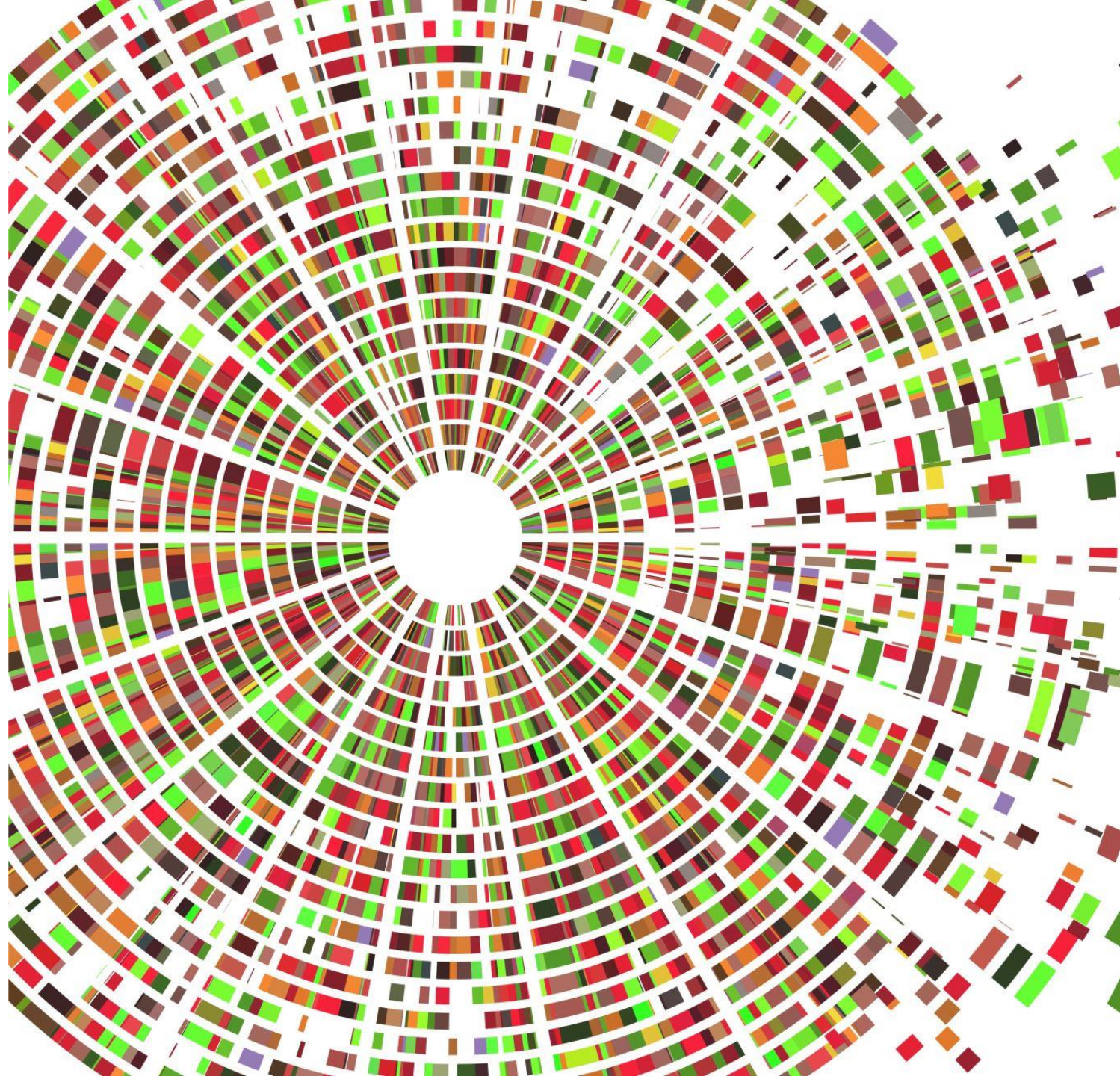


Métodos avanzados de ciencia de datos

Prof. Emily Díaz



Contenido

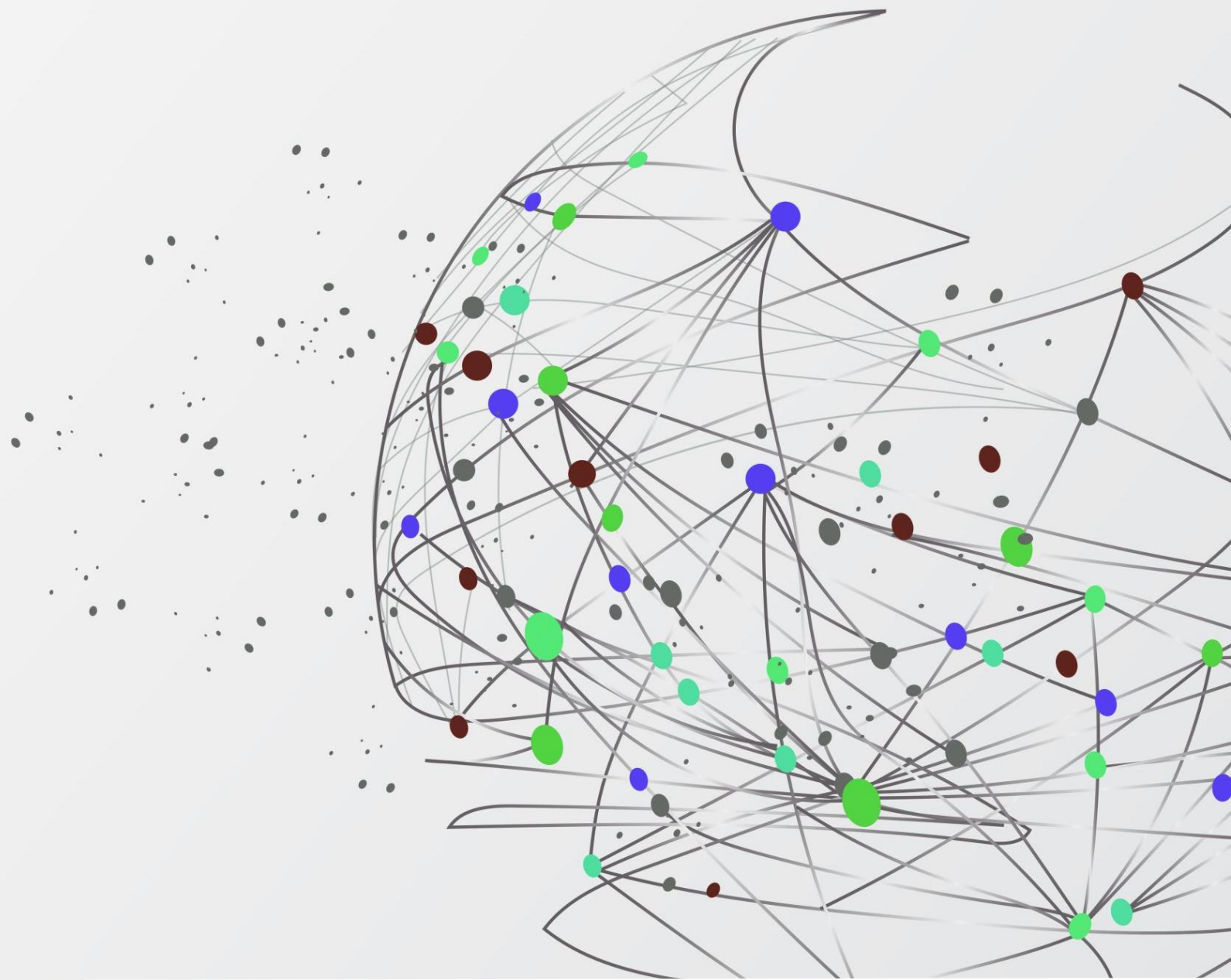


Redes neuronales con texto



Arquitecturas LSTM, GRU y Bidireccionales

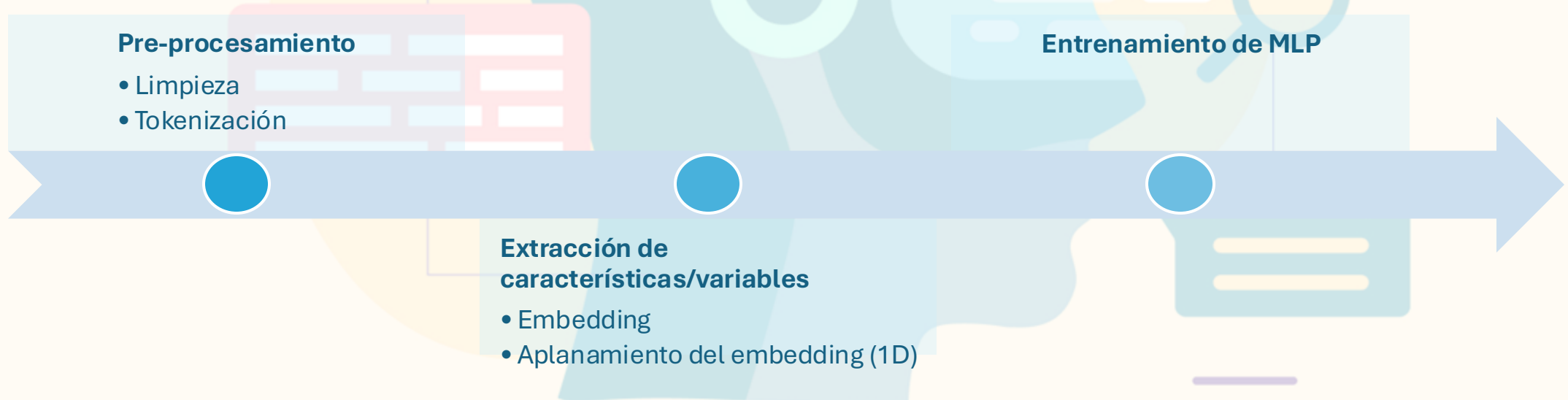
Redes Neuronales para texto



MLPs para datos de texto

¿Es posible utilizarlas?

- Sí, los MLP pueden manejar texto aplanándolo en un vector de características, generalmente con **embeddings**.
- Sin embargo, existen ciertas limitaciones para usar MLP con texto porque no están diseñados inherentemente para **datos secuenciales**.



Ventajas y desventajas de usar MLP para analizar texto

Ventajas

- Simplicidad de implementación
- Funciona con textos siempre y cuando se defina una longitud fija
- Entrenamiento rápido para datos pequeños: para conjuntos de datos pequeños y tareas donde las relaciones secuenciales no son esenciales, los MLP pueden funcionar adecuadamente y entrenarse rápidamente.

Desventajas

- **Sin conciencia de secuencia:** los MLP no capturan inherentemente relaciones secuenciales o temporales. Al aplanar el texto se pierde información valiosa sobre las dependencias de las palabras.
- **Vectores de entrada grandes**
- Baja escalabilidad: dificultades con entradas de texto grandes, ya que el aplanamiento genera una dimensionalidad excesiva.

Dependencia de palabras

”Este producto no es bueno”



Si solo analizamos las palabras individuales, podríamos ver "bueno" e interpretar erróneamente la oración como positiva. Sin embargo, si utilizamos n-gramas, captamos "no bueno", lo que hace que el sentimiento cambie a negativo.

La dependencia de las palabras en NLP se refiere a las **relaciones e interacciones entre las palabras** de una oración que ayudan a definir su significado. Comprender estas dependencias es esencial porque las palabras **dependen en gran medida del contexto** y los significados de las palabras individuales pueden cambiar en función de las palabras que las rodean.

N-gramas

- Los n-gramas son secuencias contiguas de n elementos (normalmente palabras o caracteres) en datos de texto.
- Capturan **dependencias de palabras a corto plazo** y brindan información sobre la estructura del lenguaje mediante el análisis de pequeños grupos de palabras.
- Los n-gramas se utilizan comúnmente en el procesamiento del lenguaje natural para tareas como el modelado del lenguaje, la clasificación de textos y la recuperación de
- Si bien los N-gramas capturan dependencias de corto plazo, **tienen limitaciones para comprender dependencias más largas**

N-Gram

N=1:

This is a sentence

Uni-grams

this,
is,
a,
sentence

N=2:

This is a sentence

Bi-grams

this is,
is a,
a sentence

N=3:

This is a sentence

Tri-grams

this is a,
is a sentence

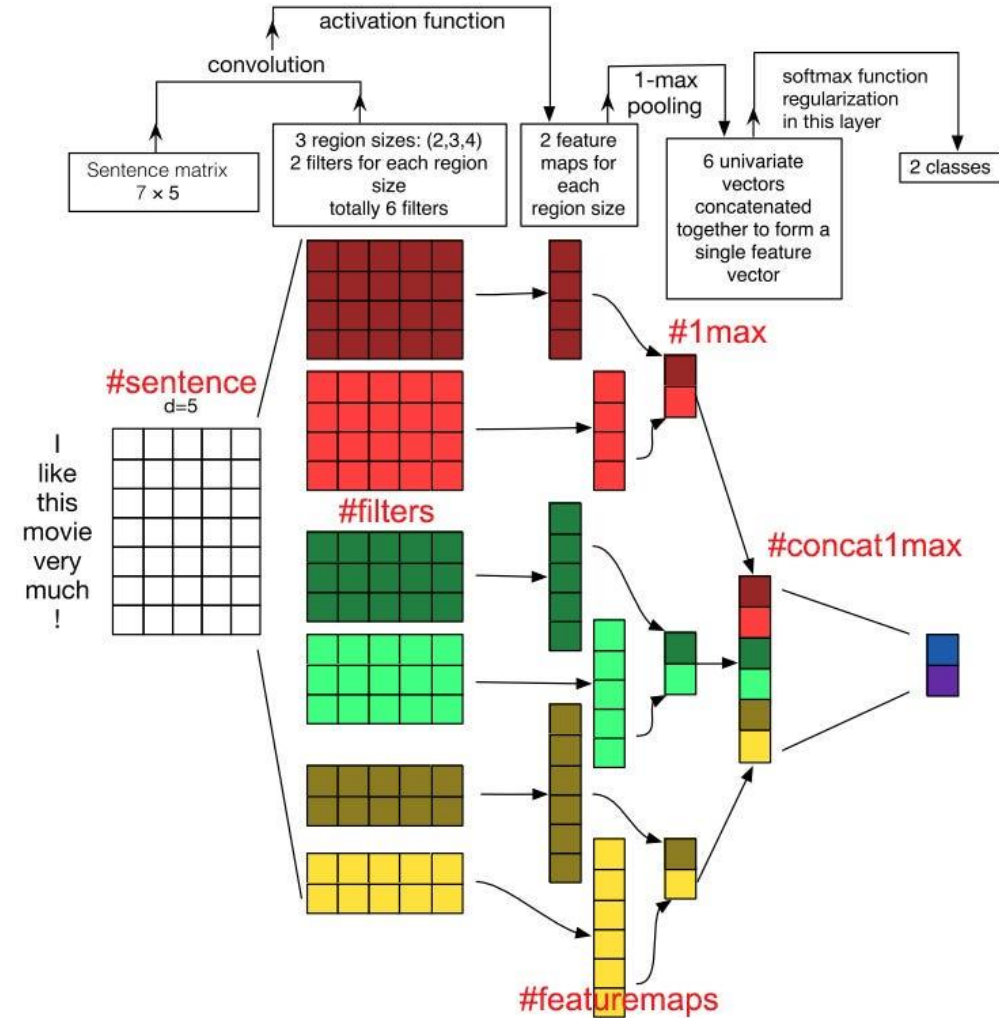
CNN para texto

Las CNN pueden manejar texto y capturar patrones locales, pero generalmente son menos efectivas que las Redes Neuronales Recurrentes (RNN)

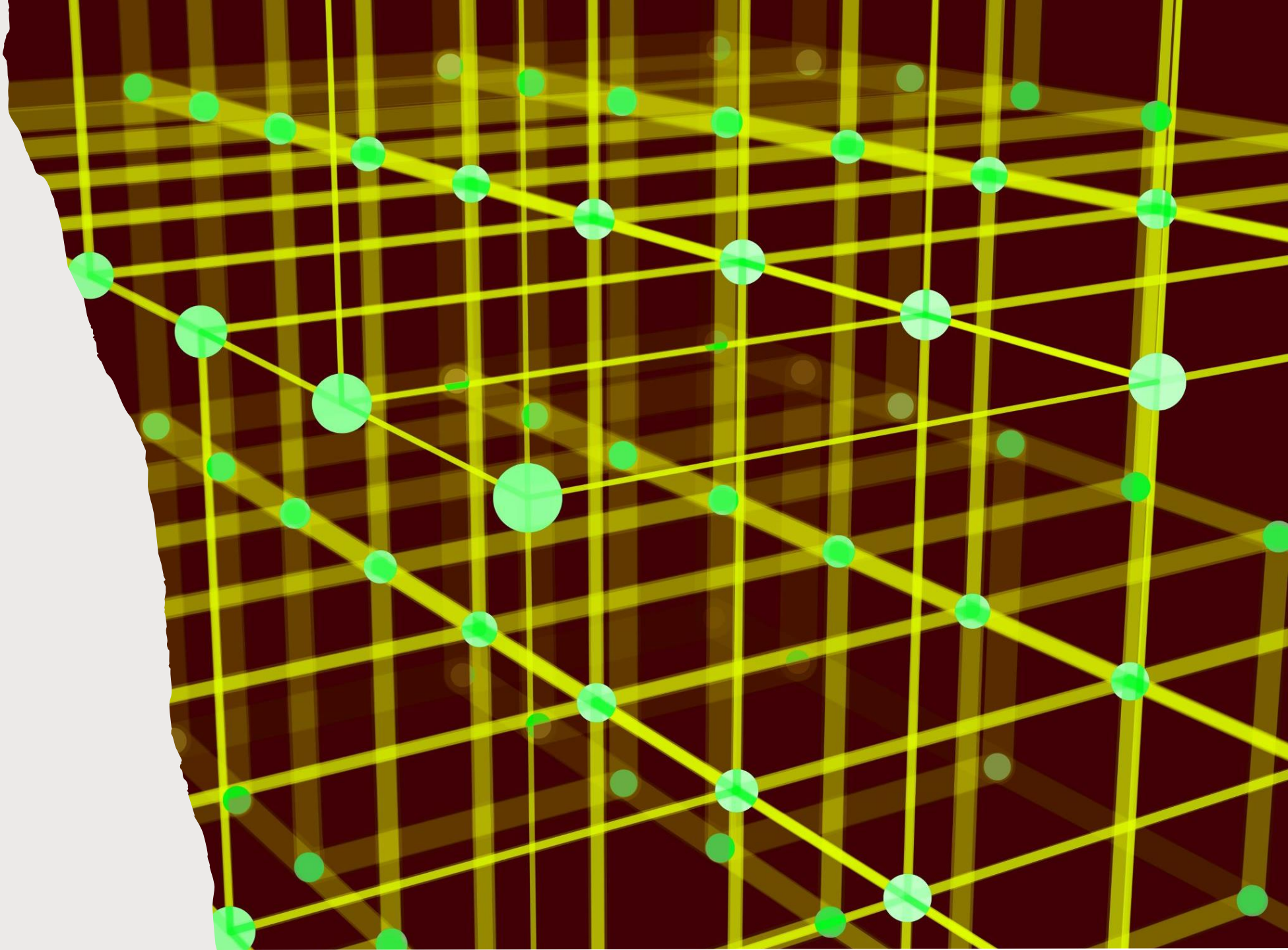
- **Campo receptivo local:** las CNN usan filtros para capturar características locales, lo que funciona bien para palabras vecinas (por ejemplo, "no es bueno") pero tiene **dificultades para capturar dependencias en fragmentos de texto más largos**.
- **Ventanas de contexto de tamaño fijo:** los filtros en las CNN capturan **n-gramas de tamaño fijo**, lo que les **dificulta modelar dependencias en diferentes partes de una oración** sin capas profundas y núcleos grandes, lo que puede ser computacionalmente ineficiente.
- **Sin mecanismo de memoria:** las CNN **no tienen un mecanismo inherente para retener información** en capas como lo hacen las RNN con sus estados ocultos.

¿Cuándo las CNN podrían funcionar bien para texto?

- Las CNN aún pueden ser efectivas para ciertas **aplicaciones de texto**, especialmente **para secuencias cortas o cuando las frases o patrones locales son más importantes que la estructura general de la secuencia**.
- A menudo se utilizan en tareas de **clasificación de oraciones**, como la detección de spam o la clasificación de temas, donde **la relación entre palabras es menos compleja**.

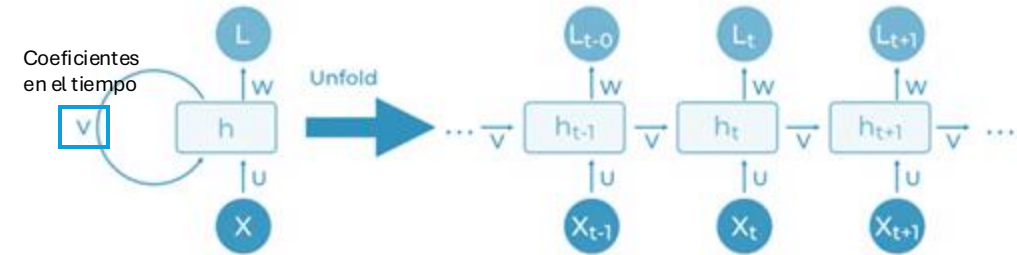


Redes
neuronales
recurrentes



¿Qué son las RNN?

- Se distinguen por su “memoria”, ya que toman información de **entradas anteriores para influir en la entrada y salida actuales**
- Otras redes neuronales asumen que las entradas y las salidas son independientes entre sí, en RNN la salida de las redes neuronales recurrentes **depende de los elementos anteriores dentro de la secuencia**
- Utilizan algoritmos de propagación hacia adelante y hacia atrás **a través del tiempo (BPTT)** para determinar los gradientes (o derivadas), lo que es **ligeramente diferente de la retropropagación tradicional**, ya que es específico para los datos de secuencia. BPTT se diferencia del enfoque tradicional en que BPTT suma los errores en cada paso de tiempo
- Sus entradas y salidas pueden variar en longitud
- Hay distintas arquitecturas dentro de la familia RNN:
 - Básica/Vainilla
 - Bidireccional
 - Long short-term memory (LSTM)
 - Gated recurrent units (GNUs)
 - Encoder-decoder (codificador-decodificador)

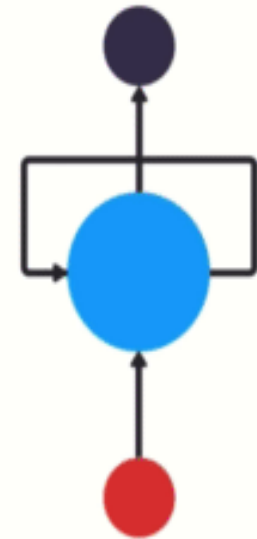


$$h^{(t)} = f(h^{(t-1)}, x^{(t)}; \theta)$$

En las RNN, el estado oculto se actualiza combinando información del estado oculto anterior $h^{(t-1)}$ y la entrada actual $x^{(t)}$. Esta recurrencia permite que la RNN procese datos secuenciales al retener información a lo largo del tiempo, capturando dependencias entre diferentes partes de la secuencia.

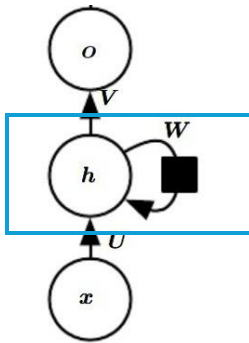
¿Por qué las RNN son generalmente mejores para texto?

- Las RNN están diseñadas para manejar datos secuenciales, lo que las hace más adecuadas para texto.
- **Retención de secuencias:** las RNN **retienen información de pasos de tiempo anteriores** en su estado oculto, lo que ayuda a capturar dependencias entre palabras (contexto).
- **Longitudes de entrada variables:** a diferencia de las MLP, las RNN pueden procesar secuencias de **longitudes variables** sin necesidad de relleno o truncamiento, lo que reduce los requisitos de procesamiento de datos y **evita ruido innecesario en los datos**.
- **Uso eficiente de parámetros:** **comparten parámetros a lo largo de pasos de tiempo**, lo que las hace eficientes y les permite generalizar mejor en datos secuenciales.
- Las RNN **pueden tener dificultades con dependencias a largo plazo y tienden a ser lentas de entrenar**, especialmente para textos extensos. Sin embargo, arquitecturas RNN más avanzadas como **LSTM y GRU mitigan cierta parte de estos problemas** al retener o descartar información de forma selectiva a lo largo de pasos de tiempo.



Ejemplo de tipo de cálculo de capas en RNN

U	Input-to-hidden
W	Hidden-to-hidden
V	Hidden-to-output



$$\mathbf{a}^{(t)} = \mathbf{b} + \mathbf{W}\mathbf{h}^{(t-1)} + \mathbf{U}\mathbf{x}^{(t)}$$

$$\mathbf{h}^{(t)} = \tanh\left(\mathbf{a}^{(t)}\right)$$

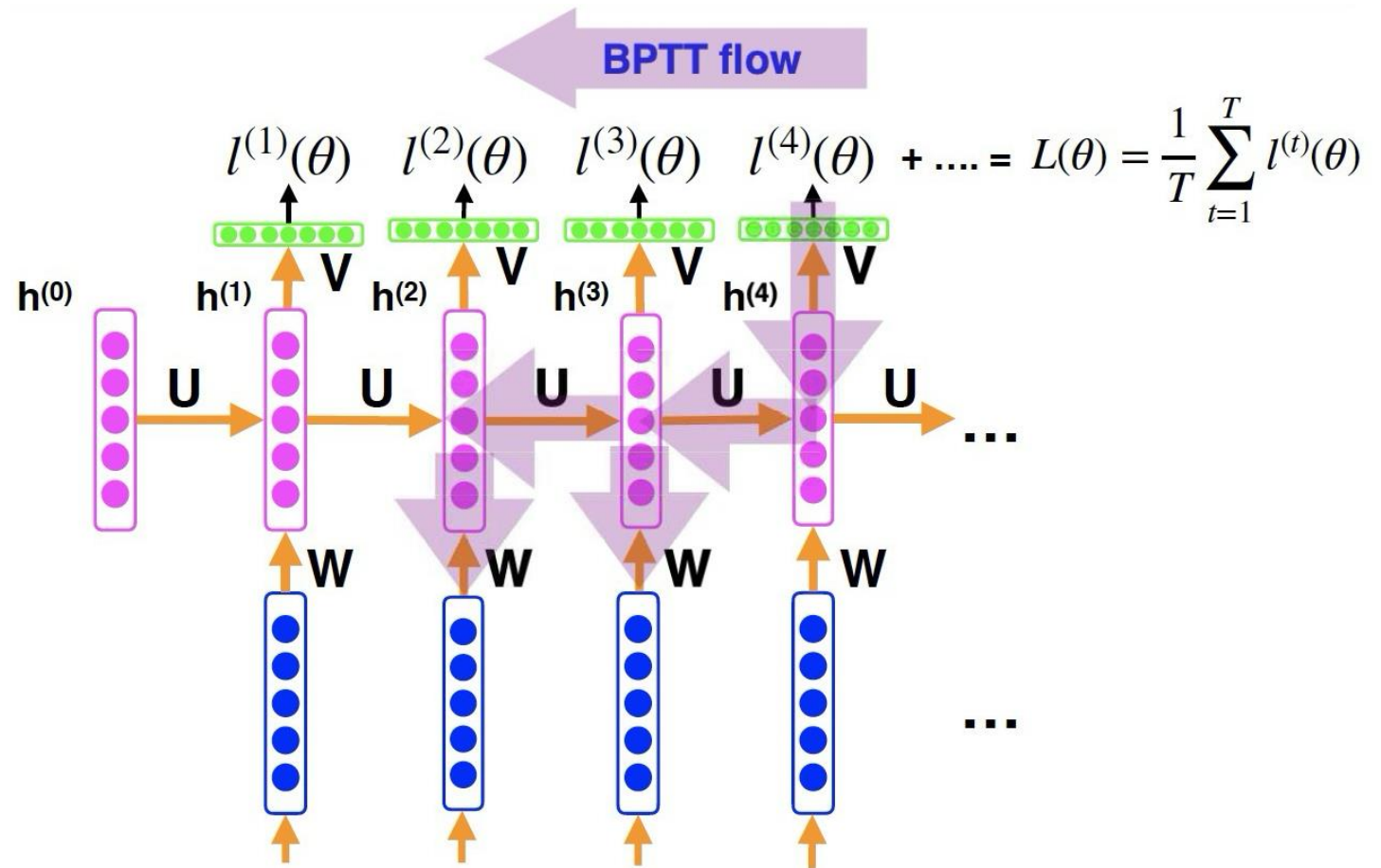
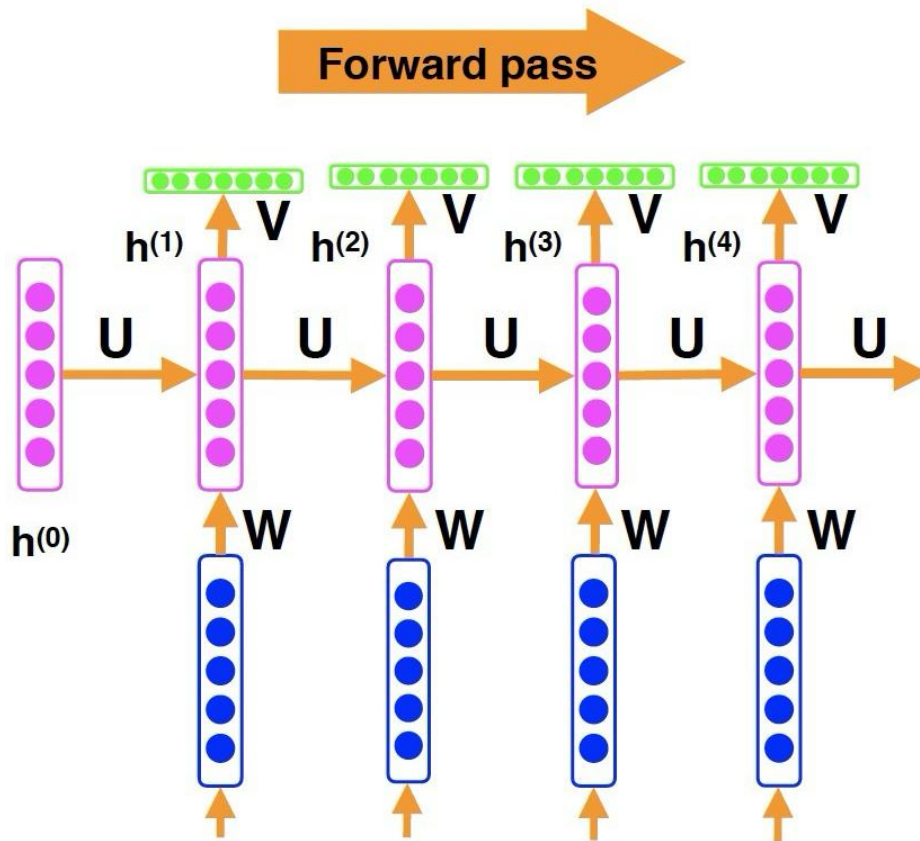
$$\mathbf{o}^{(t)} = \mathbf{c} + \mathbf{V}\mathbf{h}^{(t)}$$

$$\hat{\mathbf{y}}^{(t)} = \text{softmax}\left(\mathbf{o}^{(t)}\right)$$

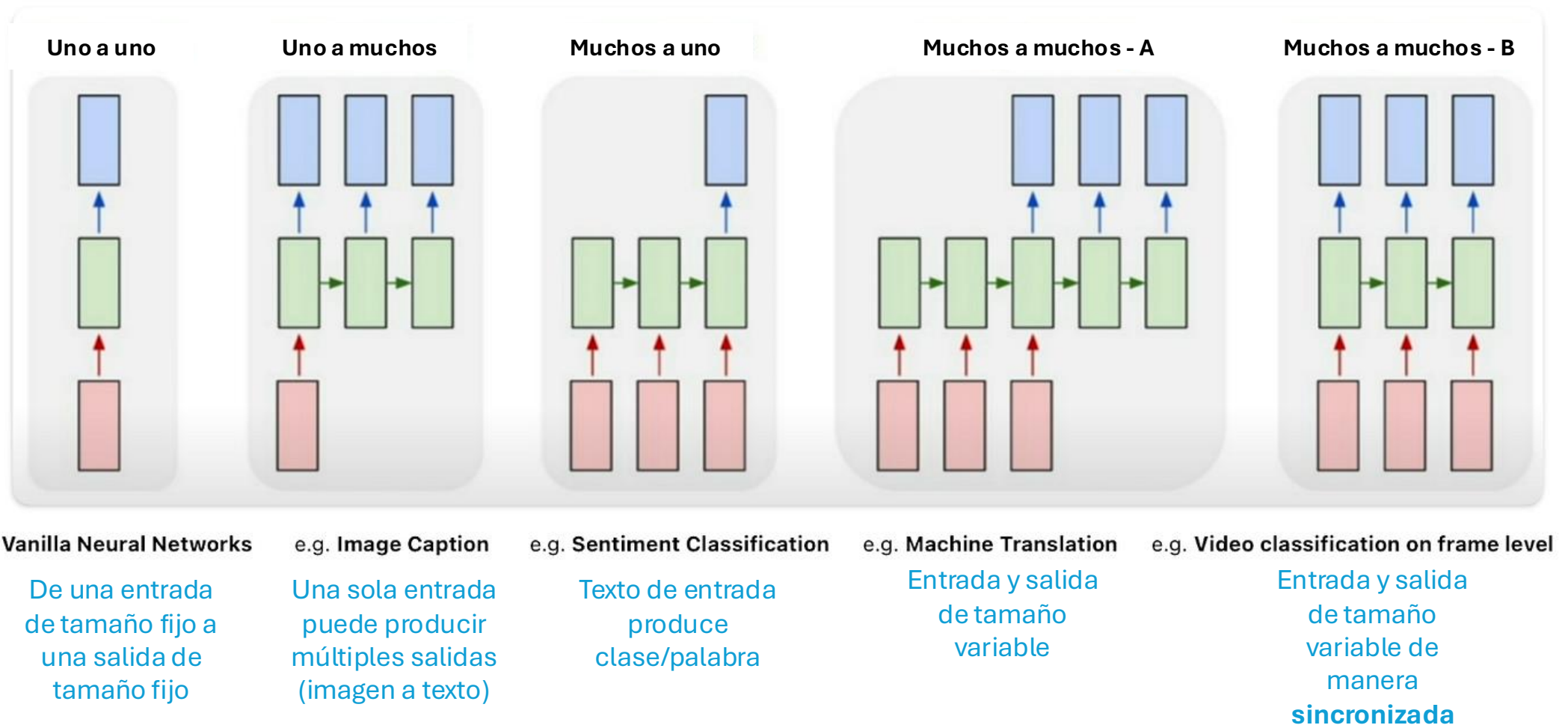
[Goodfellow 2016]

En lugar de $\mathbf{W}^T\mathbf{X} + \mathbf{b}$ como en MLP, tenemos $\mathbf{W}^T\mathbf{h}(t-1) + \mathbf{U}^T\mathbf{x}(t) + \mathbf{b}$ para las capas recurrentes

Propagación hacia delante y atrás BPTT



Distintos tipos de RNN basados en datos de entrada y salida



Limitaciones de RNNs

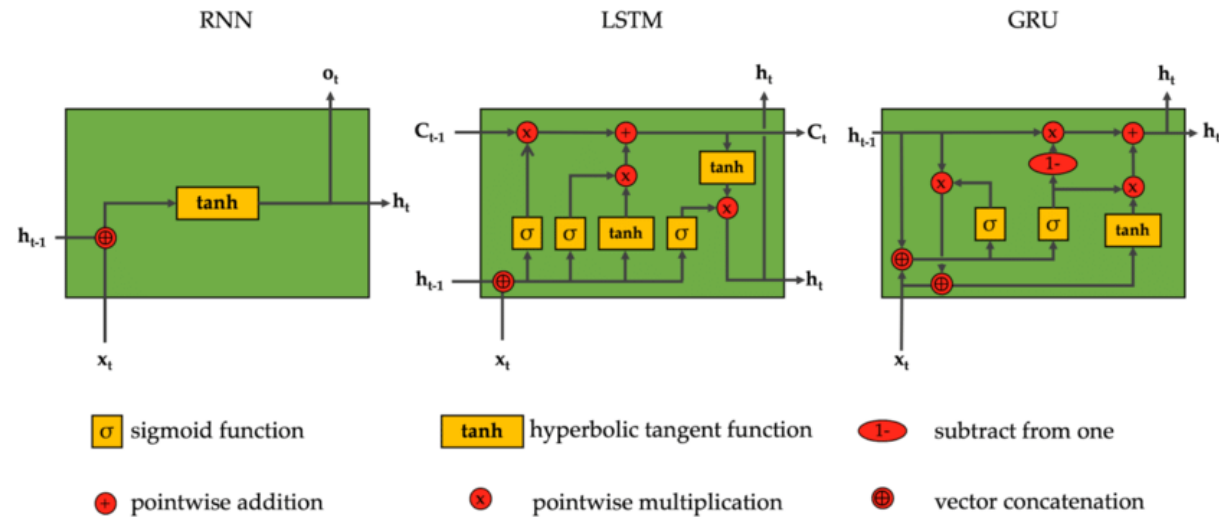
- **Gradiente explosivo:** cuando las derivadas individuales son grandes, la derivada final también se volverá enorme y los **pesos cambiarán drásticamente**.
- **Gradiente evanescente:** a medida que aumenta el número de capas ocultas, **el gradiente se vuelve muy pequeño** y los pesos apenas cambiarán, lo que dificultará el proceso de aprendizaje.



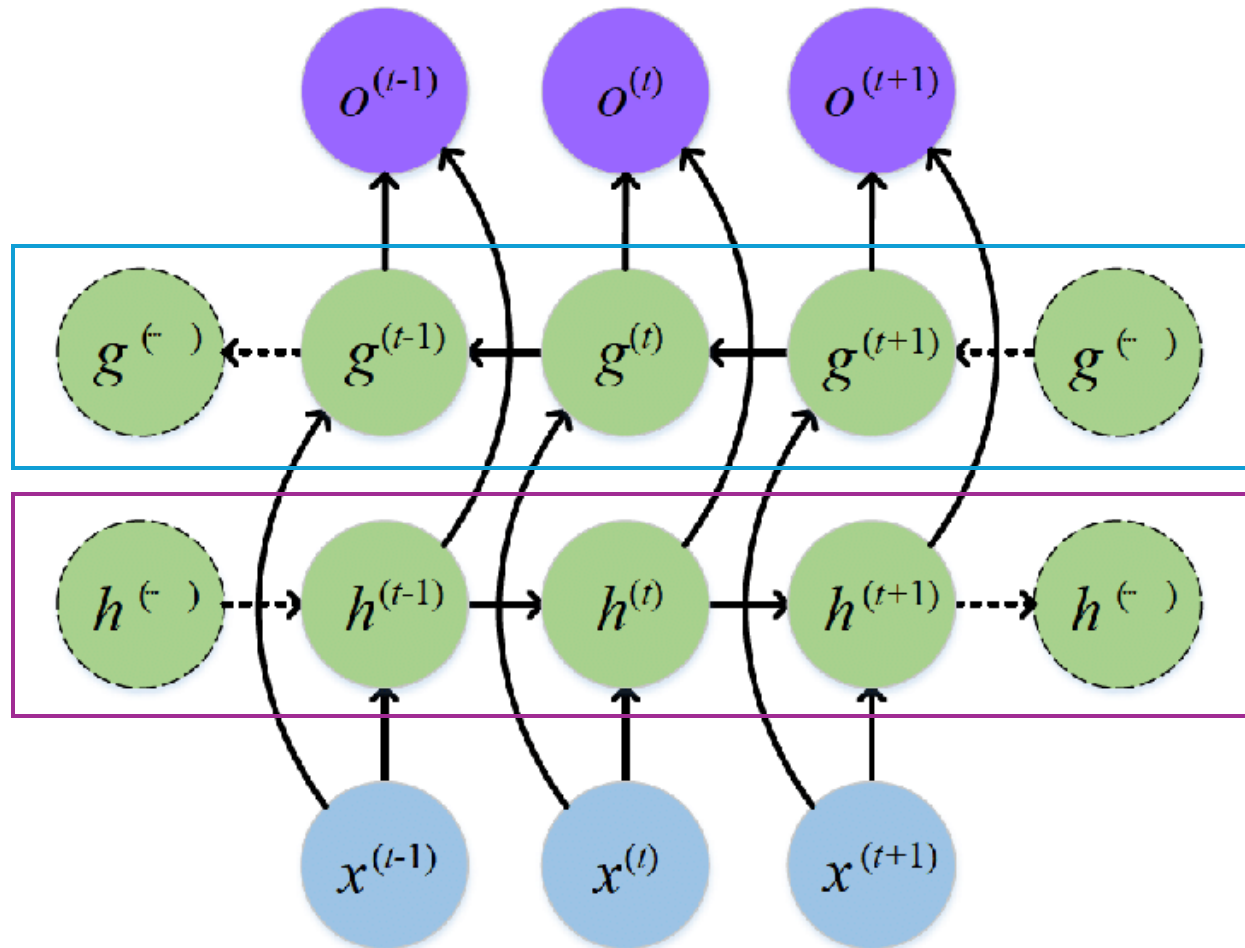
Tipos de estructuras de RNN

Los tipos que veremos además del vainilla ya visto, son..

1. **Bidireccional:** Los datos se procesan en dos direcciones con capas tanto hacia delante como hacia atrás para considerar los contextos **pasados y futuros**. La combinación de ambas capas permite que tenga una precisión de predicción mejorada en algunos casos de uso
2. **Long Short Term Memory (LSTM):** Un modelo puede ampliar su capacidad de memoria para adaptarse a una línea de tiempo más larga. Tiene un bloque de memoria especial (celdas) que está controlado por **una puerta de entrada, una puerta de salida y una puerta de olvido**, por lo que LSTM puede recordar información más útil que la RNN vainilla.
3. **Gated Recurrent Unit (GRU):** Permite la retención selectiva de la memoria. El modelo tiene una **puerta de actualización y olvido** que puede almacenar o eliminar información en la memoria.

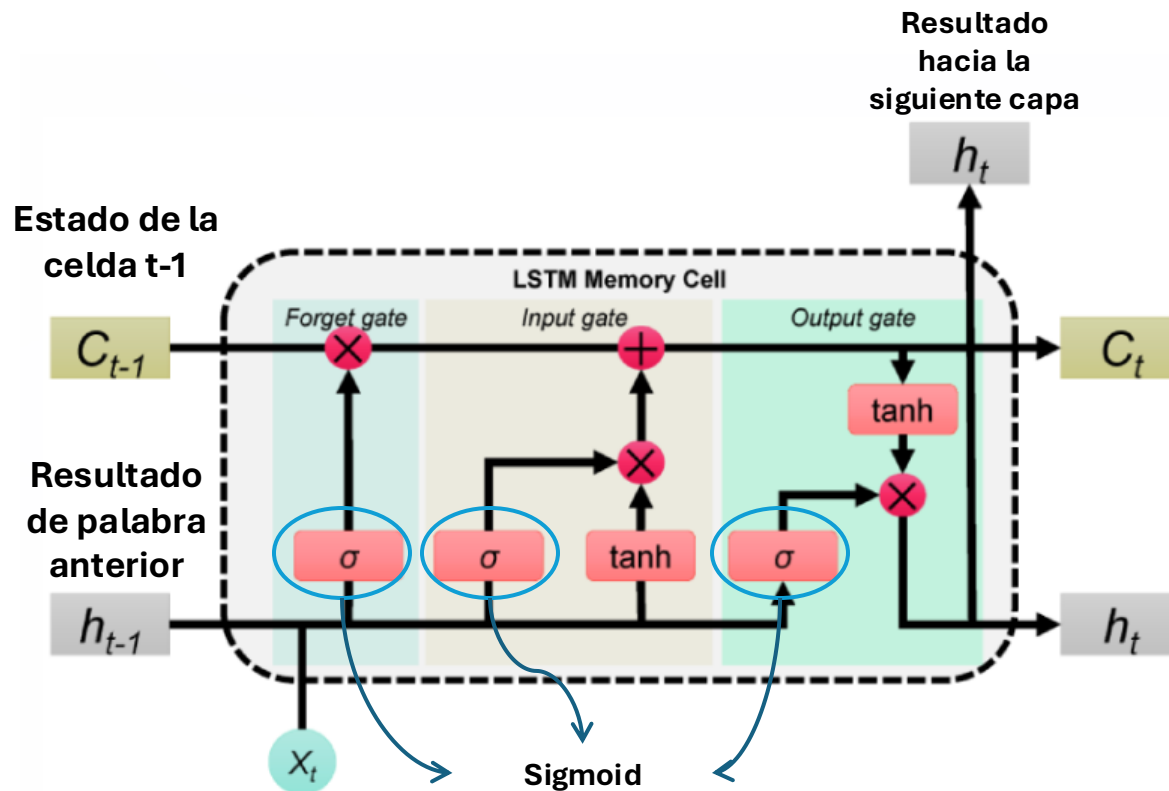


Bidireccional (BRNN)



- Conecta dos capas ocultas de **direcciones opuestas** a la misma salida. Concatena ambos resultados para enviarlos a la siguiente capa
- Se introdujeron para **aumentar la cantidad de información** de entrada disponible para la red.
- Las vainilla RNN tienen restricciones ya que no se puede acceder a la información de entrada futura desde el estado actual.
- Las BRNN son **especialmente útiles cuando se necesita conocer el contexto de la entrada**. Por ejemplo, en el reconocimiento de **escritura a mano**, el rendimiento se puede mejorar si se conocen las letras que se encuentran antes y después de la letra actual.

LSTM



- Es un tipo de red neuronal recurrente (RNN) cuyo objetivo es mitigar el problema del gradiente de desaparición que suelen encontrar las RNN tradicionales
- Tiene la capacidad de recordar dependencias a largo plazo
- Cada unidad recurrente contiene un estado de celda y tres tipos de puertas: **de entrada, de olvido y de salida.**
- La **puerta de entrada (input)** controla el flujo de nueva información hacia el estado de celda,
- La de **olvido (forget)** controla el flujo de información que ya no es relevante.
- La **puerta de salida (output)** controla el flujo de información desde el estado de celda hasta la salida de la unidad.

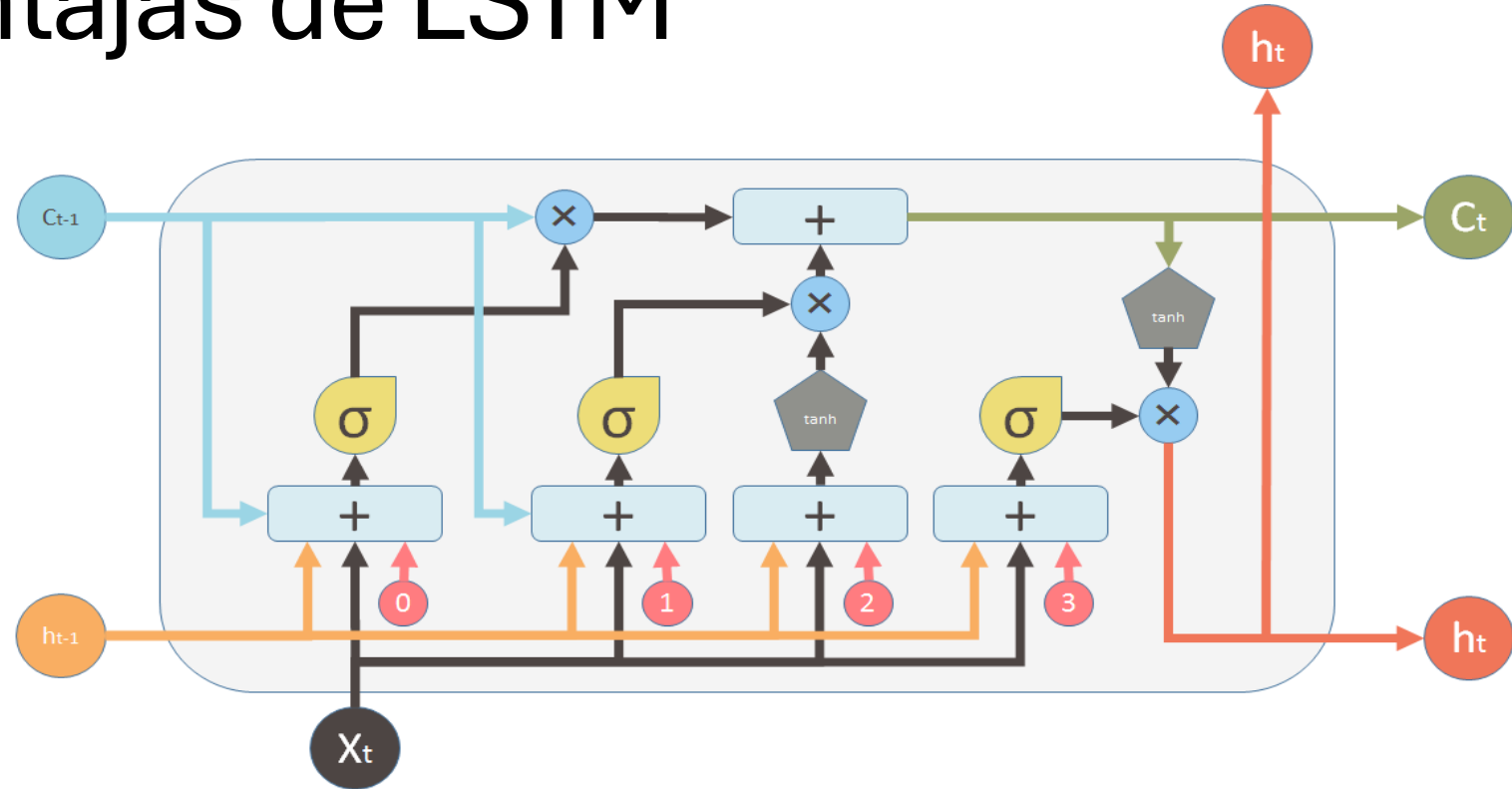
Ventajas y desventajas de LSTM

Ventajas:

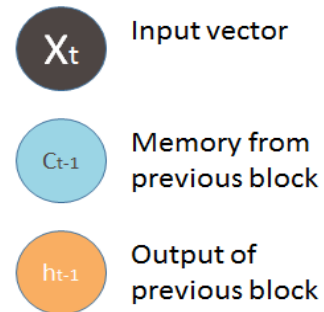
- **Capacidad para procesar datos secuenciales**
- **Capacidad para manejar dependencias a largo plazo:** los LSTM están diseñados específicamente para abordar el problema de los gradientes que desaparecen. Esto los hace adecuados para tareas que requieren procesar dependencias a largo plazo, como predecir precios de acciones o patrones climáticos.
- **Celda de memoria:** la celda de memoria en un LSTM permite que la red recuerde u olvide información de manera selectiva durante largos períodos de tiempo.

Desventajas:

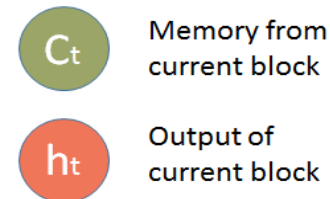
- **Complejidad de entrenamiento:** los LSTM son más complejos que las RNN tradicionales.
- **Sobreajuste:** los LSTM son propensos al sobreajuste, especialmente cuando trabajan con conjuntos de datos pequeños.
- **Costo computacional:** los LSTM requieren más recursos computacionales que las RNN tradicionales, lo que puede hacer que sean más lentos y más costosos de entrenar.



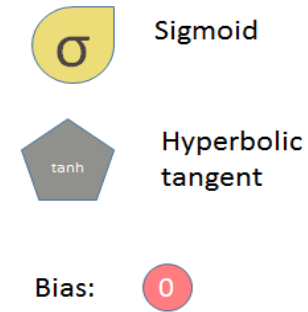
Inputs:



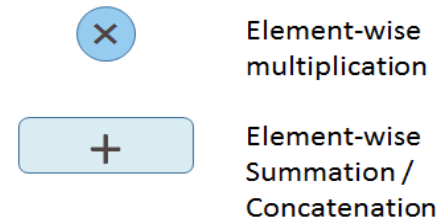
outputs:



Nonlinearities:

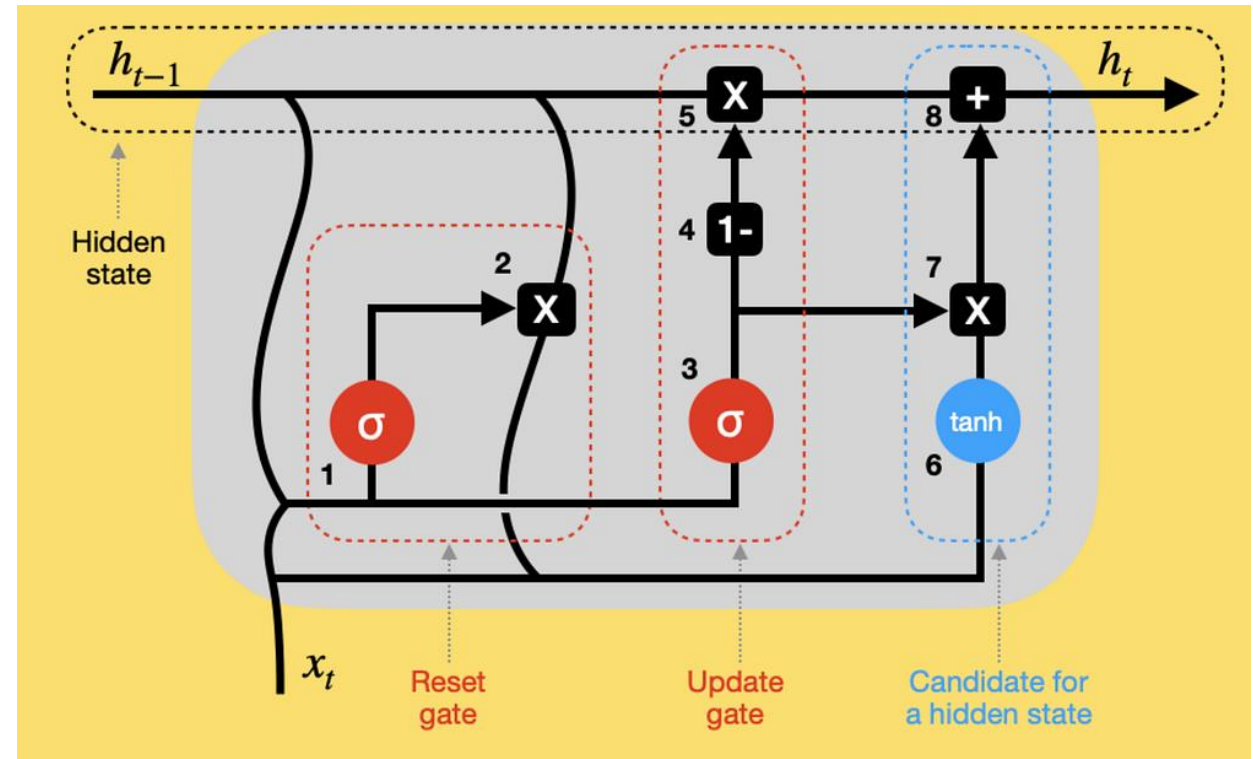


Vector operations:

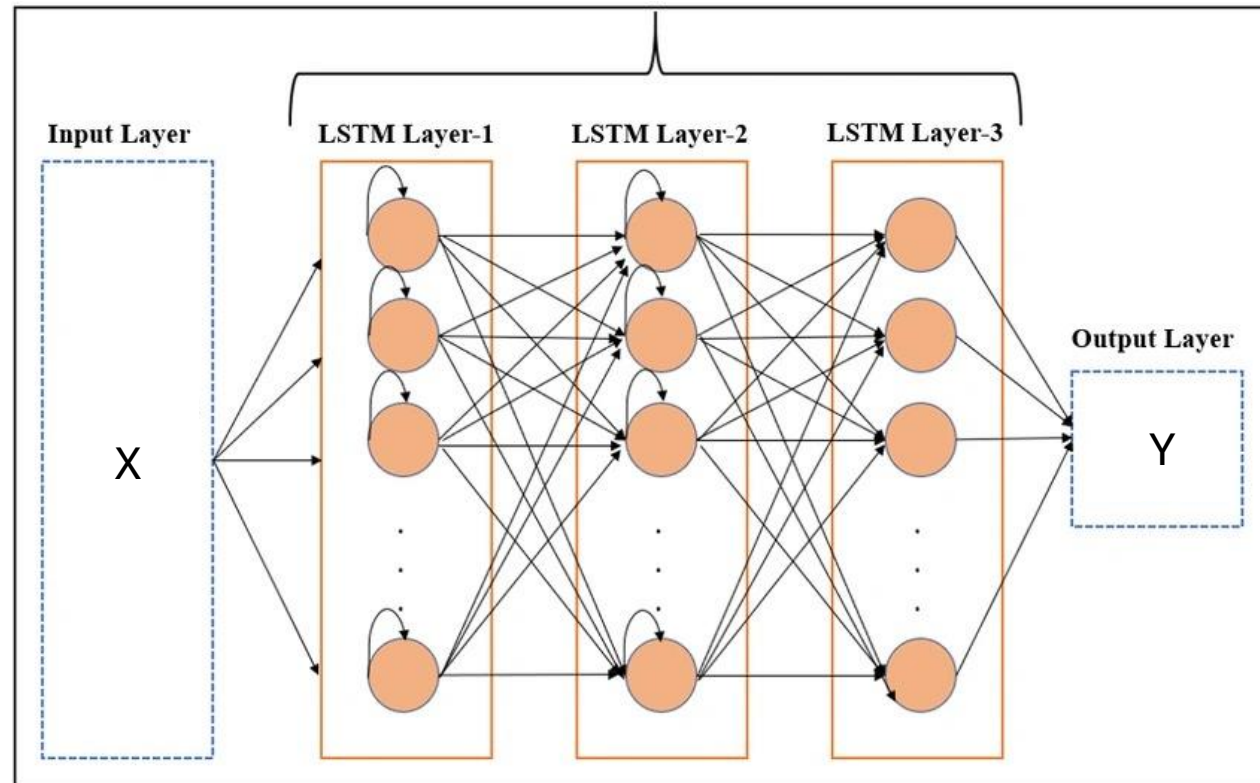


Gated Recurrent Unit (GRU)

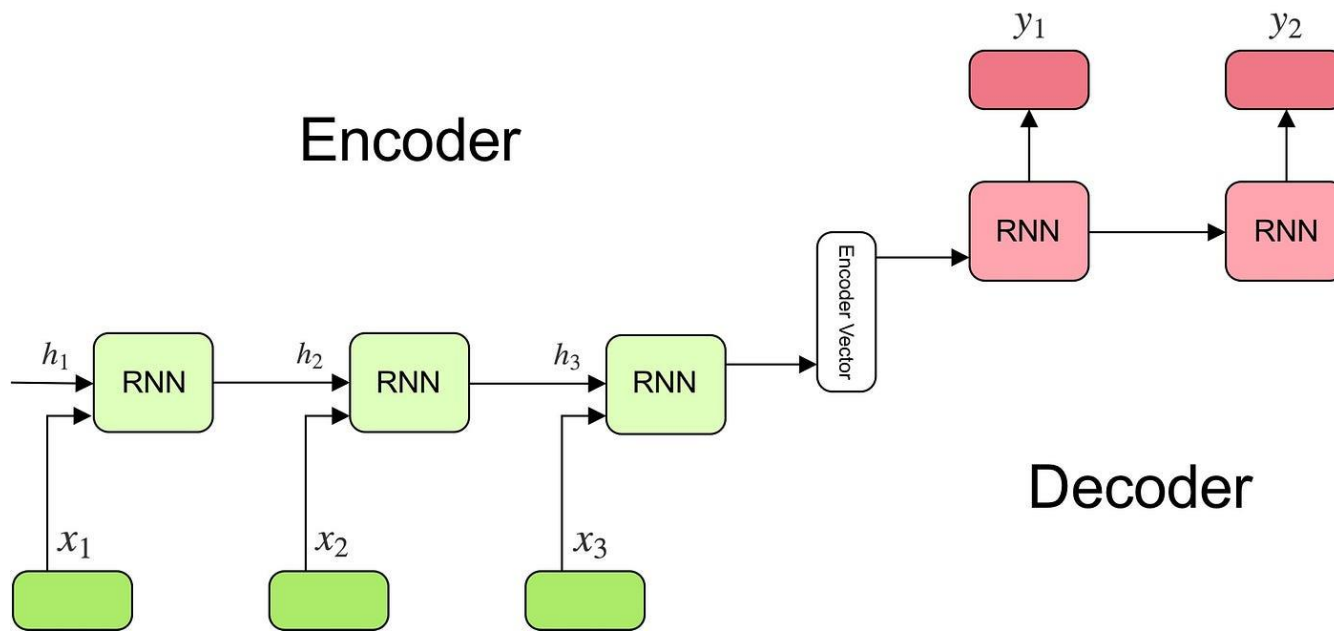
- GRU es como una LSTM con un mecanismo de puertas para ingresar u olvidar ciertas características, pero **carece de un vector de contexto y** puerta de salida, lo que resulta en **menos parámetros** que la LSTM
- Para resolver el problema del gradiente de desaparición de una red neuronal convencional, GRU utiliza las llamadas puertas de **actualización (update)** y de **reinicio (reset)**.
- **La puerta de actualización controla qué parte del estado oculto anterior** (es decir, la información anterior) **debe transferirse al estado oculto actual**. En otras palabras, determina hasta qué punto la unidad actualiza su contenido con nueva información.
- **La puerta de reinicio controla cuánto del estado oculto anterior debe olvidarse** durante el paso de tiempo actual



Podemos tener múltiples capas de estas unidades

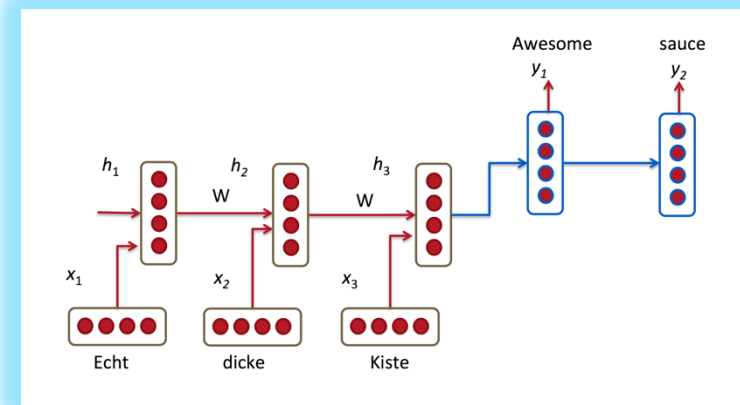


Arquitectura RNN de Codificador- Decodificador

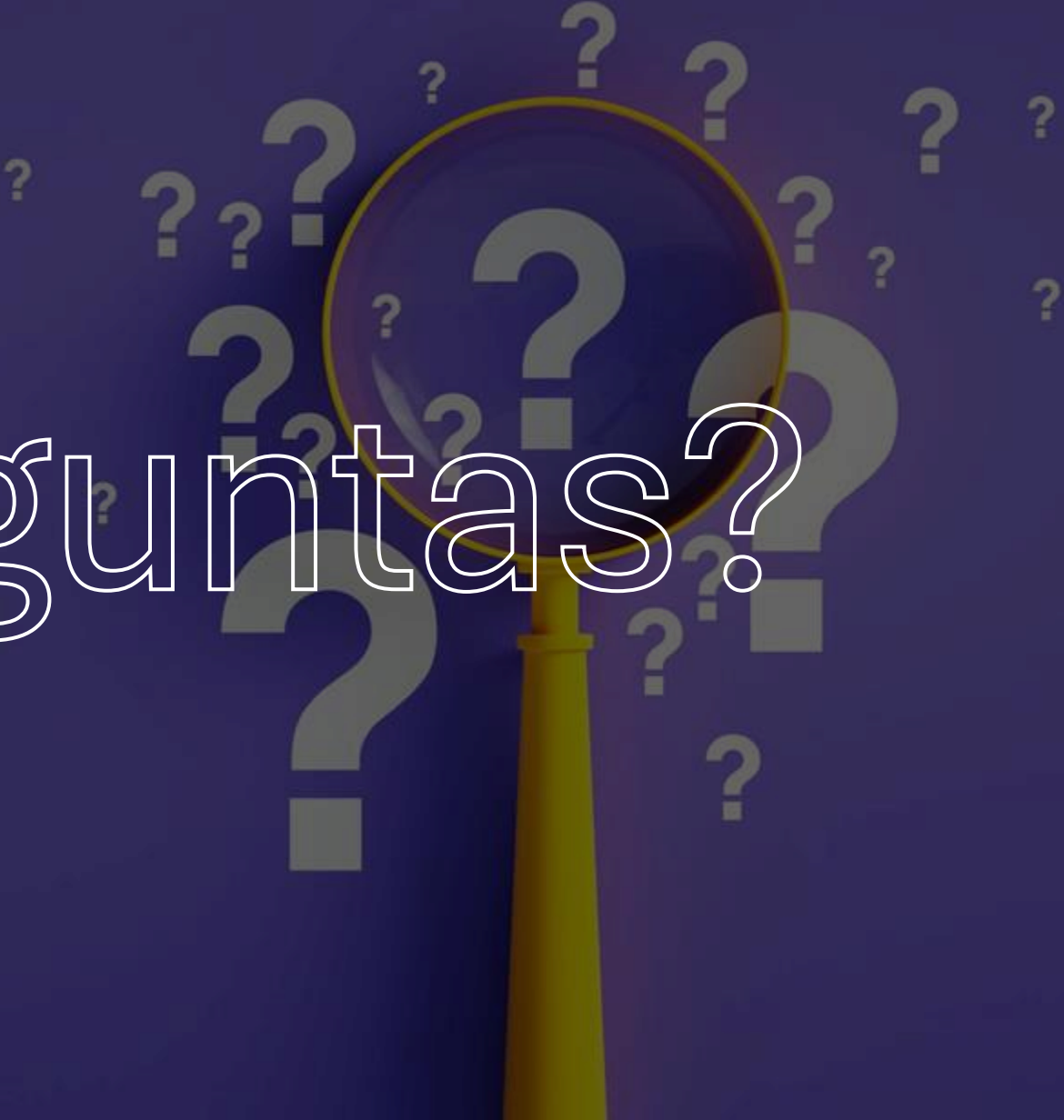


- La arquitectura codificador-decodificador para redes neuronales recurrentes es el método estándar de **traducción automática neuronal y seq2seq**.
- La arquitectura fue propuesta en **2014** pero ya es un estándar para muchos problemas
- Están diseñadas para **gestionar secuencias de variable longitud** procesando toda la secuencia de entrada en el codificador, que la condensa en un **vector de contexto de longitud fija (o estado oculto)** que se pasa al decodificador. El decodificador luego genera una secuencia de salida de cualquier longitud.
- El vector de contexto captura información sobre toda la secuencia, lo que permite que el **modelo conserve y procese detalles tanto de los elementos anteriores como de los posteriores**. Luego el decodificador expande esta representación en una secuencia de salida.

Limitaciones de RNNs



- El uso de RNN ha disminuido en inteligencia artificial, especialmente en favor de arquitecturas como los modelos **transformers**, pero no están obsoletas.
- Sin embargo, la debilidad de las RNN ante los **problemas de gradiente evanescente y explosivo**, junto con el auge de los modelos de transformadores como BERT y GPT, han resultado en este declive. Los **transformadores pueden capturar dependencias de largo alcance de manera mucho más efectiva y** son más fáciles de **paralelizar**.
- Dicho esto, las RNN aún **se utilizan en contextos específicos** donde su naturaleza secuencial y mecanismo de memoria pueden ser útiles, especialmente en **entornos más pequeños** y **con recursos limitados** o para tareas donde el procesamiento de datos se beneficia de la recurrencia paso a paso.



Preguntas?