Deep Learning en NLP

...

KeepCoding - Bootcamp de Big Data & Machine Learning

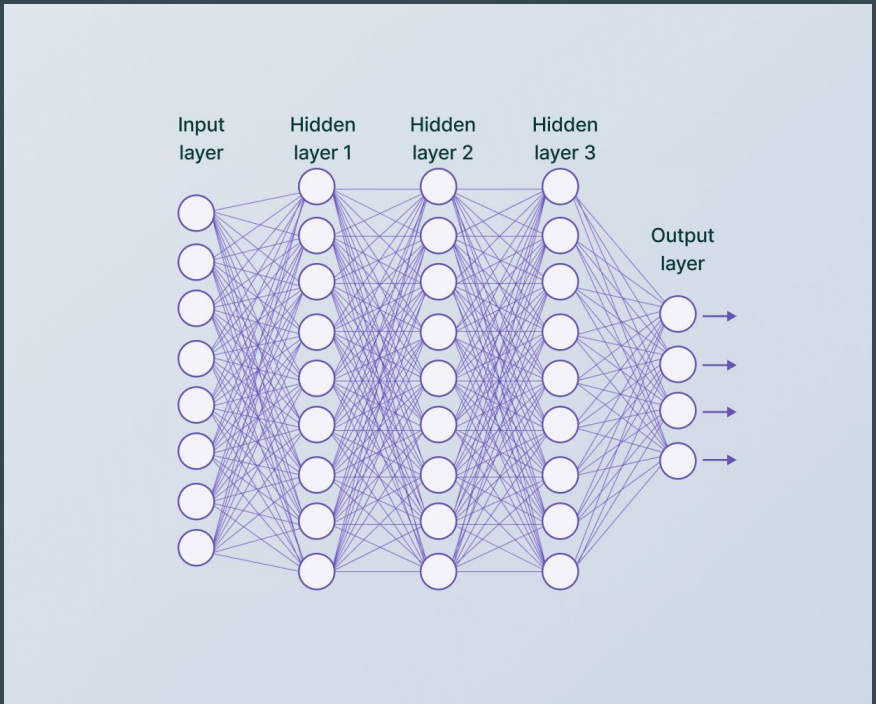
Introducción

- Necesidad de entender cada palabra en base a lo entendido en palabras previas.

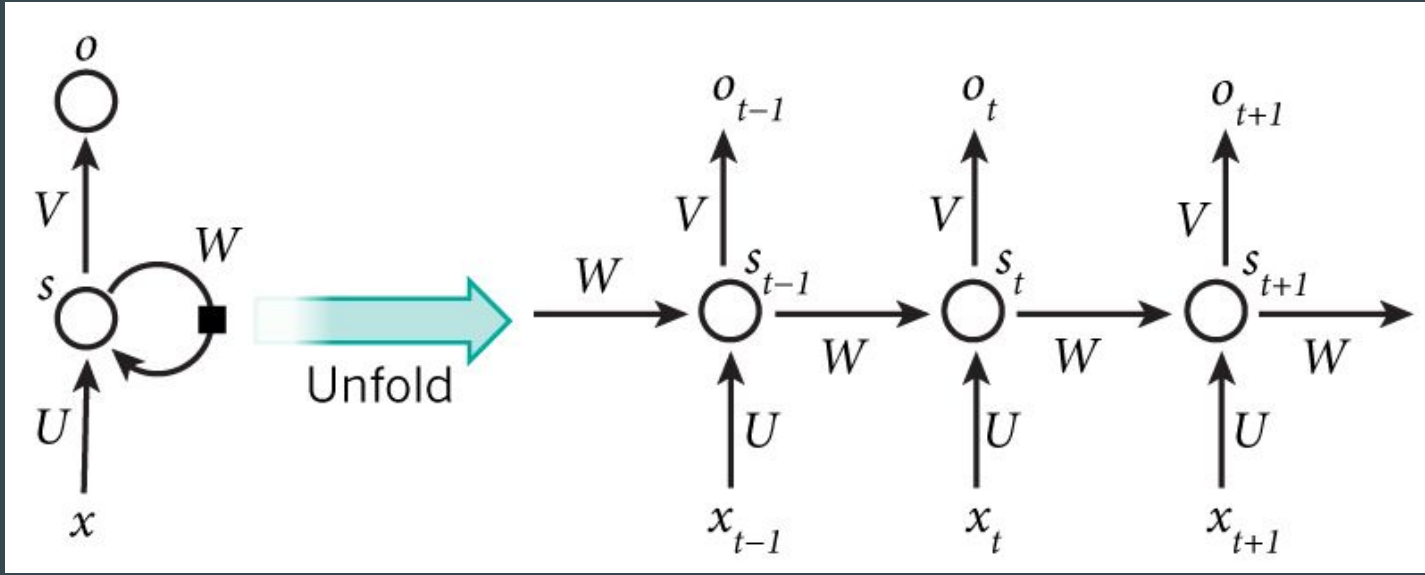
- El entendimiento tiene persistencia por lo que se tiene que capturar ese valor del texto.

- En redes neuronales tradicionales todas las entradas y las salidas son independientes entre sí. Mal planteamiento para NLP.

CNN vs RNN


Input layer
Hidden layer 1
Hidden layer 2
Hidden layer 3
Output layer

CNN vs RNN



RNN

- Unfolding es escribir la NN para la secuencia completa. Por ejemplo, si una secuencia tiene 4 palabras la red hará unfolding a 4 capas de la NN.

- S en el tiempo t es la memoria de la red en capa que se encarga de capturar lo que ha pasado en pasos previos.

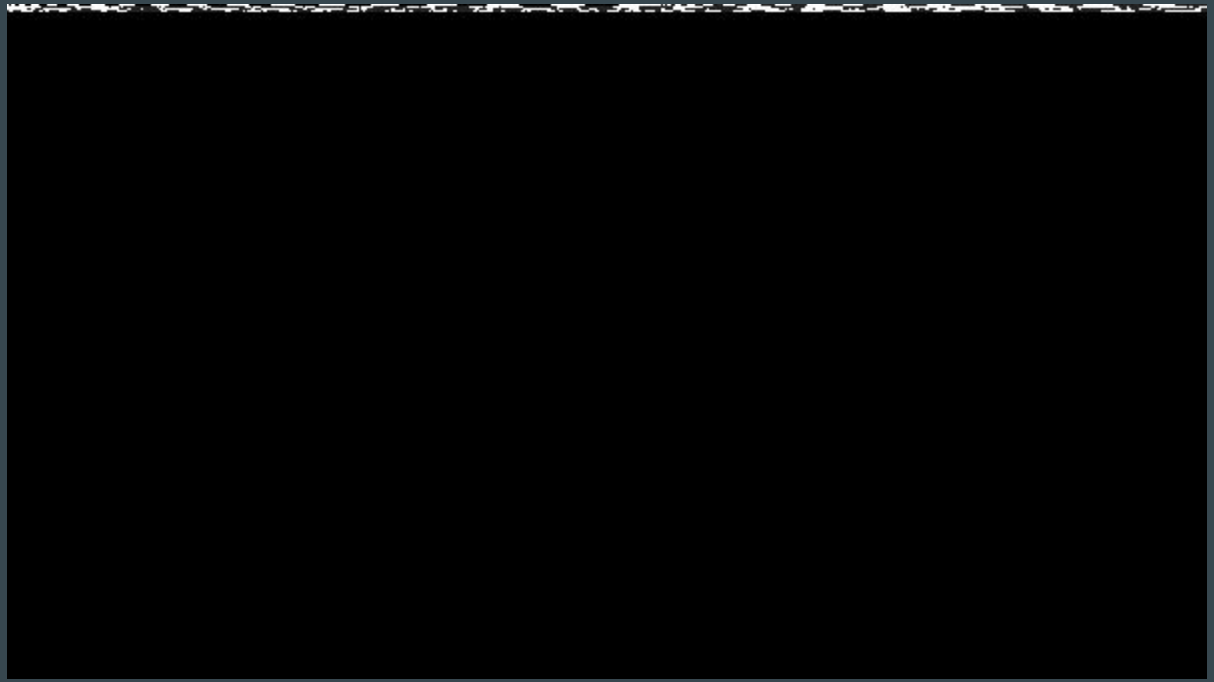
- Una NN tradicional usa diferentes parámetros para capa capa de la red. En las RNN todos los parámetros son compartidos por todas las capas.

RNN

- Compartir los parámetros en todas las capas hace que realicemos la misma tarea con diferentes entradas, reduciendo el número de parámetros a aprender.

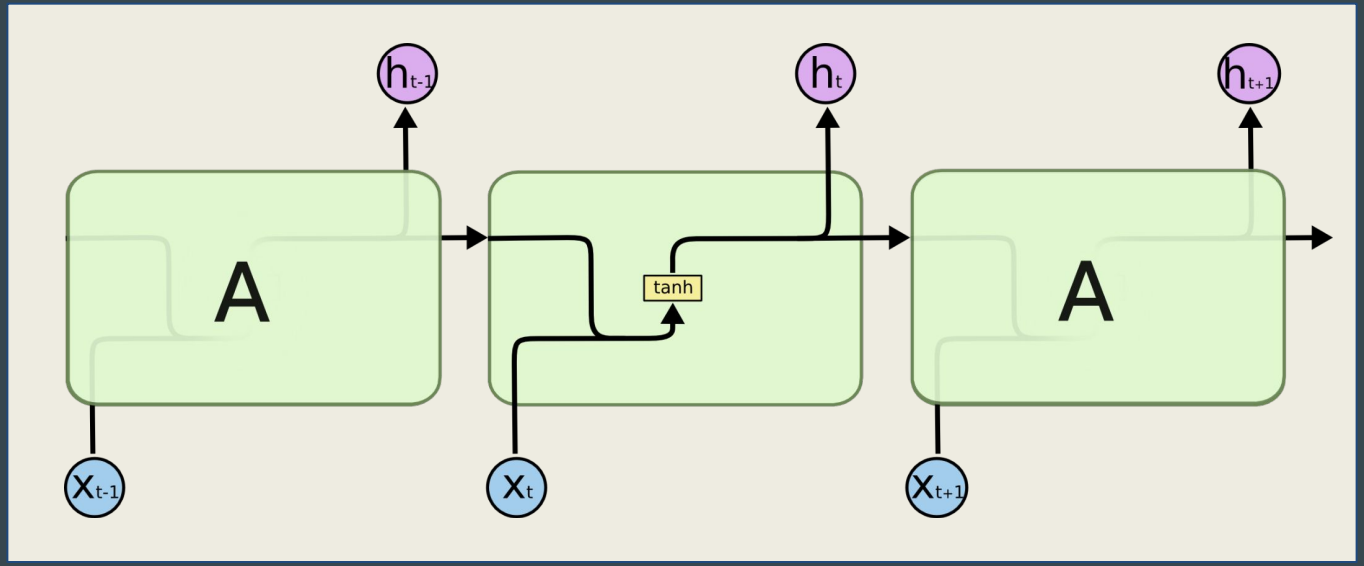
- Para usar RNN es necesario preprocesar el texto en una forma entendible para la red. Tokenización junto con One Hot Encoding / Word Embeddings.

RNN



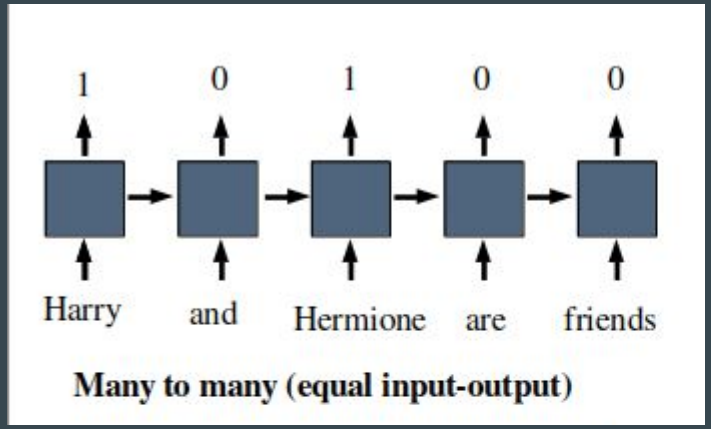
RNN

RNN



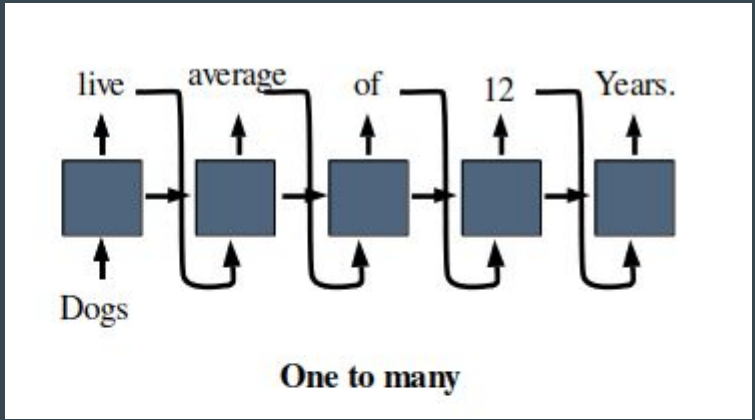
Tipos de RNN

- Propagación de las activaciones por toda la red antes de generar una predicción por capa.
- Muy útil para generación de NER.



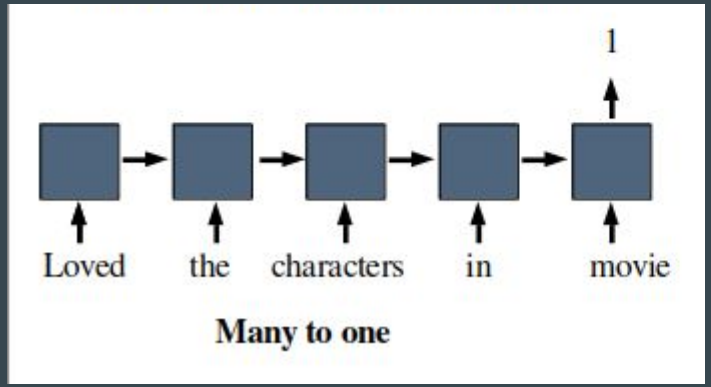
Tipos de RNN

- En cada propagación por una capa genera una predicción y su activación se pasa a la siguiente.
- Muy útil para NLG (Natural Language Generation)



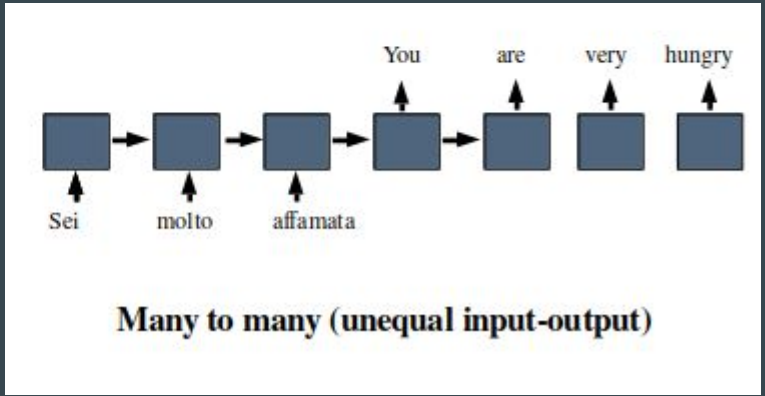
Tipos de RNN

- Propagación de las activaciones por toda la red antes de generar una única predicción.
- Muy útil para análisis del sentimiento.



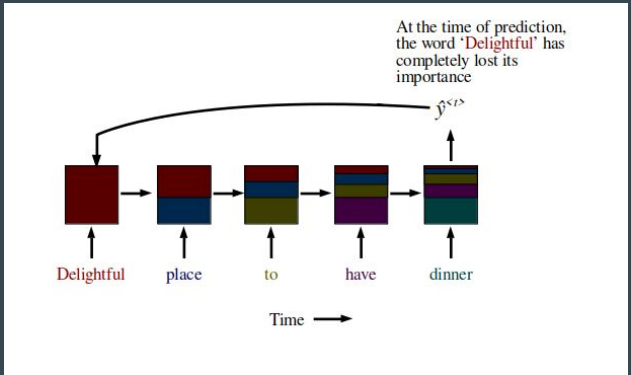
Tipos de RNN

- Propagación de las activaciones por toda la red antes de generar una predicción por capa de salida.
- Número diferente de salidas al de entrada o viceversa
- Muy útil para generación de traductores de idioma.



Vanishing and Exploding Gradient Problem

- Las redes RNN tienen solo memoria a corto plazo.
- Si usamos una gran cantidad de capas en un RNN podemos sufrir este problema.
- Los gradientes se desvanecen (valores de casi 0) cuando se hace Backpropagation hacia las capas iniciales.



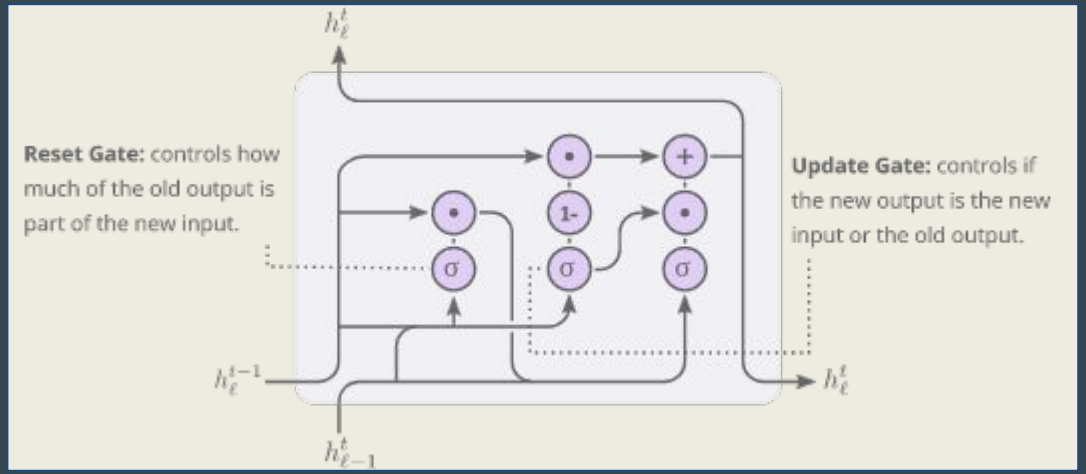
GRU Networks

- Incorporación de Update Gate y Reset Gate para resolver Vanishing Gradient problem.

- Update Gate: ayuda a decidir cuánta información de capas anteriores debe pasarse a capas futuras.

- Reset Gate: ayuda a decidir cuánta información de capas anteriores olvidar.

GRU Networks



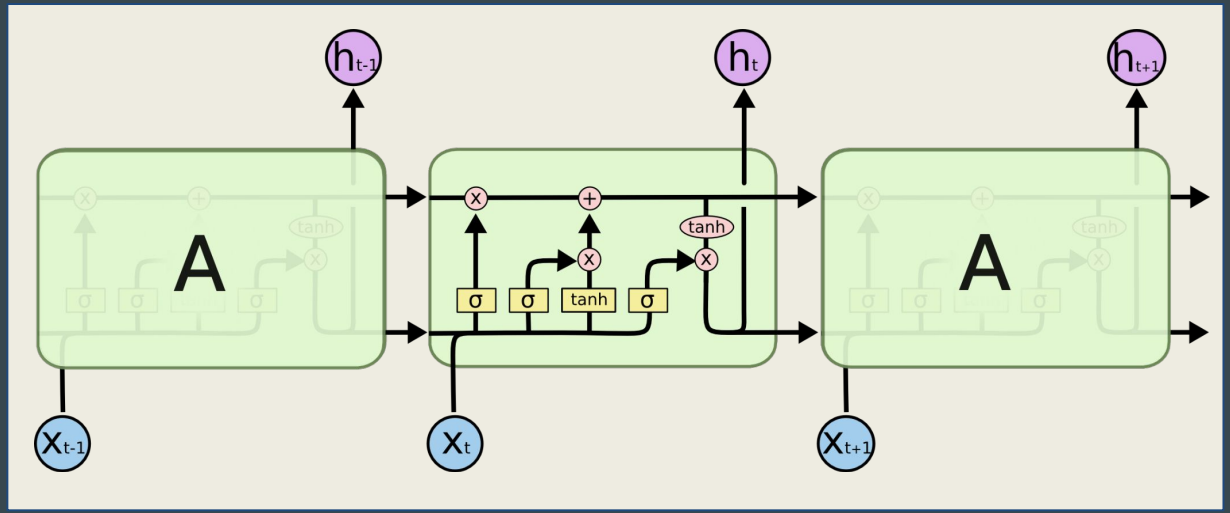
LSTM Networks

- Hace el uso de Gates como en las GRU.

- No propaga la información a posteriores capas sino que la almacena en una unidad de memoria integrada en cada capa.

- Tienen un diseño más complejo pues hace uso de más Gates.

LSTM Networks



GRU vs LSTM

- Las LSTM Son capaces de almacenar información a más largo plazo que GRU.

- Las GRU entrenan más rápido que las LSTM y tienen un rendimiento más alto con menor cantidad de datos.

- En tareas dependientes de largas relaciones temporales las LSTM tienen mejor rendimiento.

¡Vamos al lío!