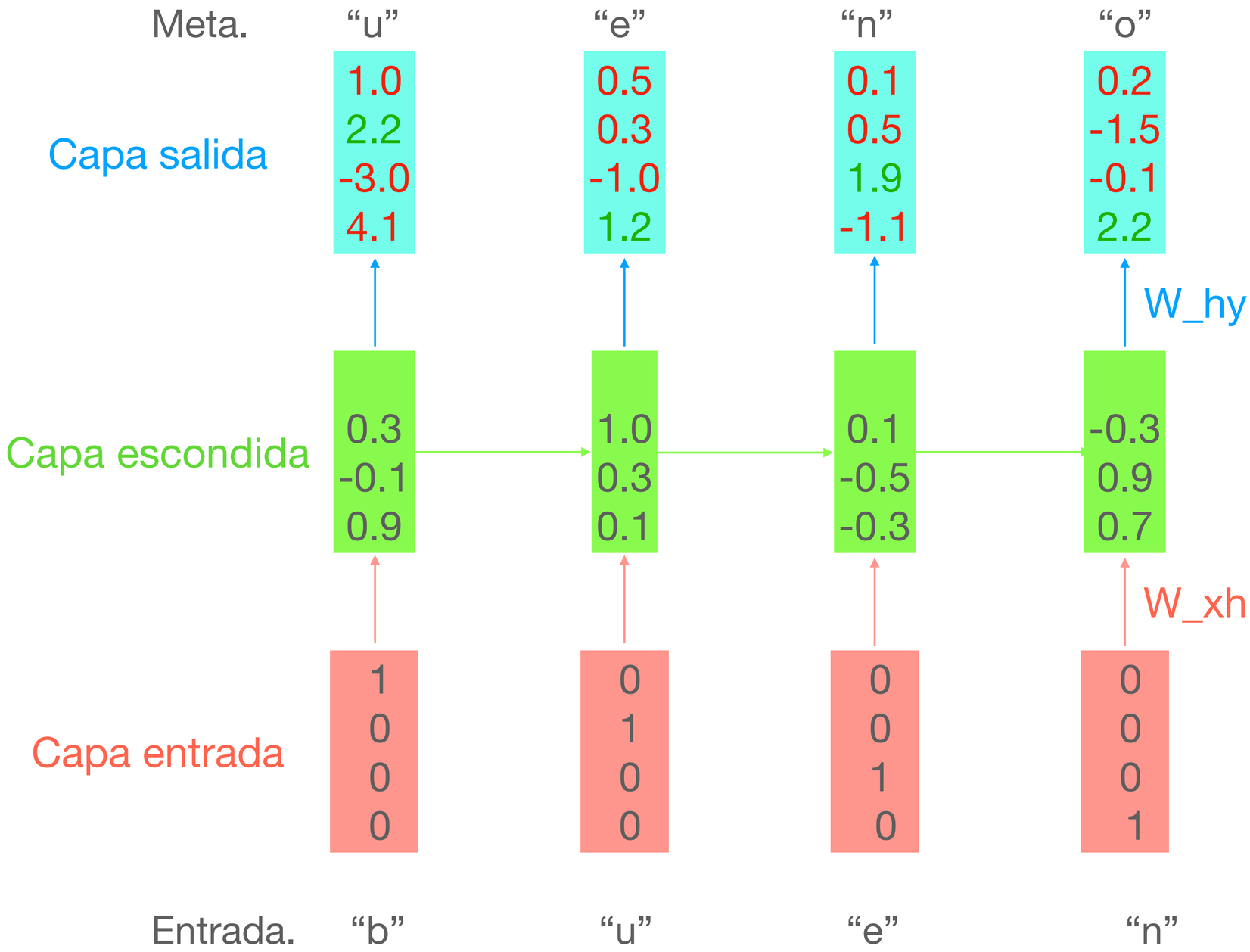
- Dada una secuencia de caracteres (string) de entrada, predecir la secuencia desplazada hacia adelante en un caracter:
  - [“b”, “u”, “e”, “n”]
  - [“u”, “e”, “n”, “o”]

NLP

¿Qué vamos a hacer, y cómo funcionará?

- La RNN basada en caracteres “aprenderá” la estructura del texto.
- Para el ejercicio, se usarán las obras de William Shakespeare.
- Podrá verse que la red claramente “aprenderá” sobre la estructura y esparcimiento de obras de teatro, simplemente a un nivel de caracteres!

NLP - el modelo



NLP

Paso 1 - Leer datos textuales

- Se usarán instrucciones básicas de Python para leer un cuerpo de texto como cadenas
- Nota: Para que esto dé un resultado realista se requieren al menos 1 millón de caracteres

NLP

Paso 2 - Procesamiento de texto y vectorización

- La red neuronal no puede “digerir” cadenas puras, Por esta razón, es necesario codificarlo todo a enteros, por ejemplo
  - A: 1
  - B: 2
  - C: 3
  - ?: 55

NLP

Paso 3 - Crear tandas

- Se usará la clase, de conjuntos de datos, de Tensorflow para crear fácilmente tandas de secuencias de texto
  - [“b”, “u”, “e”, “n”, “o”, “ “, “e”]
  - [“u”, “e”, “n”, “o”, “ “, “e”, “s”]

NLP

Paso 3 - Crear tandas

- Se usarán longitudes de secuencia que sean lo suficientemente largas para capturar la estructura y palabras previas
- Pero no tan largas que causen que las secuencias sean solo ruido histórico que no sea relevante a la secuencia desplazada un caracter hacia adelante

NLP

Paso 4- Crear el modelo

- Se usarán 3 capas
  - Incrustamiento
  - GRU
  - Denso

NLP

Paso 4- Crear el modelo

- La capa de incrustamiento convierte enteros positivos (índices) a vectores densos de un tamaño fijo, por ejemplo:
  - [[4], [20]] ——> [[0.25, 0.1, 0.3], [0.6, -0.2, 0.9]]
- Le queda al usuario definir el número de dimensiones de incrustamiento

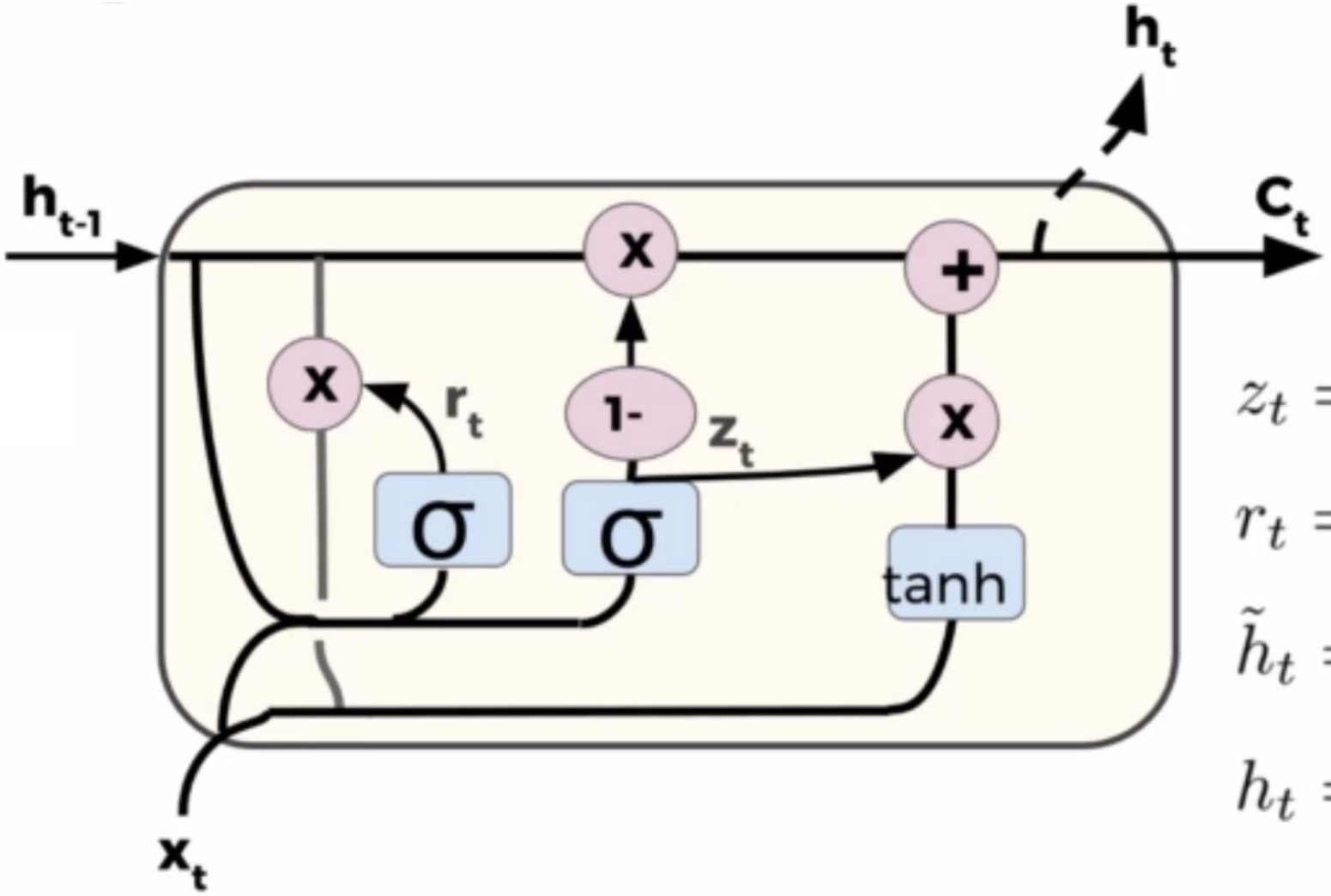
NLP

Paso 4- Crear el modelo

- La GRU (Gated Recurrent Unit) es una tipo especial de una neurona recurrente
- La GRU es como un LSTM (Long-Short Term Memory) con compuerta de olvido pero con menos parámetros que una LSTM ya que no tiene una compuerta de salida

RNN

LSTM - Hay variantes de esto, por ejemplo la GRU (Gated Recurrent Unit)



Combina las compuertas de olvido y entrada para formar una de actualización

$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

NLP

Paso 4- Crear el modelo

- Se usará una capa Densa con una neurona por caracter
- Recordar que las etiquetas (los valores) de los caracteres han sido “one hot encoded” para que la salida sea una probabilidad por caracter
- Al tener una probabilidad por caracter, se puede “jugar más” con el concepto de “temperatura”
  - Seleccionar caracteres menos probables con menor o mayor frecuencia
  - Hablaremos más de esto al ir viendo el código

NLP

Paso 5 - Entrenar el modelo

- Se armarán las tandas y se debe asegurar de realizar el “one hot encoding” a los caracteres

NLP

Paso 5 - Generar el texto nuevo

- Guardar los pesos finales del modelo
- Cargar los pesos guardados con un tamaño de tanda diferente para poder alimentar el modelo con ejemplos individuales
- Esto es similar a cómo se generaron los pronósticos con las Series de Tiempo

¡Vamos al código!