Seminario Permanente de Formación en Inteligencia Artificial Aplicada a la Defensa

Febrero, 2021

Dee Learning IV: Redes Recurrentes y PLN. Casos de Estudio.

Eugenio Martínez Cámara

Instituto Andaluz de Investigación en Data Science and Computational Intelligence (DaSCI)

Dpto. Ciencias de la Computación e I.A.

Universidad de Granada

emcamara@decsai.ugr.es

http://sci2s.ugr.es

DaSCI

UNIVERSIDAD DE GRANADA

Contenidos

- ¿Qué es una red neuronal?
- Aprendizaje basado en gradiente.
- Propagación del gradiente
  - *Back-propagation*.
- Redes Neuronales Recurrentes (RNN)
  - RNN. Procesamiento de secuencias.
  - RNN. Bidireccionales.
  - RNN. Arquitecturas de secuencia-secuencia.
  - RNN. Multicapa o profundas
  - RNN. Representación de pilas.
  - RNN. Redes recursivas.
  - RNN. Dependencias lejanas.
    - RNN. Dependencias lejanas. Soluciones.
    - RNN. basadas en puertas: LSTM.
    - RNN. basadas en puertas: BiLSTM.
    - RNN. Mecanismo de Atención.

Contenidos

- ¿Cómo diseñar una red neuronal?
  - ¿Cómo diseñar una red neuronal? A tener en cuenta.
- Caso Práctico
- Bibliografía

¿QUÉ ES UNA RED NEURONAL?

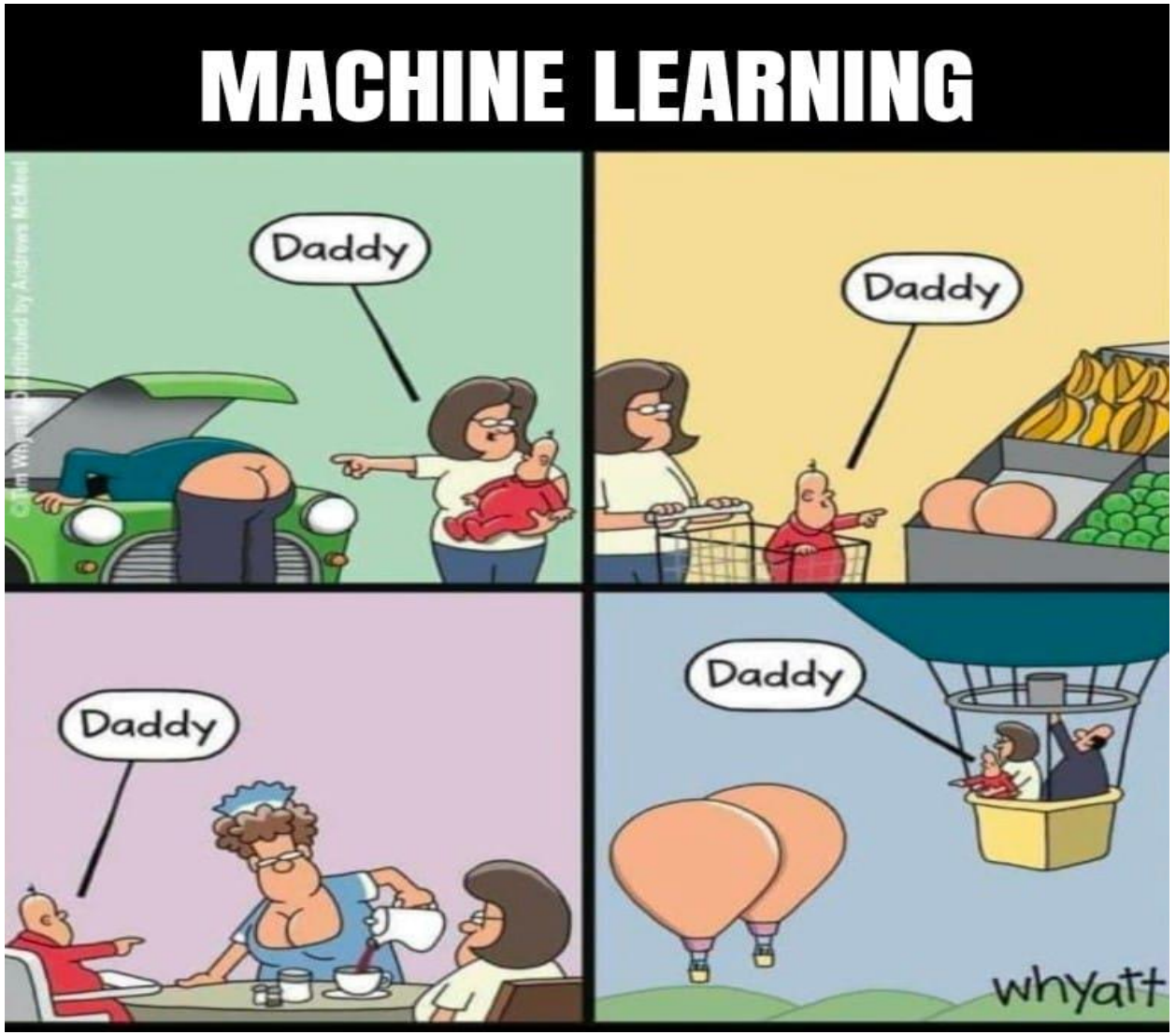
¿Qué es una red neuronal?

Es un tipo de algoritmo de aprendizaje propio de la inteligencia artificial.

¿Qué es una red neuronal?

¿Cuál es el objetivo de un algoritmo de aprendizaje?

¿Qué es una red neuronal?

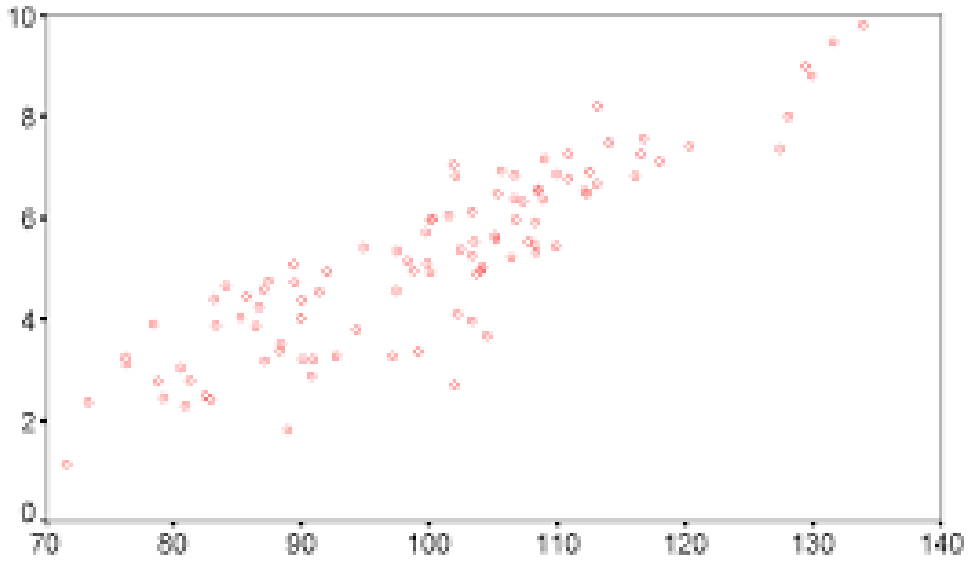


¿Qué es una red neuronal?

- Conjunto de datos: x

$$y = f^*(x)$$

- Conjunto de clases: y

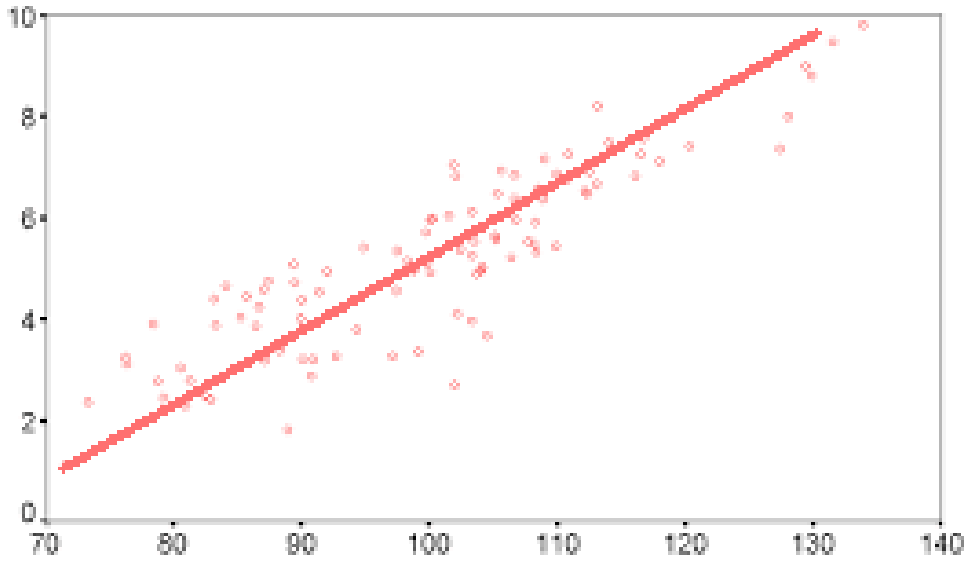


¿Qué es una red neuronal?

- Conjunto de datos: x

$$y = f(x)$$

- Conjunto de clases: y



¿Qué es una red neuronal?

- Los algoritmos de aprendizaje automático suelen ser de optimización, que tratan de encontrar aquella función que mejor se ajusta al conjunto de datos de entrenamiento.

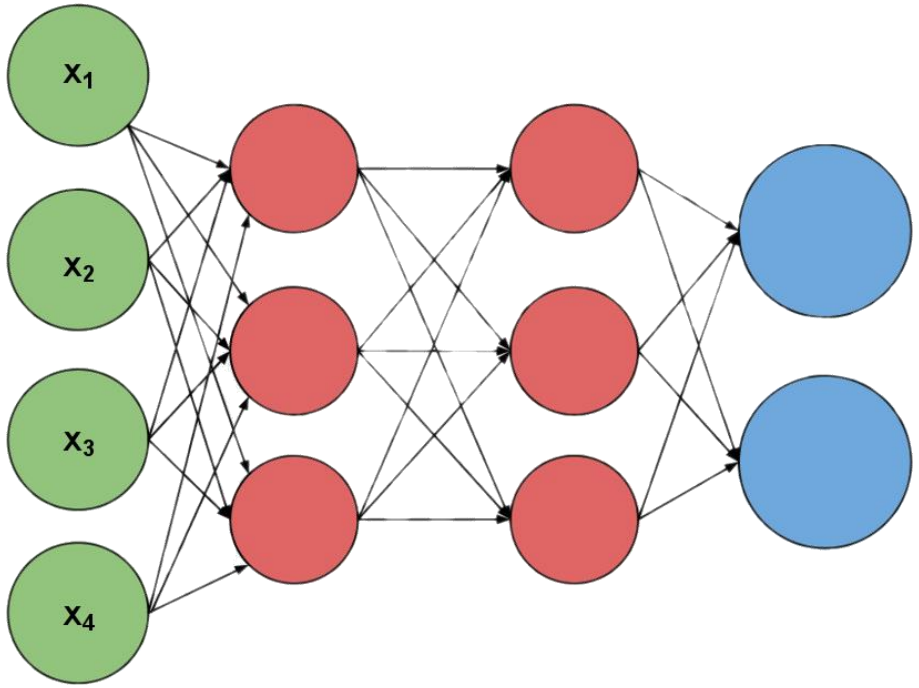
- Suposición: los datos “nuevos” o a clasificar tienen un comportamiento similar a los de entrenamiento.

- Sobre esta asunción se construye el ideal de todo sistema de aprendizaje automático: que sea **generalizable**.

¿Qué es una red neuronal?

- Las redes neuronales (RN) son algoritmos de optimización iterativa.
- Usan un conjunto de parámetros (θ) para encontrar la función que se ajusta de una manera generalizable al conjunto de datos de entrenamiento.
- Matemáticamente una red neuronal es:

$$\boldsymbol{y = f(x, \theta)}$$



¿Qué es una red neuronal?

- Una RN entonces es una función sobreyectiva: cada elemento de X se le asigna una elemento (clase) de Y.
- ¿Por qué *red*? Porque entre la entrada y la salida hay capas interconectadas, de las que se desconoce su salida → **Capas Ocultas**.
- ¿Qué son las capas ocultas? Funciones del tipo $\boldsymbol{y = f(x, \theta)}$**.**
- El nº de capas ocultas determina la profundidad de la red.
- *Deep Learning* →Aprendizaje profundo → muchas capas ocultas.
- Por tanto, una RN es una composición de funciones.

$$\boldsymbol{y = f(x, \theta) = f^{(n)} \dots \left(f^{(3)} \left(f^{(2)} \left(f^{(1)}(x, \theta) \right) \right) \right)}$$

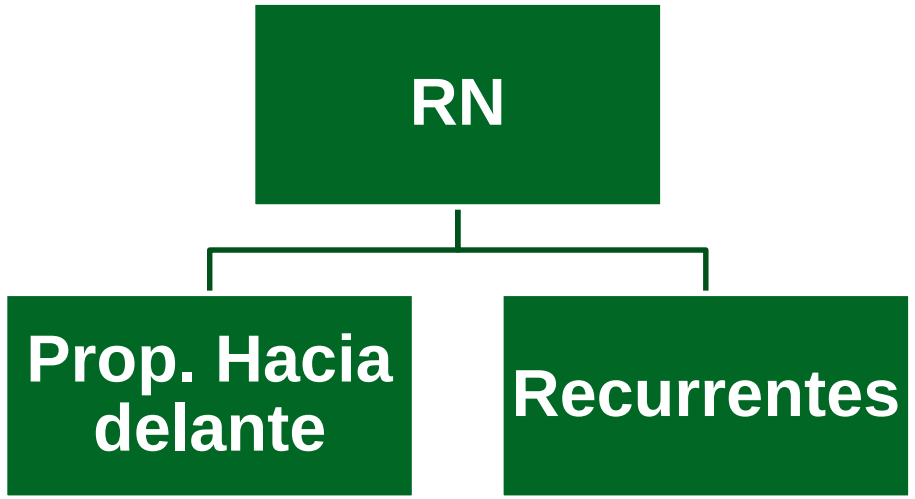
¿Qué es una red neuronal?

- ¿Por qué se llaman neuronales? Porque están inspiradas en la neurociencia.
- Cada capa oculta es un vector/matriz/tensor de elementos, cuya dimensión define la anchura del modelo.
- Cada elemento/valor/unidad se considera una neurona.
- Cada capa es un conjunto de unidades/neuronas que se modifican en paralelo.
- La operación de modificar y aplicar una función de activación en cada capa se asemeja al modo de actuar de las neuronas biológicas.

¿Qué es una red neuronal?

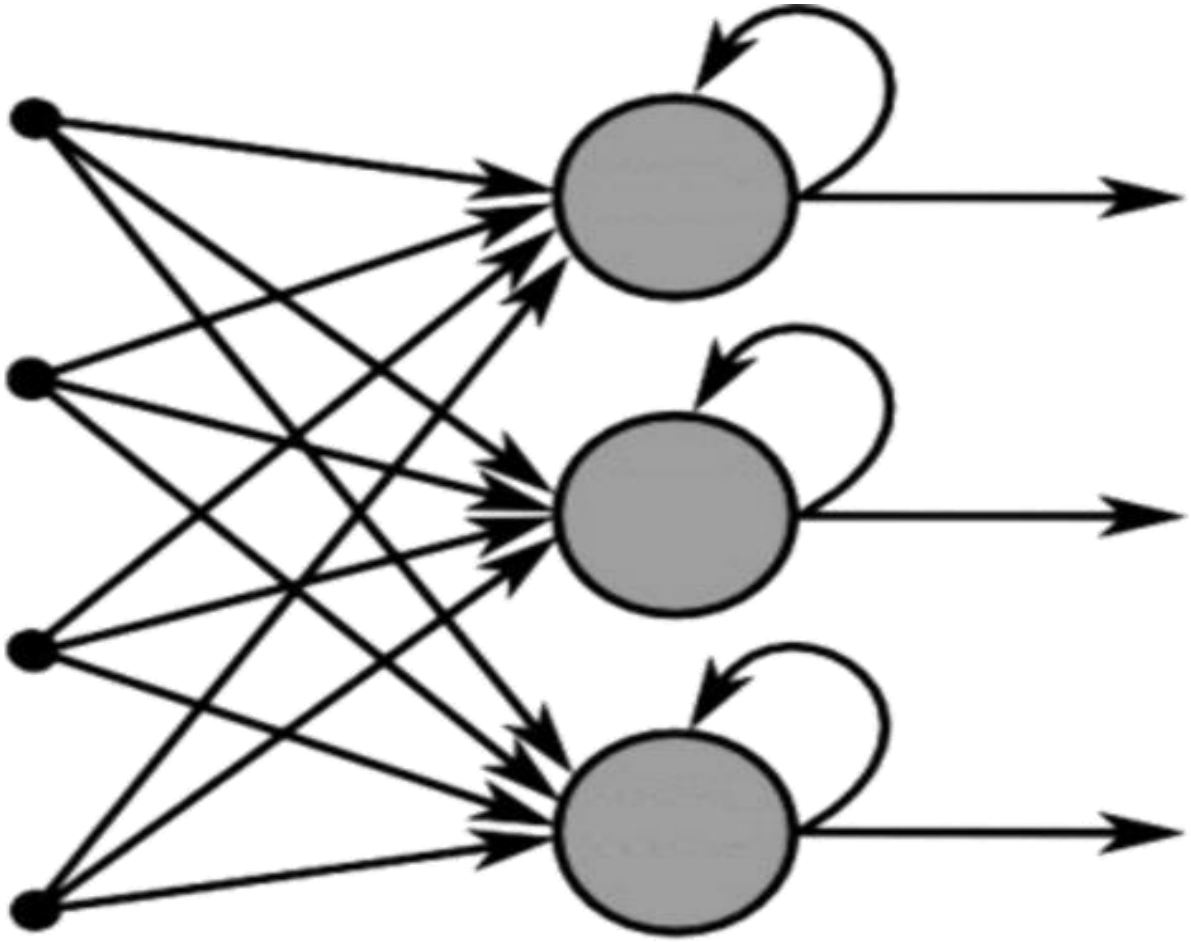
- **Propagación hacia delante (*feedforward*)**:
  Flujo de la información desde la entrada hasta la salida.
  La salida nunca vuelve a la entrada.
  Importante: no confundir la actualización hacia atrás de parámetros con una propagación hacia atrás de la información.

- **Recurrentes**: Flujo de información recurrente desde la entrada hasta la salida, es decir, **la salida vuelve a la entrada**.

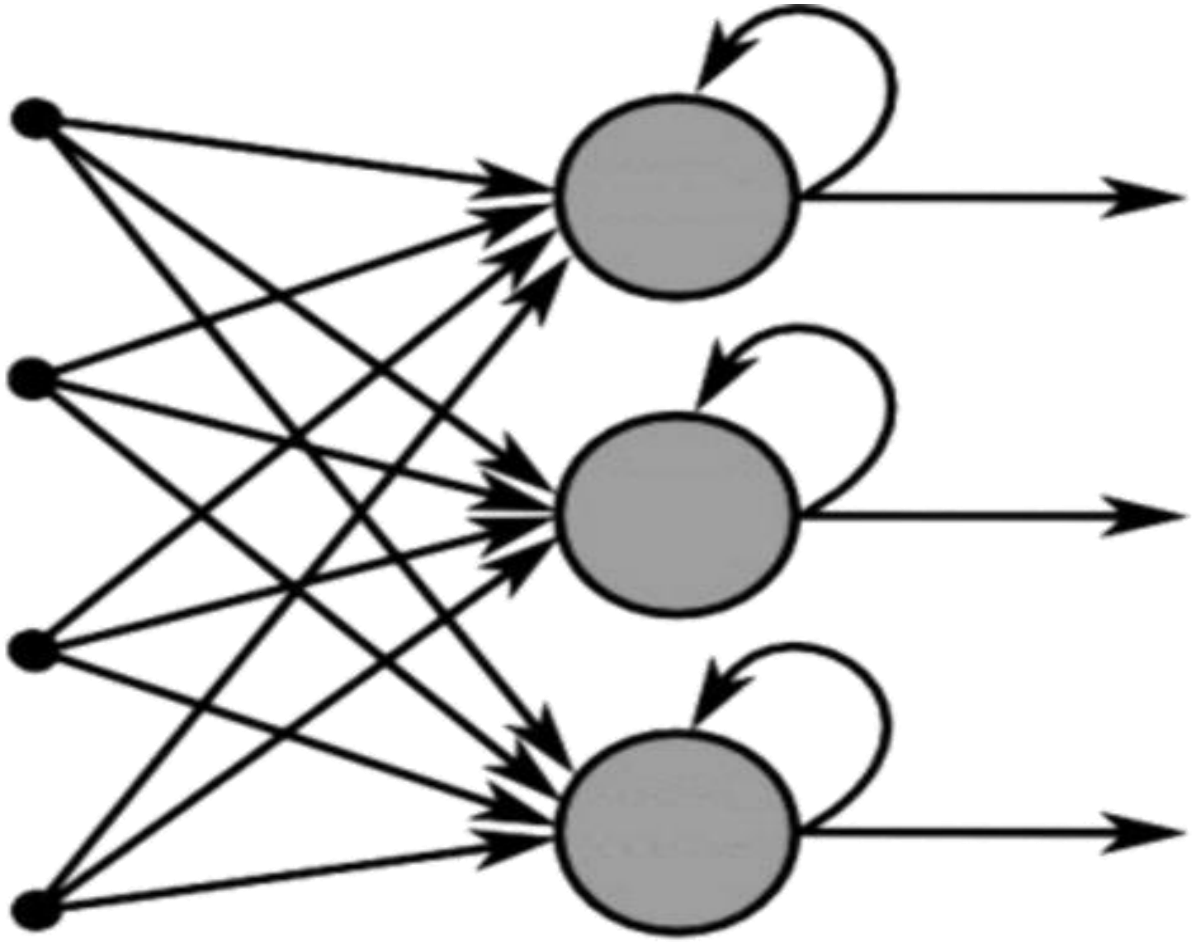


¿Qué es una red neuronal?

Red de Propagación hacia delante

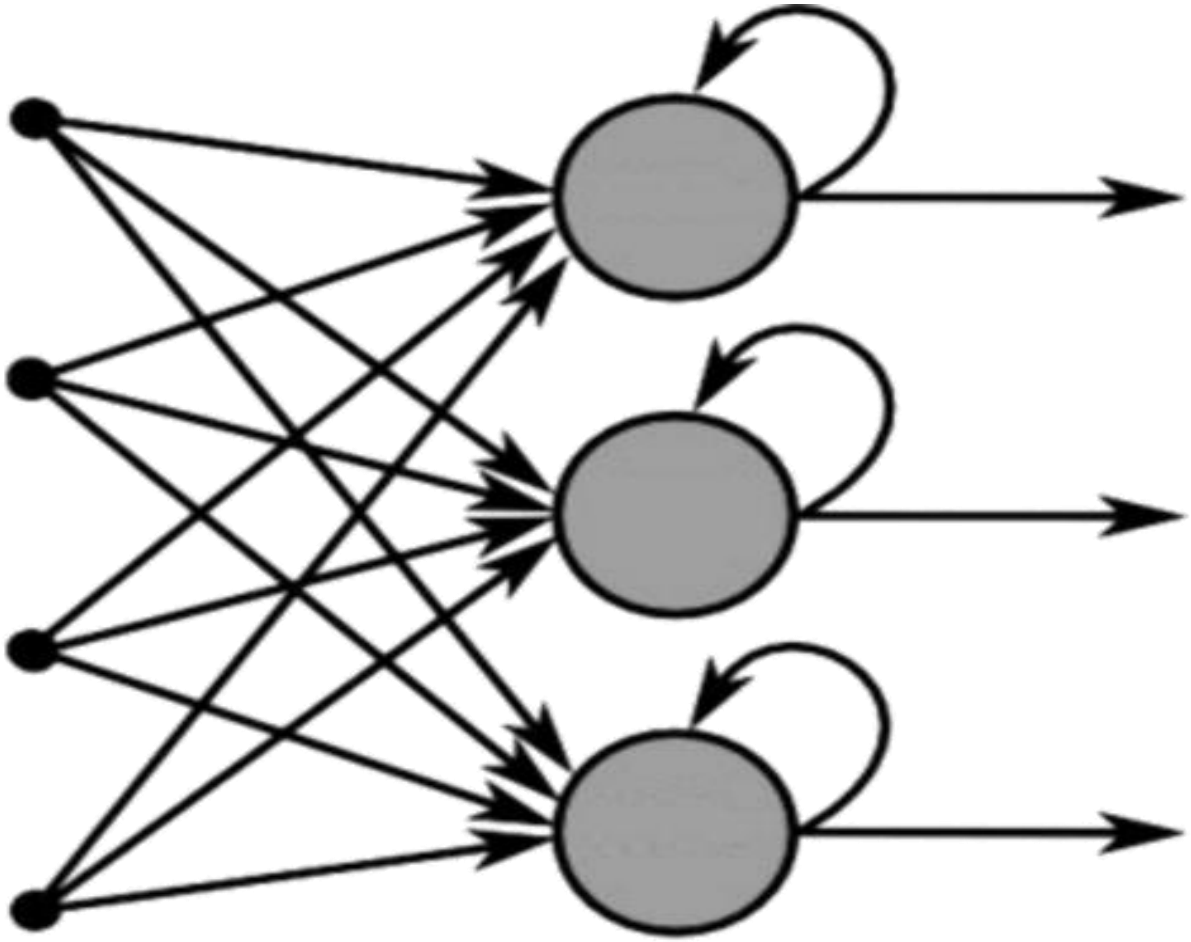


Red Neuronal Recurrente



¿Qué es una red neuronal?

Red de Propagación hacia delante



Red Neuronal Recurrente

APRENDIZAJE BASADO EN GRADIENTE

Aprendizaje basado en gradiente

- Aprendizaje: Encontrar la función (f) que aproxime la función que describe los datos de entrenamiento (f^*).

$$y = f^*(x) \approx f(x)$$

- Esto se realiza apoyando el aprendizaje en un conjunto de parámetros.

$$y = f^*(x) \approx f(x, \theta)$$

- Los parámetros se actualizan iterativamente hasta que la función de coste sea mínima.

- ¿Cómo dirigir la actualización de θ?
  - Algoritmo de *back-propagation* dirigido por el gradiente.

Aprendizaje basado en gradiente

- Consiste en encontrar el mínimo de la función de coste.
- ¿Se puede aplicar a otros algoritmos como SVM o Reg. Lineal?
- Sí, se puede, ¿entonces por qué RN?
- Por que mientras que SVM (lineal) y otros algoritmos de aprendizaje automático aproximan funciones lineales, las RN aproxima **funciones no lineales**.
- Las funciones no lineales se ajustan a un mayor nº de problemas del mundo real.

PROPAGACIÓN DEL GRADIENTE

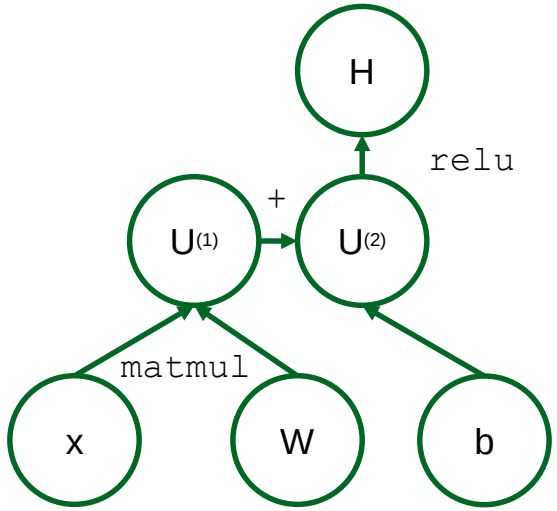
Propagación del gradiente

- La propagación hacia delante simplemente genera una salida a partir de una entrada.
- ¿Cómo aprende la red? Modificando los parámetros θ.
- ¿Qué cantidad se debe modificar? Mediante el cálculo del gradiente de la función de coste.
- ¿Cómo modificar todos los θ de todos las variables de la red? Con el algoritmo de propagación hacia atrás del gradiente: ***Back-propagation***.
- *Back-propagation* no es exclusivo de las redes neuronales.

Propagación del gradiente. *Back-propagation.*

Fundamentos:

- Grafo de computacional: Abstracción para representar la relación entre variables mediante operaciones que se produce en una red neuronal.
  - Nodos: Variables.
  - Aristas: Operaciones que se aplican sobre un nodo (variable) y genera como salida otro nodo (variable).



UNIVERSIDAD DE GRANADA
DaSCI

Propagación del gradiente. *Back-propagation.*

Fundamentos:

- Cálculo de derivadas en cadena

$$y = g(x);\ z = f\big(g(x)\big) = f(y)$$

$$\frac{dz}{dx} = \frac{dz}{dy}\frac{dy}{dx}$$

- Bachiller: $\Big(f\big(g(x)\big)\Big)' = f'\big(\mathrm{g(x)}\big) * \mathrm{g'(x)}$

- Derivación en cadena a nivel vectorial/matricial/tensonrial, obteniendo los jacobianos (1ª derivada) de dichos elementos.

$$\nabla_x z = \frac{dy}{dx}^T \ \nabla_y z$$

Propagación del gradiente. *Back-propagation.*

- Computacionalmente, el grafo y la aplicación de las derivadas requiere una representación: **representación simbólica**. Tipos:

- **Símbolos-Números**: Sustitución de las variables por los números que toman. De esta manera ya se calculan las derivadas directamente.
  Librerías: Torch y Caffe.

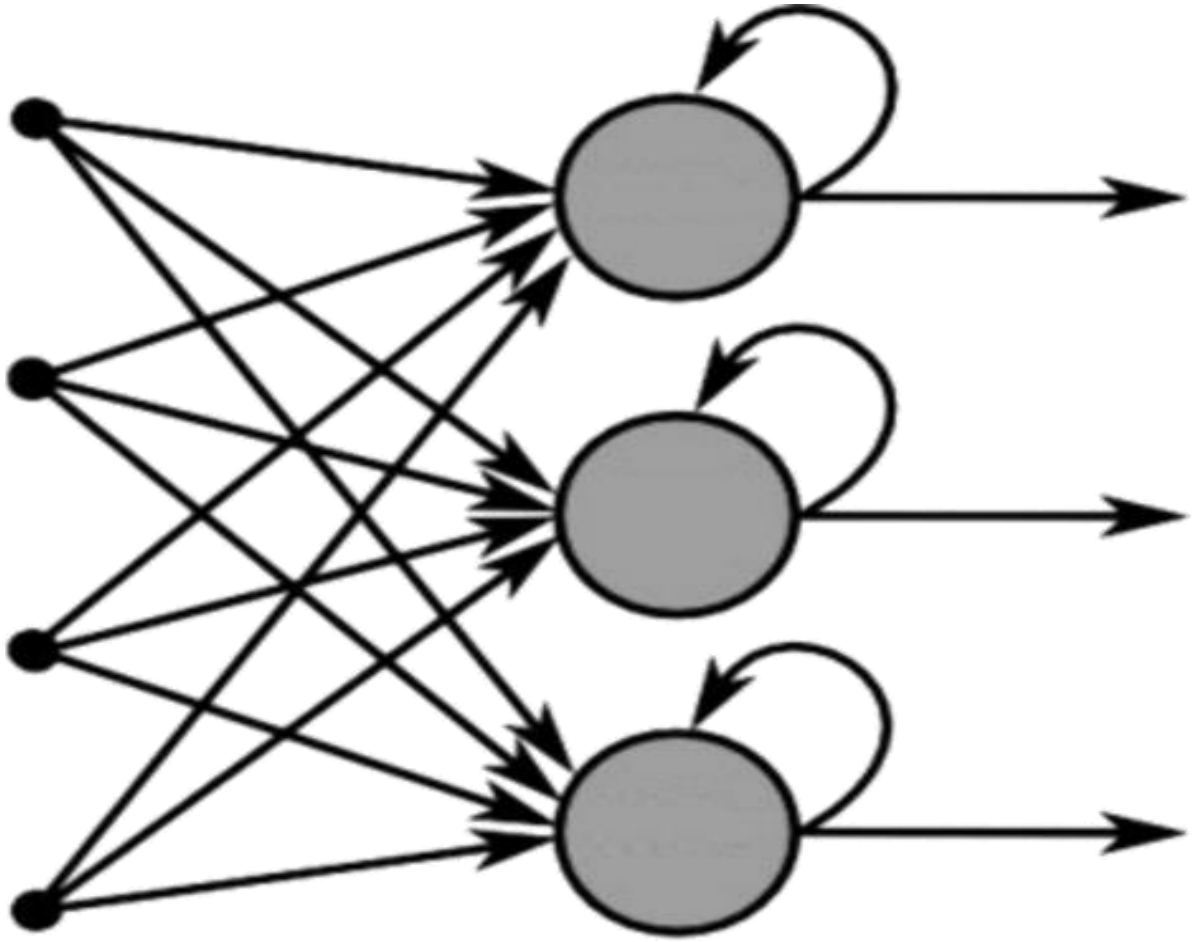
- **Símbolos-símbolos**: Se genera otro grafo simbólico que representa las derivadas a aplicar al grafo original:
  Librerías: TensorFlow y Theano.

RNN. PROCESAMIENTO DE SECUENCIAS DE DATOS

RNN. Procesamiento de secuencias de datos

Red de Propagación hacia delante

Red Neuronal Recurrente



RNN. Procesamiento de secuencias de datos

- Existen problemas de clasificación y detección que simplemente requieren del reconocimiento de un patrón.
- La presencia de dicho patrón siempre se asocia a una misma clase.

RNN. Procesamiento de secuencias de datos

- MonuMAI

RNN. Procesamiento de secuencias de datos

- Hay problemas que lo relevante es una secuencia de patrones. P. Ej. Un vídeo, se trata de una secuencia de imágenes.
- Puede que un mismo patrón no esté asociado a una misma clase o una clase a distintos patrones, por:
  - Dominio. Análisis de opiniones. P. Ej.: ¿conducción predecible = argumento predecible?
  - Semántica. Análisis de opiniones. P. Ej.: No me gusta el pescado = Me disgusta el pescado.
  - Sintaxis: Yo visité Jaén en 2018 = En 2018, yo visité Jaén.
  - Léxico-Morfológico: Entidades. Vaya banco de peces hay en el río. El Banco de España emite moneda. ¿banco de peces = Banco de España?
  - Posición. Anáfora. Elena compró tomates ayer. Hoy ella se los comió y les encantó. → ¿Quién es ella?
  - Predicción de palabras. Modelos de lenguaje. ¿Después de “La” qué es más probable la palabra “moto” o el nombre propio “Antonia”?

- Son relevantes todos los elementos de la secuencia.

RNN. Procesamiento de secuencias de datos

- Para solucionar los problemas anteriores sería necesario aprender todos los posibles patrones.

- En una secuencia finita de imágenes, se podría aprender los patrones de todas las posibles secuencias, dado que son finitas, pero ¿se podrían clasificar secuencias similares? ¿sería posible obtener un modelo generalizable?

- ¿Sería posible aprender todos los posibles usos del lenguaje de un idioma?

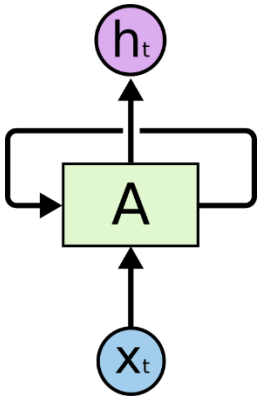
RNN. Procesamiento de secuencias de datos

- En estos problemas, el procesamiento de un elemento depende del procesamiento del anterior.

- Se requieren de RNs que permitan codificar un elemento como el producto de la codificación anterior y la nueva entrada.

- La codificación de la secuencia requeriría que se compartieran los parámetros (W).

- Esto lo permiten las RNN.



RNN. Procesamiento de secuencias de datos

- Las RNN permiten combinar la información pasada (memorizada) con la información actual.
- Para ello se combinan los parámetros y se mantiene una información de estado a través de las capas ocultas (h).
- Matemáticamente:

$$\boldsymbol{h^{(t)} = f(h^{(t-1)}, x^{t};\ \theta)}$$

Donde:

- h son los valores de las capas ocultas.
- x Secuencia de entrada.
- t el instante (índice) de la secuencia.
- θ los parámetros.

RNN. Procesamiento de secuencias de datos

- La información pasada (memorizada) se va olvidando conforme se avanza en el procesamiento de la secuencia.

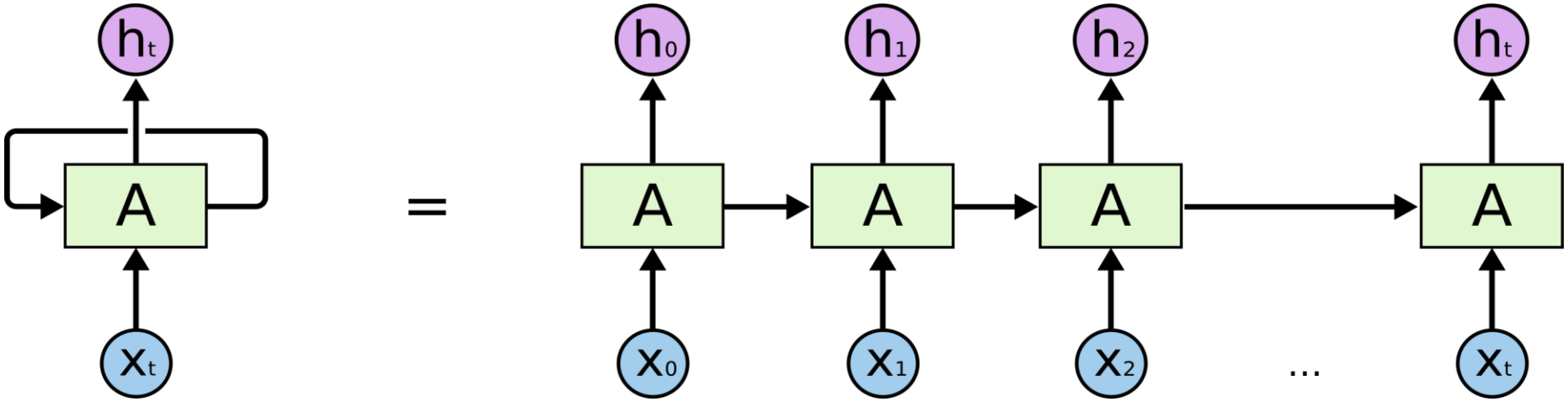
- Las diferentes versiones de RNN se fundamentan en como mejorar y aumentar la capacidad de memoria y de selección de qué memorizar. Ya lo veremos.

RNN. Procesamiento de secuencias de datos

- La recurrencia de las RNN se puede desarrollar:

$$\boldsymbol{h^{(t)} = f\left(h^{(t-1)}, x^{t};\, \theta\right) = g^{(t)}\left(x^{(t)}, x^{(t-1)}, x^{(t-2)}, \ldots, x^{(2)}, x^{(1)}\right)}$$

$$h^{(t)} = f\left(f\left(f\left(f\left(h^{(0)}, x^{(0)}; \theta\right), x^{(1)}\right), x^{(2)}\right), \ldots, x^{(t)}\right)$$



UNIVERSIDAD DE GRANADA
DaSCI

RNN. Procesamiento de secuencias de datos

- ¿Con esta arquitectura se podrán procesar secuencias independientemente de su longitud? **Pues sí**.

- Ventajas:
  - El modelo aprendido siempre tiene el mismo tamaño, dado que está definido en función de un conjunto de transiciones de un estado a otro. Por tanto, no importa el tamaño de la secuencia.
  - Se puede usar la misma función de transición (f) con los mismos parámetros en cada transición.

- De esta manera se puede usar el mismo modelo para aprender secuencias de cualquier tipo de longitud.

- No sería necesario definir un modelo para longitudes distintas de secuencias.

RNN. Procesamiento de secuencias de datos

RNN. Procesamiento de secuencias de datos

Ej. Dos opiniones sobre un hotel.

1. Los árabes y después los cristianos sabían lo que se hacían. No hay mejor vista de Jaén que la que ofrecen los miradores del castillo de Santa Catalina y del Parador, que forman un conjunto arquitectónico muy armónico. Desde luego, debes de ser un amante de la paz y de la tranquilidad porque no puedes estar bajando a la capital a tomar un café a todas horas. Tanto la ubicación, como los largos pasillos y las salas comunitarias, mantienen ese aire inconfundible de los paradores que, en cierta medida, te aísla de lo cotidiano y te permiten recrearte algo mas contigo mismo. Todo lo demás, está en línea con lo que se espera de esta excepcional marca española que son Los Paradores, calidad y potenciación de los valores regionales. Totalmente recomendable y si no os alojáis allí, al menos subir a tomar un café, os encantará.

2. Es un lugar excelente para pasar unos días de vacaciones, pero en mi caso fue la noche de mi Bodas de Plata, la habitación donde estuvimos mi marido y yo es una preciosidad, muy amplia y muy grande unas vista espectaculares, una atención buenísima, y el desayuno magnifico. En general todo fue un sueño. por que además no me lo esperaba ya que me la regalaron algunos de los invitados que fueron a la boda, y pase una luna de miel de Bodas de Plata inolvidable. Seguro que algún día repetiré y se lo recomiendo a todos/as que quiera vivir unos momentos inolvidables.

RNN. Procesamiento de secuencias de datos

- Las RNN permiten el procesamiento de datos secuenciales, pero no son suficientes para obtener buenos resultados en clasificación/extracción.

- Son elementos que dentro de redes más complejas ofrecen buenos resultados.

- Deben considerarse como elementos que codifican propiedades/características de la secuencia de entrada de valor para el procesamiento posterior en la red neuronal.

RNN. Procesamiento de secuencias de datos

- ¿Cómo aprende una RNN?
- Adaptando el algoritmo de *Backpropagation* al procesamiento de secuencias.
- La adaptación se llama: *Backpropagation through Time* (Propagación hacia atrás a través del tiempo).
- En el cálculo de las derivadas se tienen que tener en cuenta las distintas variables que intervienen en una RNN.

RNN. Procesamiento de secuencias de datos

- Varias arquitecturas de RNN existen en la literatura atendiendo a las ideas de (1) desarrollo del grafo computacional y (2) compartición de parámetros.
- Se pueden agrupar en:

1. **RNN_1**: Generan una salida en cada transición, y tienen conexiones recurrentes entre sus capas ocultas.
2. **RNN_2**: Generan una salida en cada transición, y tienen conexiones recurrentes entre la salida de la transición anterior y las capas ocultas de la siguiente transición.
3. **RNN_3**: Tienen conexiones recurrentes entre las capas ocultas, que generan una única salida al final del procesamiento de la secuencia completa.

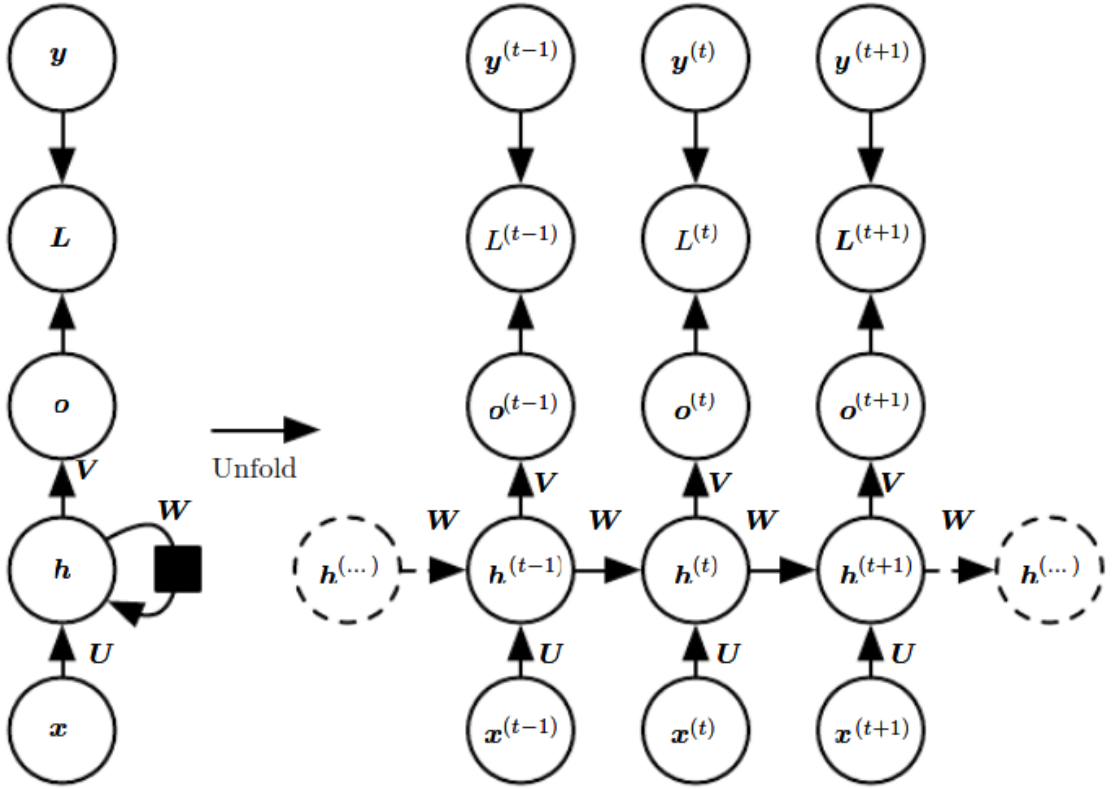
RNN. Procesamiento de secuencias de datos

RNN_1

- En cada instante (t) las capas ocultas ($h(t)$) tienen información del pasado (t-1).

- Matemáticamente:

$$a^{(t)} = b + Wh^{(t-1)} + Ux^{(t)},$$
$$h^{(t)} = \tanh\left(a^{(t)}\right),$$
$$o^{(t)} = c + Vh^{(t)},$$
$$y^{(t)} = softmax(o^{(t)})$$



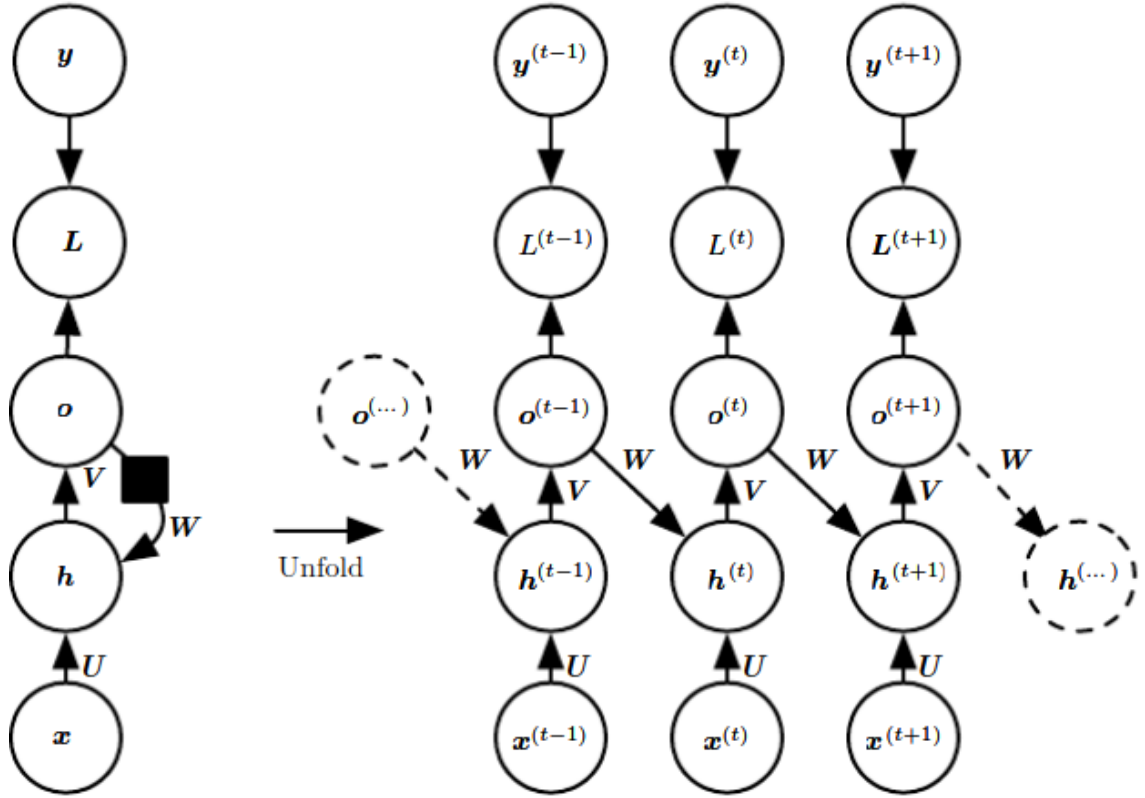
UNIVERSIDAD DE GRANADA
DaSCI

RNN. Procesamiento de secuencias de datos

RNN_2

- Esta arquitectura tiene menos capacidad de representación que RNN_1.

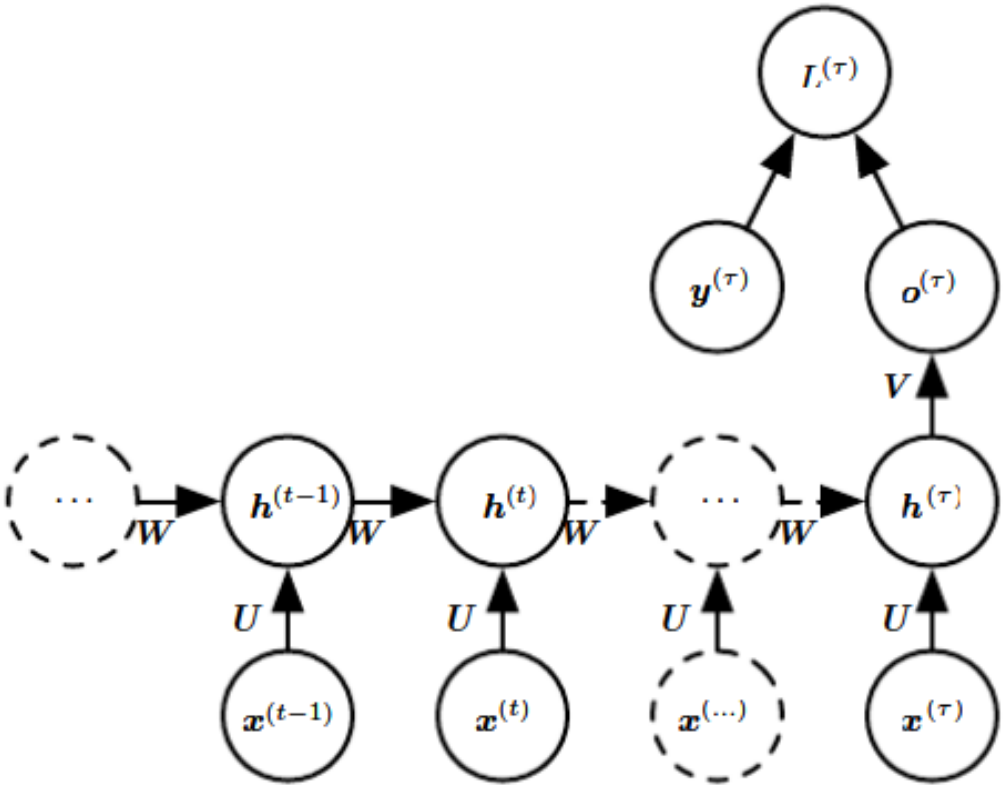
- La información en cada instante (*t*) de las capas ocultas (*h(t)*) no depende de la capa oculta del instante anterior (*h(t-1)*), sino de la salida (transformación adicional a la capa oculta (h(t-1)) anterior (*o(t-1)*), por tanto se pierde información.



RNN. Procesamiento de secuencias de datos

RNN_3

- Similar a RNN_1, dado que cada capa oculta (h(t)) depende de la anterior.
- Se produce una única salida que resumen la información de toda la secuencia.
- La última salida se puede usar para clasificar o como entrada de otras capas ocultas.
- **Aceptadora (*accepter*)***:* La última salida se usa para un procesamiento posterior.
- **Codificador (*encoder*)**: La última salida y otras características se usan como base para un procesamiento posterior.
- Estos dos tipos son muy similares y es más común hablar de **Codificador**.



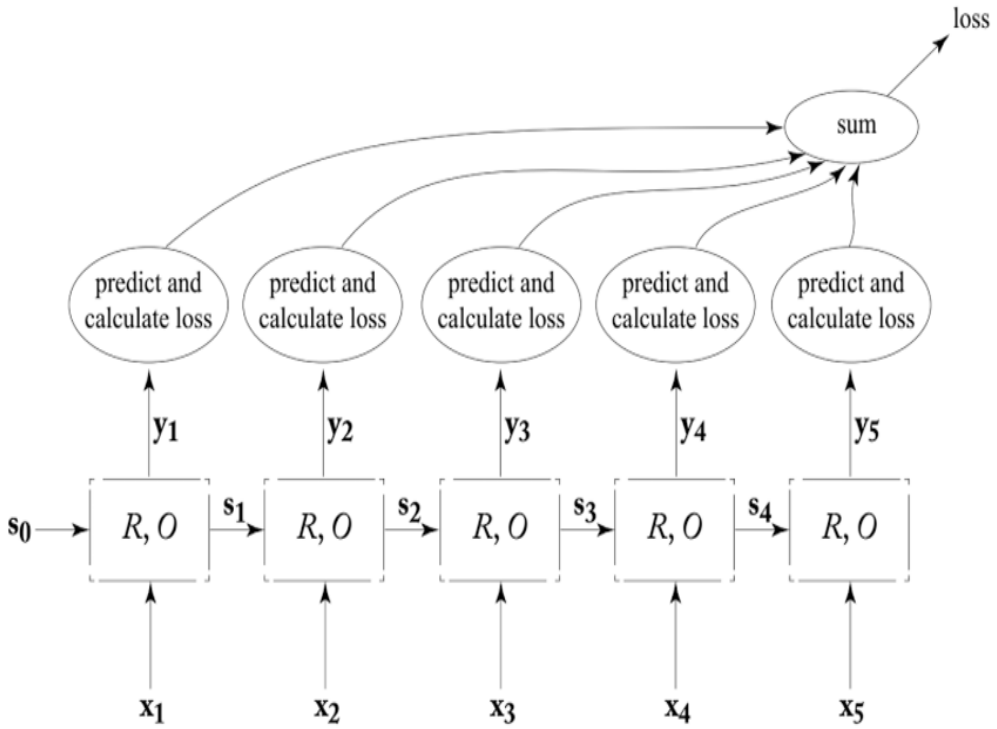
RNN. Procesamiento de secuencias de datos

Transductor (*transducer*)

- Usar cada salida de la secuencia para clasificar y calcular el coste (*loss*).

- Coste total: suma de los costes de cada salida de la secuencia.

- Aplicación:
  - Análisis morfológico (*pos-tagging*).
  - Análisis combinatorio categórico (CCG). Análisis morfológico con mayor nivel de detalle.
  - Modelos de lenguaje. Dado un conjunto de palabras determinar la siguientes.



RNN. Procesamiento de secuencias de datos

- Las RNN son una construcción que por lo general procesa/codifica la entrada de izquierda a derecha.
- El procesamiento de derecha a izquierda también se puede realizar.
- Depende del problema que el procesamiento se haga de izquierda a derecha o viceversa.

RNN. Bidireccionales.

- Hay problemas que requieren de la codificación de ambos sentidos de la secuencia. P. Ej.

El **banco** del parque es muy cómodo.

El **banco** no me ha prestado dinero.

- ¿Permitiría un procesamiento solo de izquierda a derecha desambiguar la palabra banco?

RNN. Bidireccionales.

- Según la hipótesis distribucional del lenguaje [1], palabras que aparecen en contextos similares tienen significados similares.

- ¿El contexto de una palabra sólo está determinado por los términos que la preceden?

- ¿Sólo lo determinan las palabras que suceden a un determinado término?

RNN. Bidireccionales.

El **banco** del parque es muy **cómodo**.

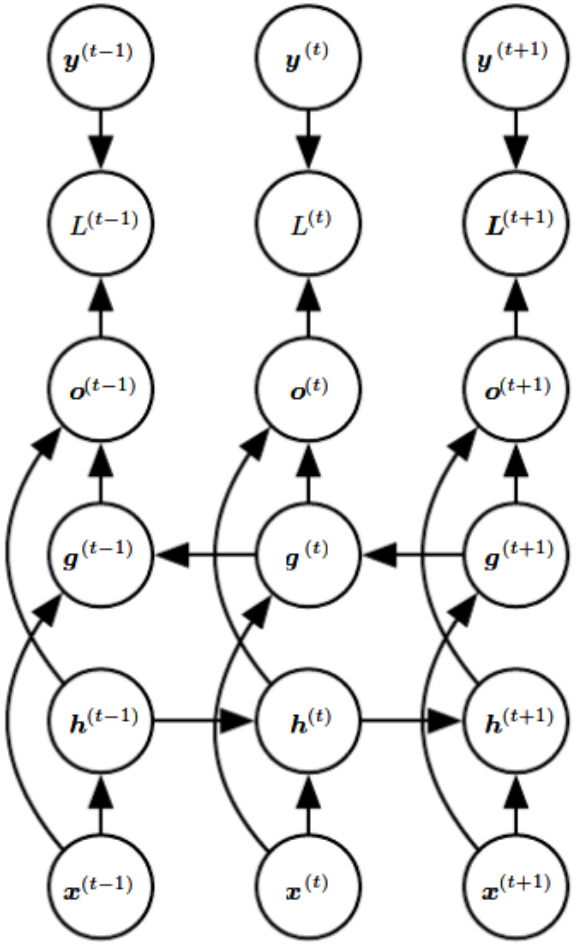
El **banco** no me ha **prestado dinero**.

- El contexto de una palabra lo determinan las palabras que la circundan.
- En nuestro ejemplo, los términos del contexto que determinan el significado concreto de la palabra **banco** son aquellas que la suceden.
- ¿Una RNN unidireccional de izquierda a derecha la hubiera podido codificar adecuadamente nuestro ejemplo?

RNN. Bidireccionales.

- Las RNN Bidireccionales son aquellas en las que se usan dos RNN aplicadas cada una en un sentido (izquierda a derecha; derecha a izquierda).

- La unidad de salida aglutina tanto información del pasado (precedente) como del futuro (posterior) de la secuencia.



RNN. Arquitecturas secuencia-secuencia

- Se usan en problemas en los que se pretende generar una secuencia a partir de otra secuencia.

- P. Ej. en traducción automática.

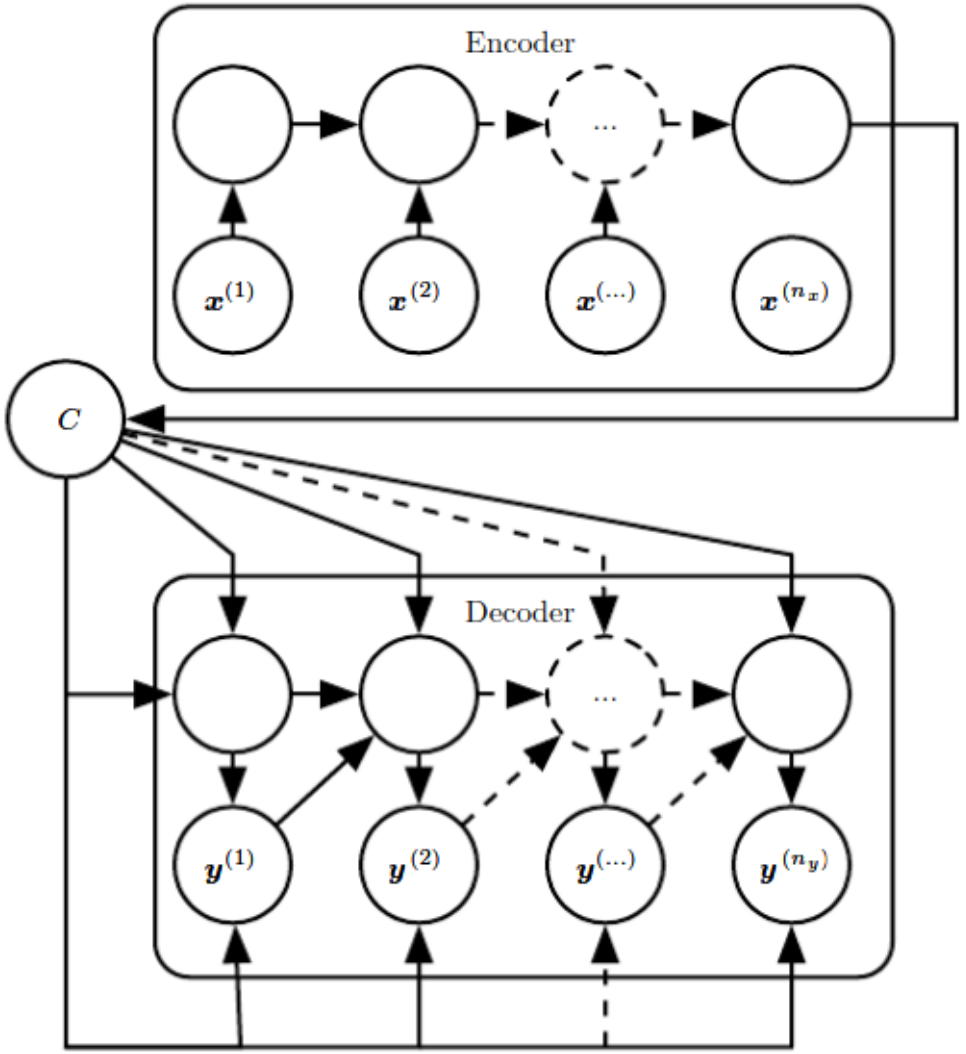
  Hello my name is Eugenio → Hola mi nombre es Eugenio

- Arquitectura de secuencia-secuencia o codificador-decodificador.

- Idea:
  - Usar una RNN para codificar la secuencia de entrada y generar una representación de alto nivel (C).
  - Usar una segunda RNN que toma de entrada C y genera una nueva secuencia.

- Importante: La secuencia de entrada y salida pueden ser de tamaños diferentes.

RNN. Arquitecturas secuencia-secuencia



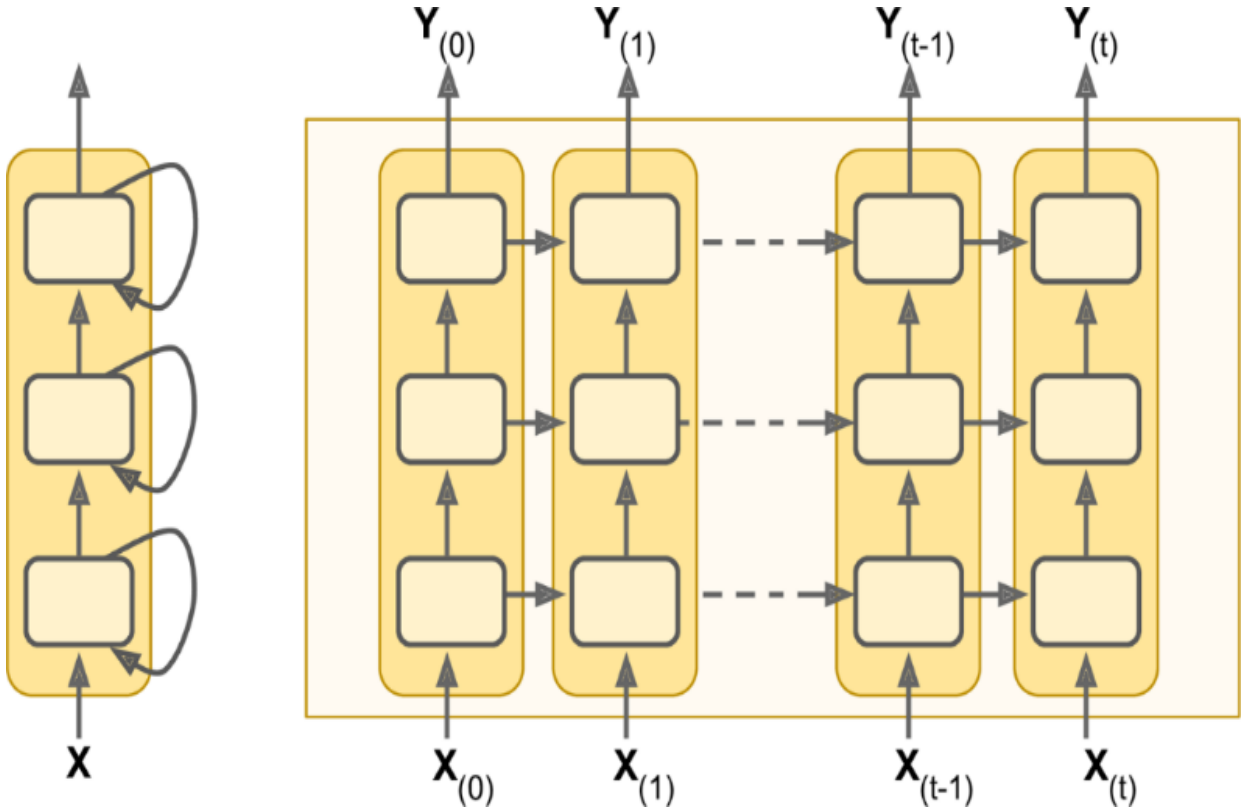
RNN. Arquitecturas secuencia-secuencia

- Ejemplo Traducción Automática con DeepL Pro.
- Inglés: The writing is weak not only English but also
- https://www.deepl.com/translator
- Español (salida): La escritura es débil, no sólo en inglés, sino **también en español**.
- **¿Qué hay extraño en la traducción?**

RNN. Multicapa o profundas

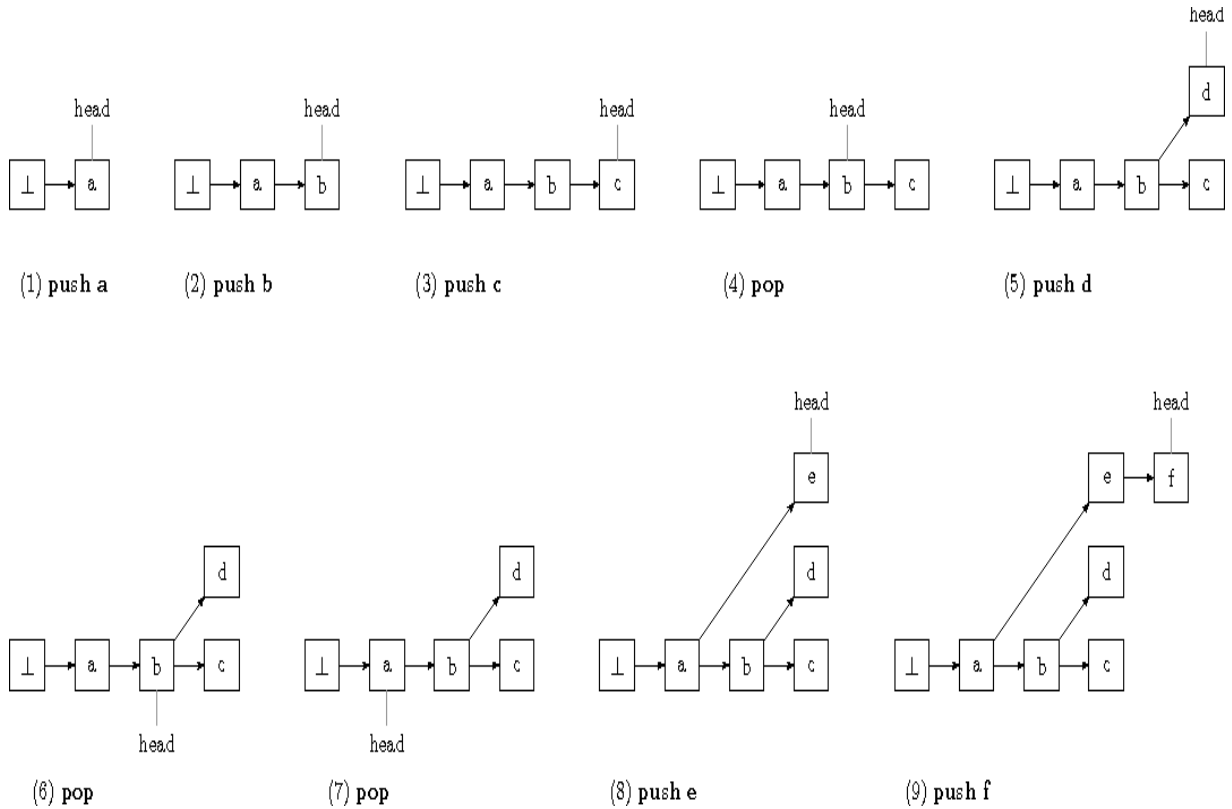
- Combinación secuencial de RNN.

- La salida de cada célula de la secuencia es la entrada de la célula de la siguiente capa de RNN. La salida de RNN multicapa son las salidas de las células de la última RNN.

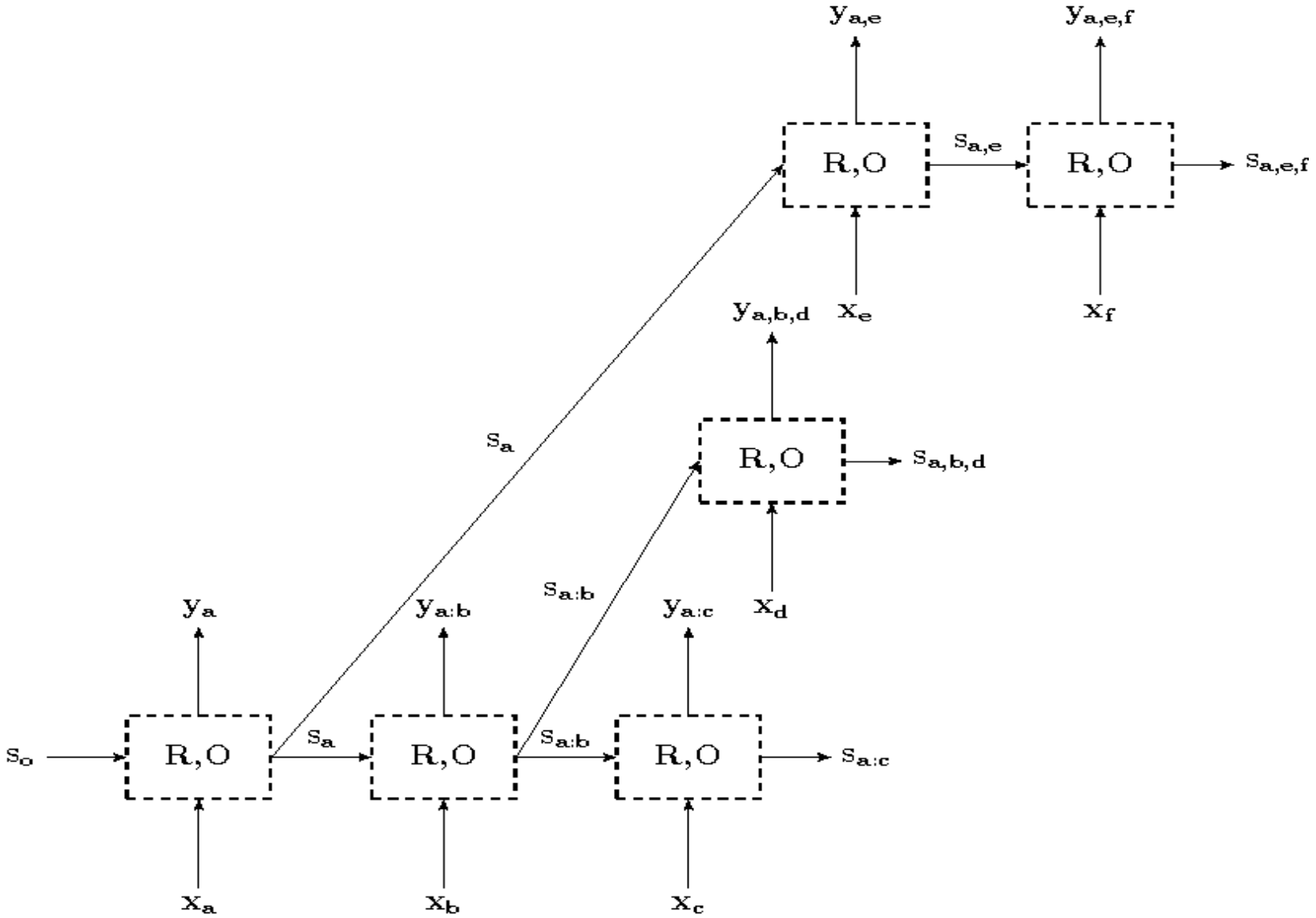


RNN. Representación de pilas

- Muchos analizadores en procesamiento del lenguaje se implementan usando una pila.
- En el procesamiento de pilas se suele trabajar sólo con la cabecera.
- Las RNN se usan para representar toda la pila.
- Para no tener que codificar la pila cada vez que ésta se modifica (sobretodo cuando se sacan elementos), se mantienen conjuntamente el estado de la pila y de la RNN.
- Se consigue usando pilas persistentes [2, 3], que mantienen los estados antiguos de la estructura de datos.



RNN. Representación de pilas



UNIVERSIDAD DE GRANADA
DaSCI

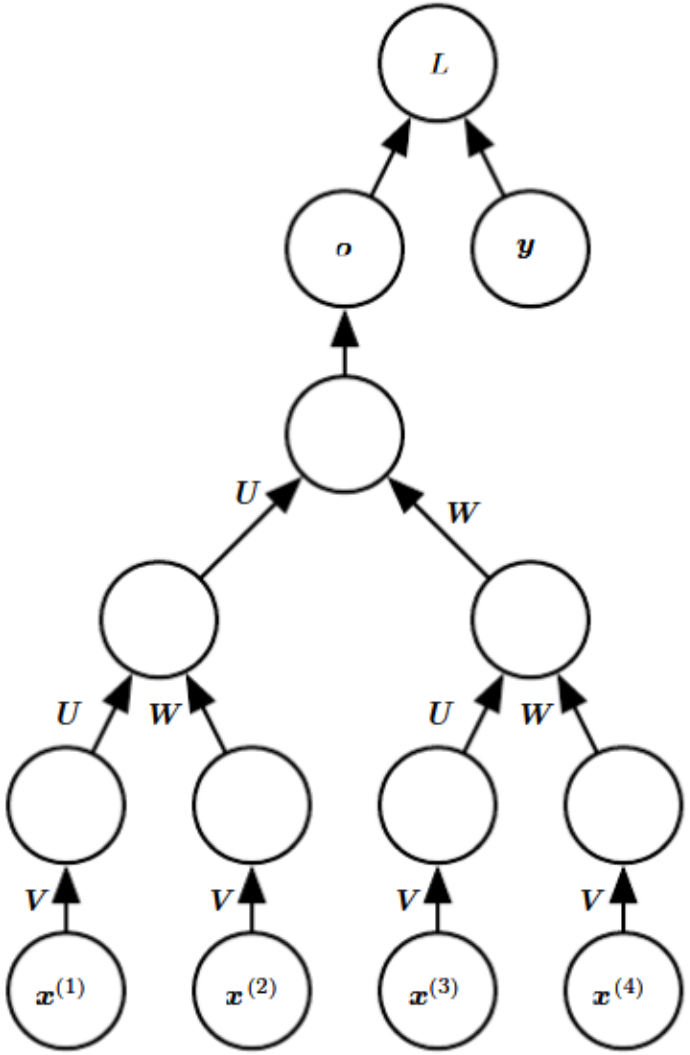
RNN. Redes recursivas

- Otra tipo de generalización de RNN, es decir, donde el procesamiento de una secuencia o elemento depende de otro elemento de la secuencia.

- En este caso el procesamiento toma estructura de árbol.

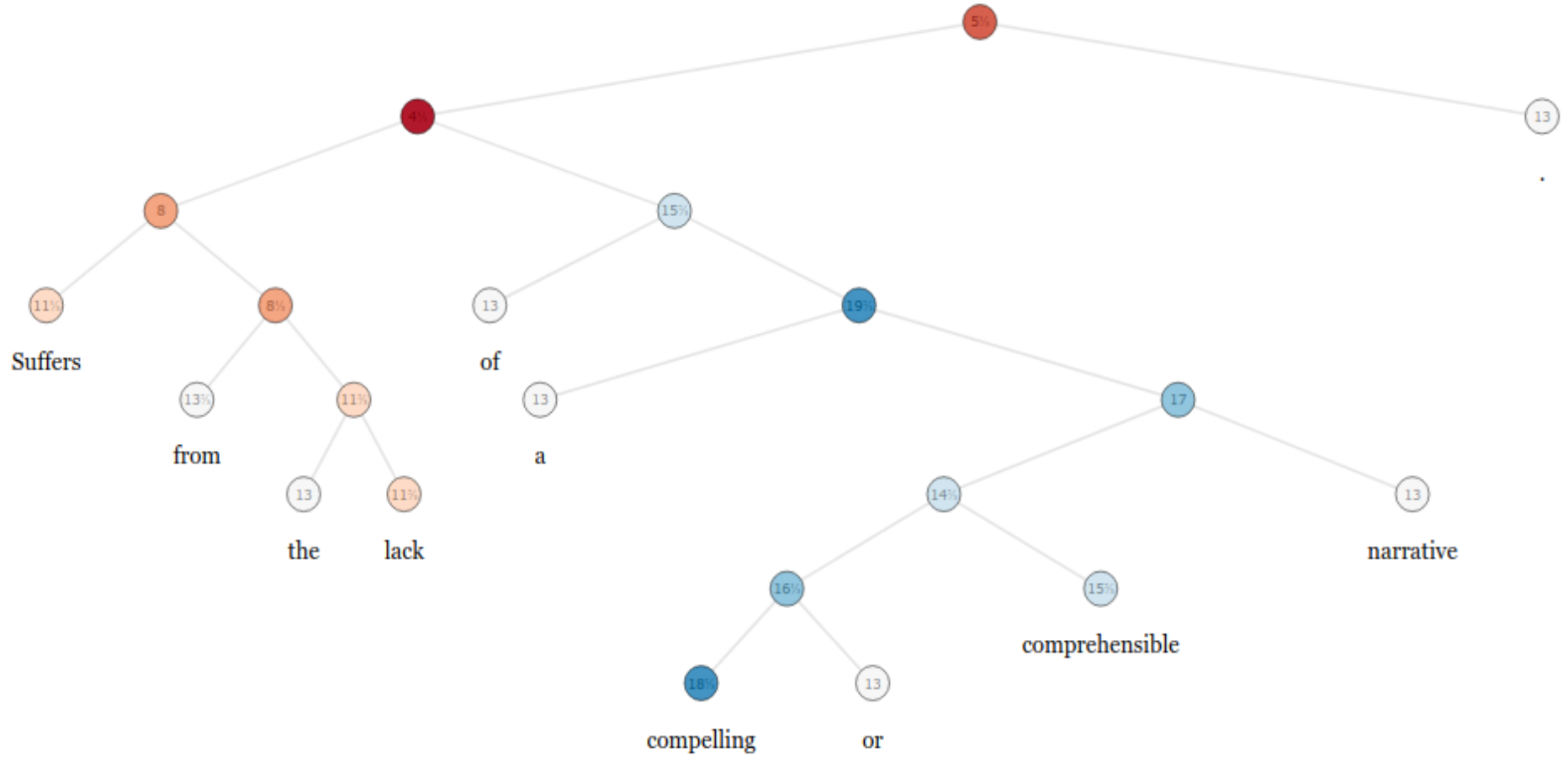
- Útil cuando se quiere modelar la estructura sintáctica de una oración.

RNN. Redes recursivas



RNN. Redes recursivas

- Stanford SentiBank [4, 5].



RNN. Dependencias lejanas

- Cuanto mayor es la secuencia, mayor es la distancia entre elementos, y por ende las dependencias se extienden (trivial).

- Esto ocasiona que las RNN puedan sufrir de dos problemas.
  - Desvanecimiento del gradiente. El valor del gradiente es cero o muy cercano a cero. Muy probable.
  - Explosión del gradiente. El valor del gradiente es muy elevado. Poco probable.

- ¿Por qué?

RNN. Dependencias lejanas

- Una RNN no es más que una composición de funciones. Recordemos.

$$\boldsymbol{h^{(t)} = f\left(h^{(t-1)}, x^{t};\ \theta\right) = g^{(t)}\left(x^{(t)}, x^{(t-1)}, x^{(t-2)}, \ldots, x^{(2)}, x^{(1)}\right)}$$
$$h^{(t)} = f\left(f\left(f\left(f\left(h^{(0)}, x^{(0)}; \theta\right), x^{(1)}\right), x^{(2)}\right), \ldots, x^{(t)}\right)$$

- Recordando que $f(x;\theta) = g(W^T x + b)$, y de menara simplificada sin tener en cuenta la función de activación (g) ni el sesgo (b), tenemos que

$$h^{(t)} = W^T h^{(t-1)}$$

- Se trataría de una multiplicación vectorial/matricial/tensorial iterativa a lo largo de toda la secuencia.

RNN. Dependencias lejanas

- ¿Qué ocurre con un escalar < 1 multiplicado muchas veces? P. Ej.

$$0{,}3^{400}$$

- ¿Qué ocurre con un escalar >1 multiplicado muchas veces? P. Ej.

$$1{,}3^{4000}$$

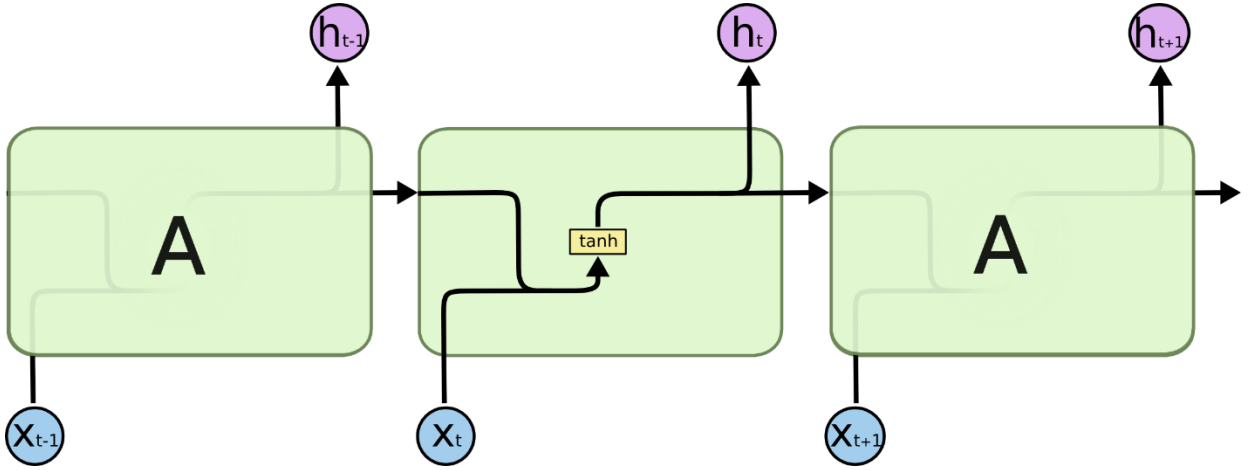
- Que dichos valores no se pueden representar por su ínfimo o gran tamaño.

- Igual le ocurre a la fórmula $h^{(t)} = W^T h^{(t-1)}$, y por ende el valor del gradiente es ínfimo, retrasando o haciendo casi impracticable el aprendizaje de la red.

RNN. Dependencias lejanas. Soluciones

- Una RNN es la repetición de un módulo de procesamiento (célula) a través de una secuencia.

- Dicha repetición puede provocar el desvanecimiento o la explosión del valor del gradiente.



RNN. Dependencias lejanas. Soluciones

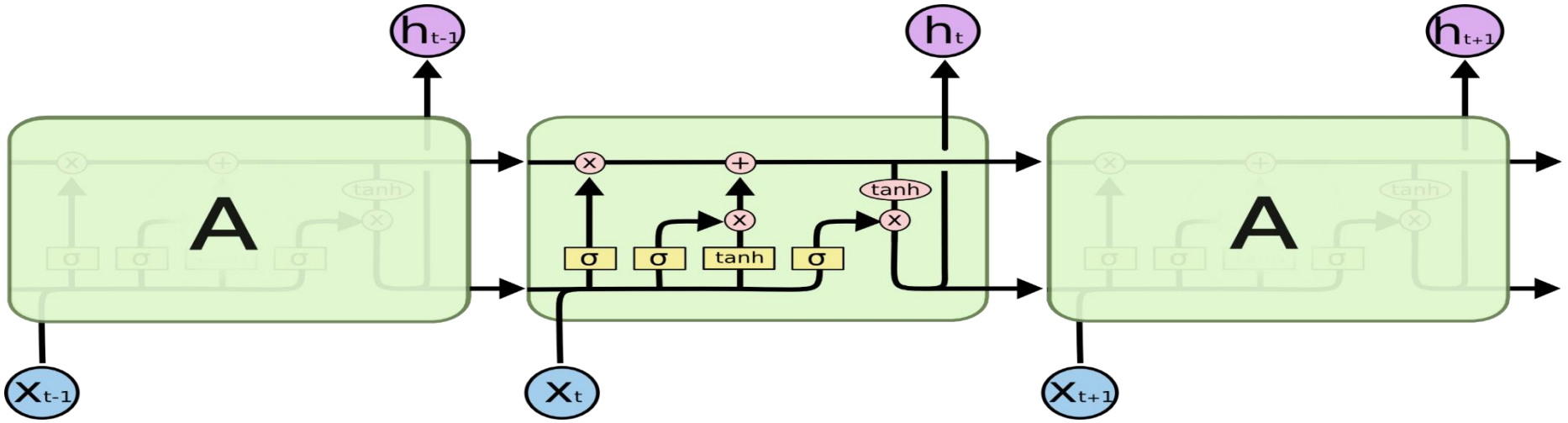
- Las soluciones pasan por definir rutas o atajos para saltarse varios elementos de la secuencia (*skip connections*), o para centrarse en determinados elementos de la secuencia definiendo distintas ponderaciones a las conexiones de las células de las RNN (*leaky units*).

- Soluciones basadas en fijación de saltos y pesos:
  - Redes Echo.
  - Redes con unidades con fuga (*leaky units*) y derivados.
  - Añadir saltos en la recurrencia.

- Soluciones basadas en aprendizaje automático de dichas rutas y ponderaciones: RNN basadas en puertas:
  - *Long Short-Term Memory* (LSTM) [6].
  - Otras arquitecturas basadas en puertas: GRU [7].

RNN basadas en puertas. LSTM

- Tienen la capacidad de aprender a olvidar la información no relevante.
- Para saber qué memorizar y qué olvidar añaden un conjunto de puertas (operaciones) a la célula base de las RNN.
- Puertas: Entrada (*input gate*), Olvido (*forget gate*) y Salida (*output gate*).



RNN basadas en puertas. LSTM

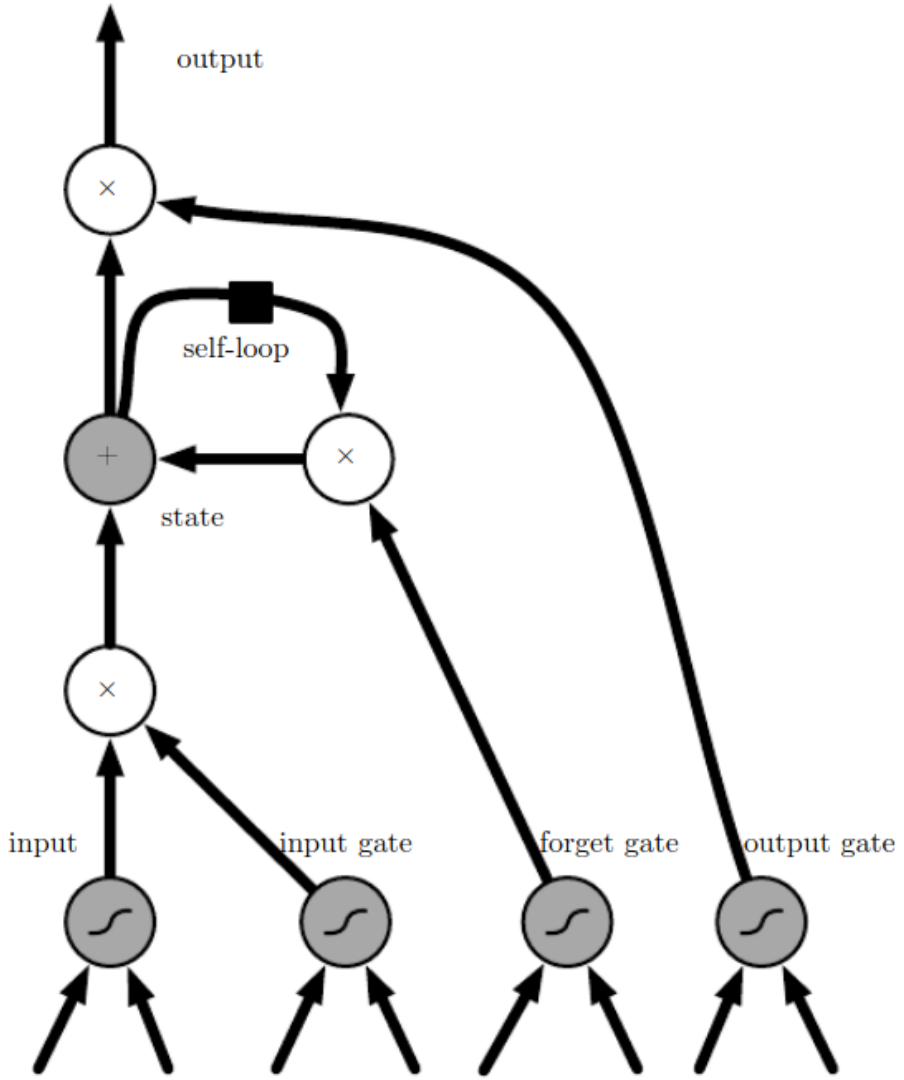
Puerta de olvido (*forget, f*)

- Función: Teniendo en cuenta la nueva entrada, determina que olvidar de lo procesado anteriormente.
- Matemáticamente:

$$\boldsymbol{f^{(t)} = \sigma\left(W_f\left[h^{(t-1)}, x^{(t)}\right] + b_f\right)}$$

Donde:

- $\boldsymbol{x^{(t)}}$ es la entrada en el momento t.
- $\boldsymbol{h^{(t-1)}}$ es el valor de salida de la célula anterior.
- $\boldsymbol{W_f}$ parámetros de la puerta de olvido.
- $\boldsymbol{b_f}$ parámetros de sesgo de la puerta de olvido.
- $\boldsymbol{\sigma}$: Función sigmoide de activación. Devuelve valores entre 0 (desactivación) y 1 (activación).



UNIVERSIDAD DE GRANADA
DaSCI

RNN basadas en puertas. LSTM

Puerta de entrada (*input gate, i*)

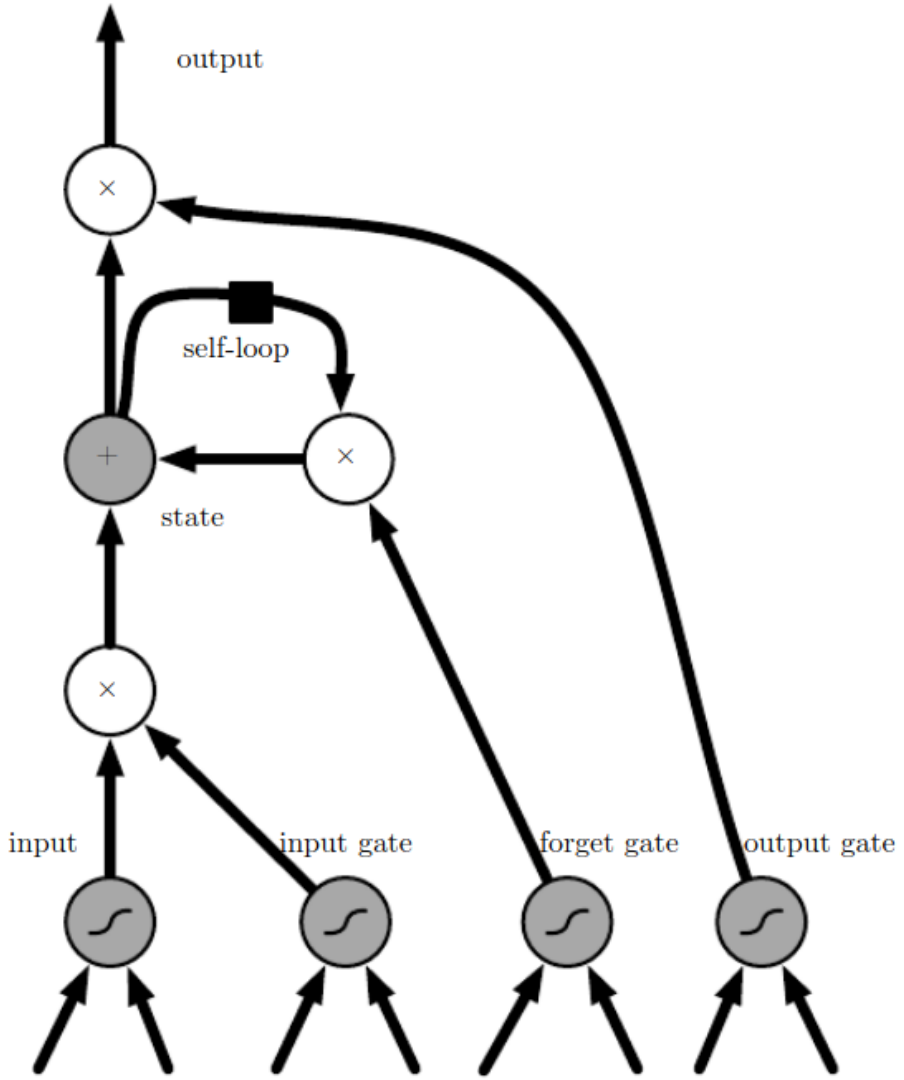
- Función: Aprender los parámetros de entrada para su composición con el estado (memoria) de la RNN.
- Fases:
  - Seleccionar la información que será usada.
  - Calcular el valor de los candidatos que podrán ser usados en la actualización del estado.
- Matemáticamente:

$$i^{(t)} = \sigma\left(W_i\left[h^{(t-1)}, x^{(t)} + b_i\right]\right)$$

$$\widetilde{s^{(t)}} = \tanh\left(W_s\left[h^{(t-1)}, x^{(t)} + b_s\right]\right)$$

Donde:

- $\widetilde{s^{(t)}}$: Se refiere a un cálculo previo del estado (estado local) en el instante t de la LSTM.



RNN basadas en puertas. LSTM

Combinación entrada (i) y olvido (f).

- Función: Actualizar el estado de la red (s) en función de la puerta de olvido y salida.
- Matemáticamente:

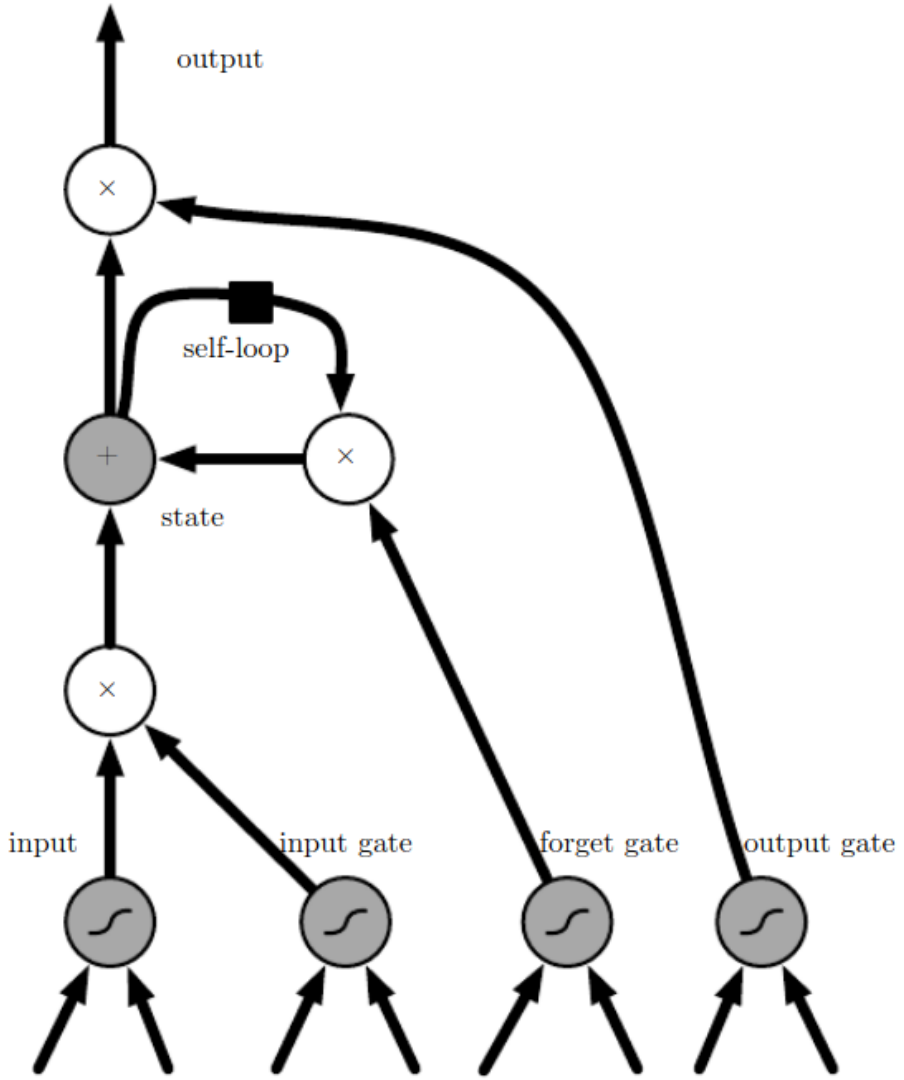
$$s^{(t)} = f_{(t)} * s^{(t-1)} + i^{(t)} * \widetilde{s^{(t)}}$$

Donde:

- $s^{(t)}$: Se refiere al estado de la LSTM en el instante t.

Interpretación:

- Parte del estado anterior ($s^{(t-1)}$) se olvida y se añade a la composición de la entrada con el estado local (instante t) de la red.



RNN basadas en puertas. LSTM

Puerta de salida (*ouptut gate, o*).

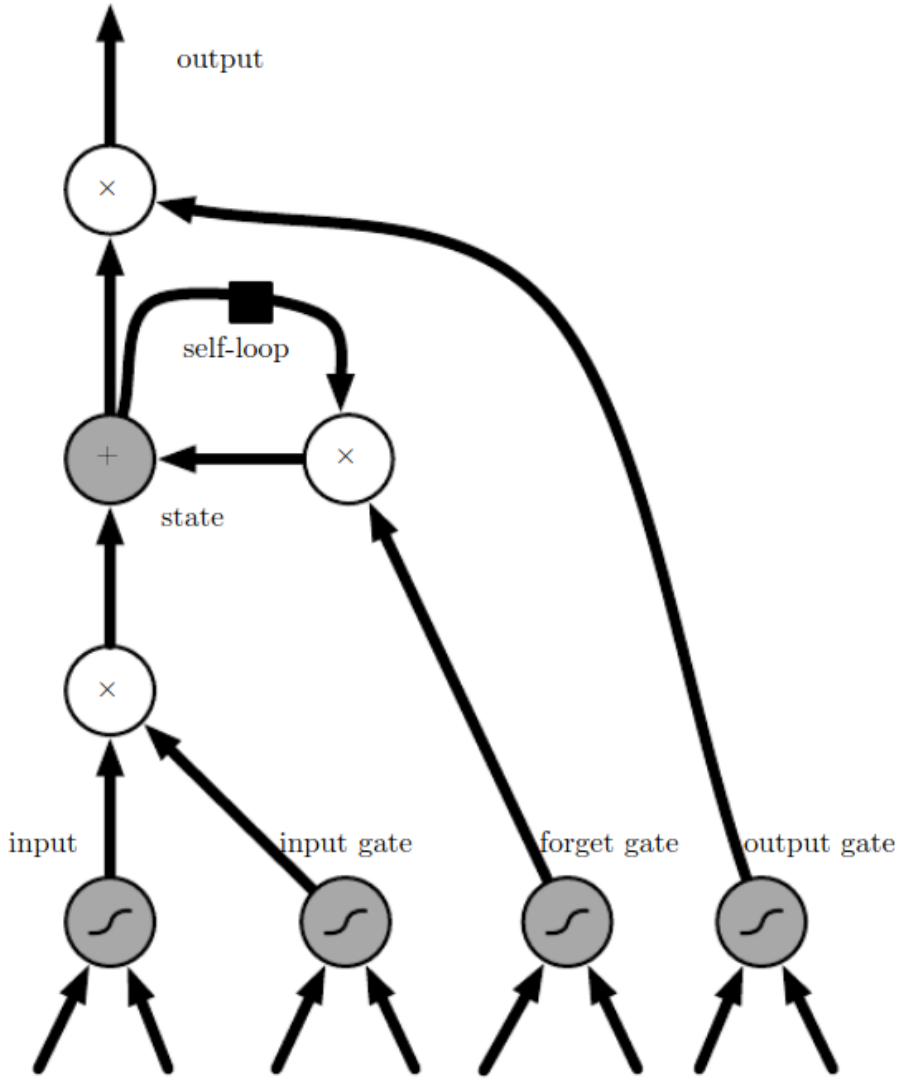
- Función: Aprender los parámetros de la salida.

- Matemáticamente:

$$o^{(t)} = \sigma\left(W_o\left[h^{(t-1)}, x^{(t)} + b_o\right]\right)$$

Donde:

- $\boldsymbol{\sigma}$: Función sigmoide de activación. Devuelve valores entre 0 (desactivación) y 1 (activación).



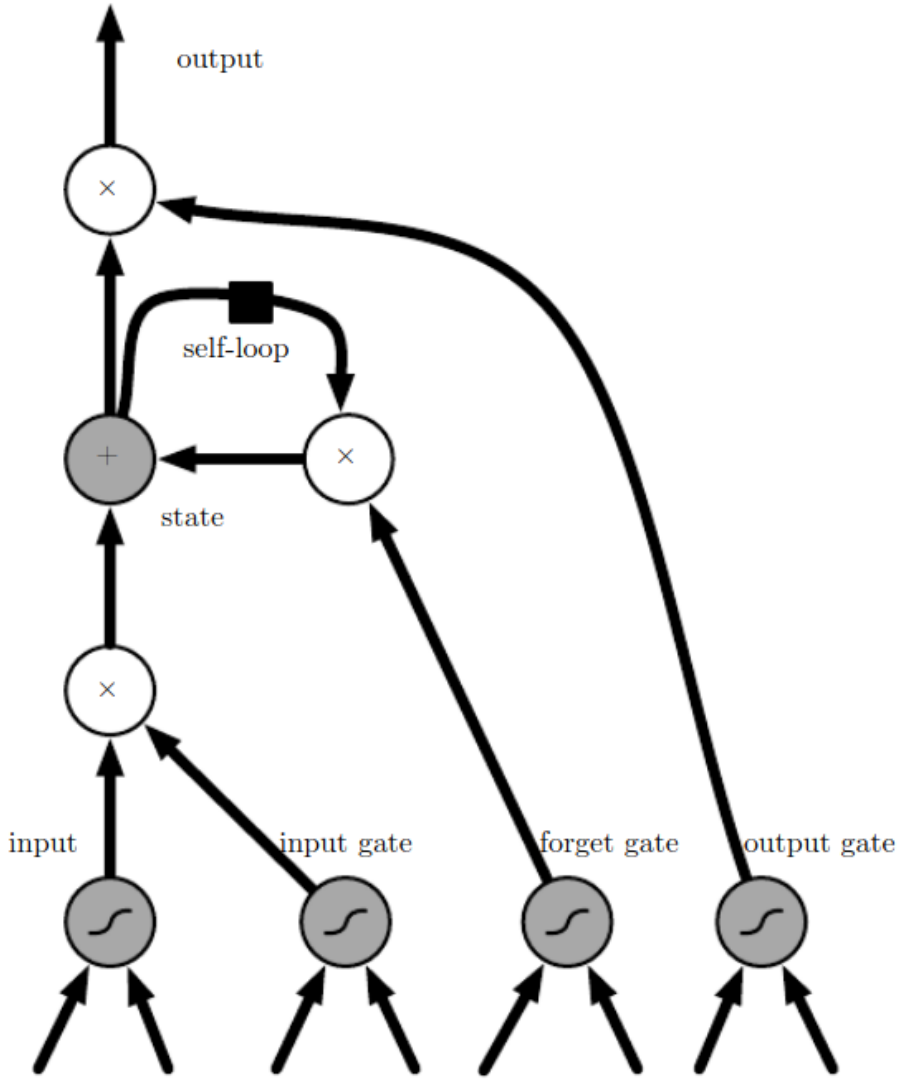
RNN basadas en puertas. LSTM

Cálculo salida de la célula.

- Función: Combinar los parámetros de la salida con el estado (s) calculado en función de los parámetros de entrada y olvido.

- Matemáticamente:

$$\boldsymbol{h^{(t)} = o_t * \tanh(s_t)}$$



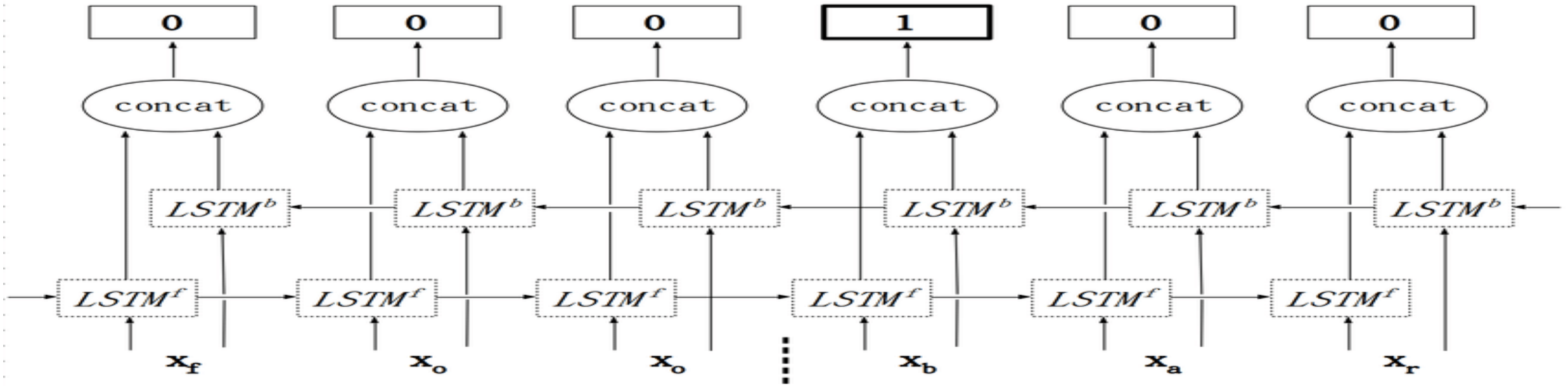
RNN basadas en puertas. LSTM

- Las RNN LSTM han demostrado que son capaces de aprender secuencias con dependencias amplias.

- Suelen estar presentes en sistemas que son el estado del arte en muchas tareas de clasificación, como son muchas de las que tienen que ver con el **Procesamiento del Lenguaje Natural**.

RNN basadas en puertas. BiLSTM

- Las LSTM son un tipo de RNN, por lo que también se pueden construir arquitecturas basadas en una LSTM que procesa la secuencia de izquierda a derecha, y otra de derecha a izquierda.
- Estas arquitecturas se llaman BiLSTM.



UNIVERSIDAD DE GRANADA
DaSCI

RNN. Mecanismo de Atención

- Las secuencias de datos suelen ser amplias.
- No todos los datos de la secuencia puedan ser de interés.
- En algunas situaciones puede ser interesante que la red sólo preste atención a un determinado subconjunto de elementos de la secuencia.

- P. Ej.

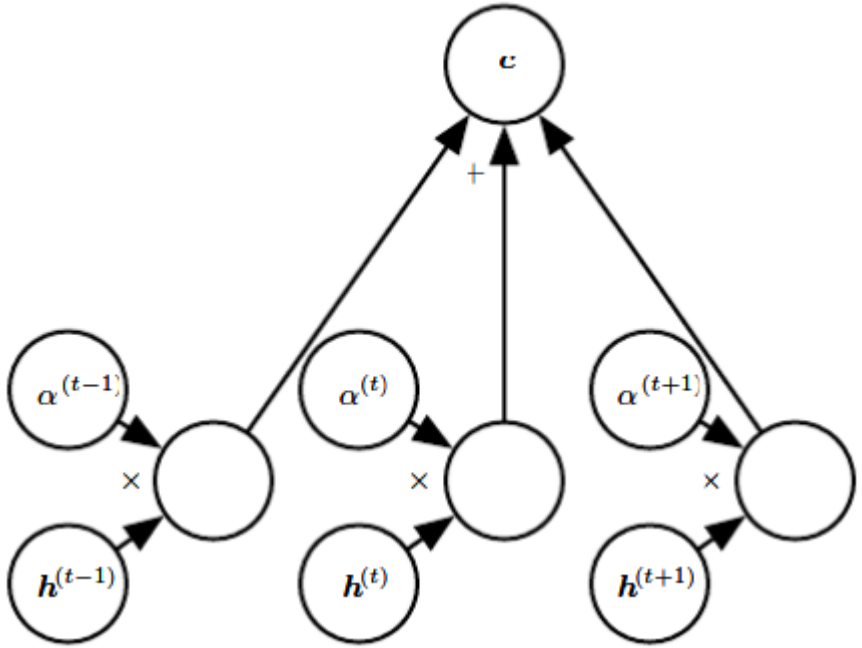
El **banco** del **parque** es muy **cómodo**.

El **banco** no me ha **prestado dinero**.

- Para desambiguar la palabra banco puede que la red deba prestar una mayor atención a un determinado subconjunto de palabras.
- ¿Cómo hacerlo dinámicamente? Método de atención [8].

RNN. Mecanismo de Atención

- La atención suele estar compuesta de:
  - Lectura y codificación de la secuencia (en crudo o ya procesada).
  - Determinación de la importancia de cada elemento de la secuencia. Esto se suele hacer usando la función de activación *softmax*.
  - Combinación de la salida de *softmax* con el flujo de procesamiento de la red.
- Interpretación: La atención permite aprender la importancia de cada elemento de la secuencia, de manera que se seleccionan aquellos que aportarán más a la clasificación.



¿CÓMO DISEÑAR UNA RED NEURONAL?

¿Cómo diseñar una red neuronal?

- Tras tanta teoría sobre RN (CNN+RNN), ¿cómo se deben articular los diferentes módulos de una RN para obtener buenos resultados?
- ¿El diseño de una RN es tan automático como parece que muestran los tutoriales que hay en Internet?
- Pues no es tan fácil como se suele pintar.

- Cada problema de IA requiere un análisis profundo para determinar:
  - El objetivo.
  - Entender los datos de los que se dispone.

- Estos dos elementos determinarán el proceso de diseño.
- Como se puede intuir, no es tan automático como se sugiere.

¿Cómo diseñar una red neuronal? A tener en cuenta

- Identificar el objetivo a resolver:
  - Problema: Clasificación de objetos, detección de objetos, clasificación de secuencias, extracción de secuencias.
  - Medida de error: Esta medida indicará si nuestro está haciendo lo correcto, por lo que es fundamental seleccionar la medida más adecuada: *Accuracy*, *Precision*, *Recall*, F1, Error medio (MAE)…

- Definición de caso base: Esencial para conocer el rendimiento mínimo que debe presentar la red. Estar por debajo del caso base es una catástrofe.

¿Cómo diseñar una red neuronal? A tener en cuenta

- Cantidad de datos. Determinar si para mejorar el rendimiento de la red se precisan de más datos o de mejorar el ajuste de todos los parámetros.
- Ajuste de parámetros. Se trata de un proceso tedioso donde se deben evaluar la influencia de cada parámetro en el resultado de la red. Se pueden establecer estrategias:
  - Producto cartesiano (*Grid search*): Se trataría de seleccionar un pequeño conjunto de parámetros, calcular su producto cartesiano y realizar todas las pruebas oportunas. Muy costoso en tiempo.
  - Búsqueda aleatoria. Buscar y evaluar aleatoriamente el valor de los parámetros. Empíricamente se ha demostrado que suele reportar mejores resultados que el producto cartesiano.

¿Cómo diseñar una red neuronal? A tener en cuenta

- Técnicas de depuración:
  - Analizar el comportamiento de la red en la fase de entrenamiento y evaluación.
    - Imprimir valores.
    - Usar herramientas del tipo TensorBoard de Tensorflow.
  - Análisis de errores.
    - Estudiar la razón de los errores graves.
    - Estudiar la razón de los errores no graves.
    - Puede ayudar el análisis de los valores de la capa de salida (sigmoide o softmax).
- Para ganar capacidad de generalización, añadir regularización
  - Dropout.
  - EarlyStopping
  - Añadir ruido (l1/l2) a la función de coste o a los parámetros de las capas ocultas.
  - Procesamiento por lotes (mini-batches).

CASO PRÁCTICO

Proyecto

- Problema: Clasificación de opiniones a nivel de tuit en español.
- Conjunto de datos: Corpus General de TASS.
- Entrenamiento: 7.218.
- Número de clases: 4. P, NEU, N, NONE
- Lenguaje de programación: Python 3.
- Librerías PLN: NLTK 3.0.
- Librería de redes neuronales: Keras (2.3.0).
- Se evaluará con una RNN con embeddins aleatorios y otra con *embeddins* pre-entrenados.

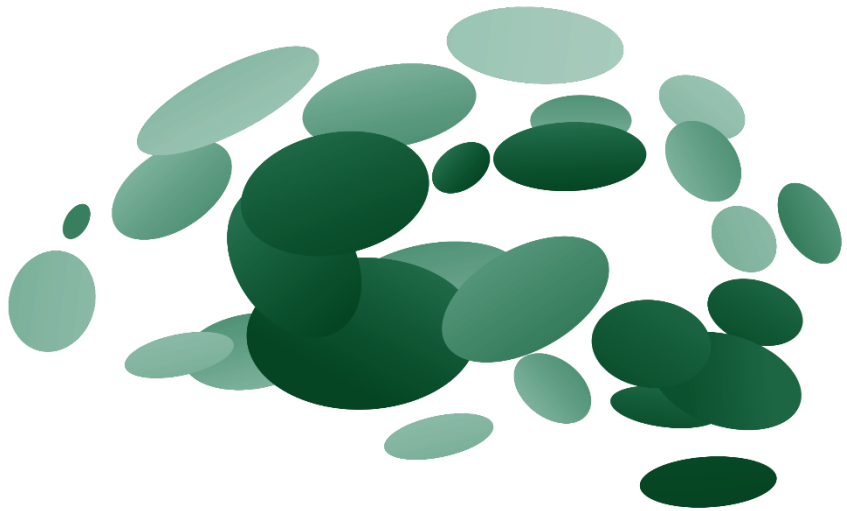
TASS
Taller de Análisis Semántico en la SEPLN

Bibliografía

- [1] Firth, J.R. (1957). "A synopsis of linguistic theory 1930-1955". Studies in Linguistic Analysis: 1–32. Reprinted in F.R. Palmer, ed. (1968). Selected Papers of J.R. Firth 1952-1959. London: Longman.
- [2] Okasaki, C. (1999). Purely functional data structures. Cambridge University Press.
- [3] Goldberg, Y., Zhao, K., & Huang, L. (2013). Efficient implementation of beam-search incremental parsers. In Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Vol. 2: Short Papers) (Vol. 2, pp. 628-633).
- [4] https://nlp.stanford.edu/sentiment/
- [5] Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A., & Potts, C. (2013). Recursive deep models for semantic compositionality over a sentiment treebank. In Proceedings of the 2013 conference on empirical methods in natural language processing (pp. 1631-1642).

Bibliografía

- [6] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9*(8), 1735-1780.
- [7] Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2015, June). Gated feedback recurrent neural networks. In *International Conference on Machine Learning* (pp. 2067-2075).
- [8] Bahdanau, D., Cho, K. & Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. In ICLR 2015.
- [9] Pang, B., & Lee, L. (2008). Opinion mining and sentiment analysis. Foundations and Trends in Information Retrieval, 2(1–2), 1-135.
- [10] Cambria, E., & Hussain, A. (2015). SenticNet. In Sentic Computing (pp. 23-71). Springer, Cham.
- [11] Martínez Cámara, E., Almeida Cruz, Y., Díaz Galiano, M. C., Estévez-Velarde, S., ... & Piad-Morffis, A. (2018). Overview of TASS 2018: Opinions, health and emotions. URL: http://ceur-ws.org/Vol-2172/p0_overview_tass2018.pdf



DaSCI