Universitat Oberta de Catalunya
Universitat de Barcelona
MÁSTER UNIVERSITARIO EN BIOINFORMÁTICA Y BIOESTADÍSTICA

Machine Learning

PEC2: Actividad y Debate

Diciembre de 2023

Alumno: Demetrio Muñoz Álvarez

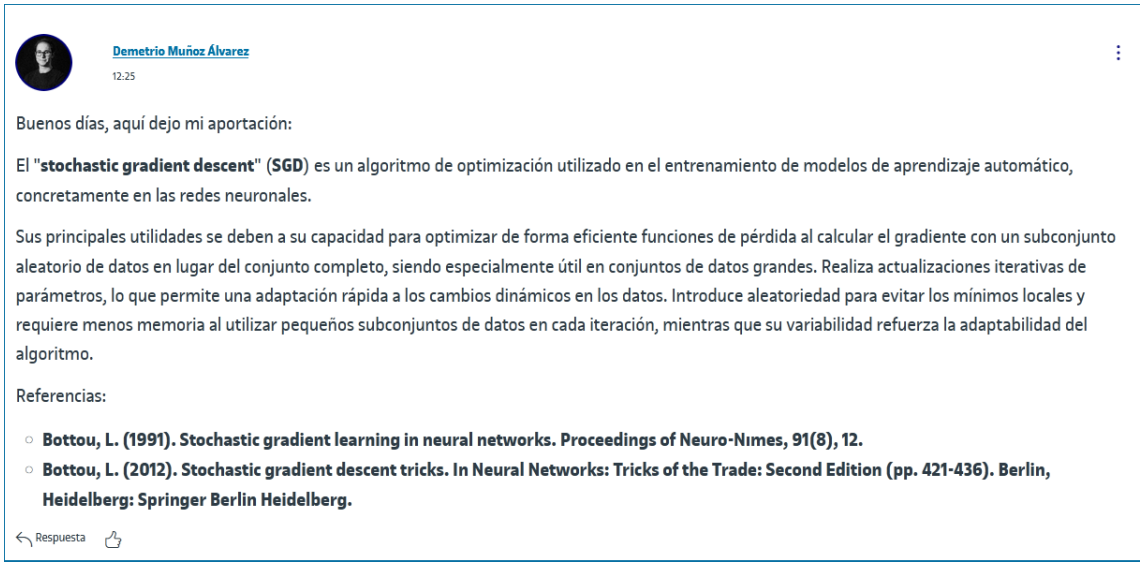
¿Cuál es la utilidad del “stochastic gradient learning”?

El "**stochastic gradient descent**" (**SGD**) es un algoritmo de optimización utilizado en el entrenamiento de modelos de aprendizaje automático, concretamente en las redes neuronales.

Sus principales utilidades se deben a su capacidad para optimizar de forma eficiente funciones de pérdida al calcular el gradiente con un subconjunto aleatorio de datos en lugar del conjunto completo, siendo especialmente útil en conjuntos de datos grandes. Realiza actualizaciones iterativas de parámetros, lo que permite una adaptación rápida a los cambios dinámicos en los datos. Introduce aleatoriedad para evitar los mínimos locales y requiere menos memoria al utilizar pequeños subconjuntos de datos en cada iteración, mientras que su variabilidad refuerza la adaptabilidad del algoritmo.

Referencias:

- **Bottou, L. (1991). Stochastic gradient learning in neural networks. Proceedings of Neuro-Nımes, 91(8), 12.**
- **Bottou, L. (2012). Stochastic gradient descent tricks. In Neural Networks: Tricks of the Trade: Second Edition (pp. 421-436). Berlin, Heidelberg: Springer Berlin Heidelberg.**



Demetrio Muñoz Álvarez
12:25

Buenos días, aquí dejo mi aportación:

El "**stochastic gradient descent**" (**SGD**) es un algoritmo de optimización utilizado en el entrenamiento de modelos de aprendizaje automático, concretamente en las redes neuronales.

Sus principales utilidades se deben a su capacidad para optimizar de forma eficiente funciones de pérdida al calcular el gradiente con un subconjunto aleatorio de datos en lugar del conjunto completo, siendo especialmente útil en conjuntos de datos grandes. Realiza actualizaciones iterativas de parámetros, lo que permite una adaptación rápida a los cambios dinámicos en los datos. Introduce aleatoriedad para evitar los mínimos locales y requiere menos memoria al utilizar pequeños subconjuntos de datos en cada iteración, mientras que su variabilidad refuerza la adaptabilidad del algoritmo.

Referencias:

- **Bottou, L. (1991). Stochastic gradient learning in neural networks. Proceedings of Neuro-Nımes, 91(8), 12.**
- **Bottou, L. (2012). Stochastic gradient descent tricks. In Neural Networks: Tricks of the Trade: Second Edition (pp. 421-436). Berlin, Heidelberg: Springer Berlin Heidelberg.**

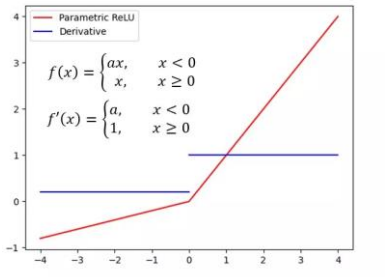
Respuesta

¿Cuál es la función ReLU? y ¿Cuál es su utilidad?

La función **ReLU** (Rectified Linear Unit) es una función de activación común en redes neuronales y otros modelos de aprendizaje automático. Su definición y comportamiento es simple, si la entrada es positiva, devuelve la misma entrada; si es negativa, devuelve cero:

$$f(x)=max\ (0, x)$$

Devuelve "x" si "x" es positivo o cero, y cero si "x" es negativo. Gráficamente, se puede visualizar como una rampa que pasa a través del origen:



En general, **ReLU** activa las neuronas cuando la entrada es positiva y las desactiva cuando la entrada es negativa. Lo que ayuda a introducir no linealidades en el modelo y es una función efectiva en el entrenamiento de redes neuronales profundas.

La utilidad de la función **ReLU** viene dade de su capacidad de no saturar en las regiones positivas, lo que evita imponer penalizaciones en los gradientes positivos y facilita el entrenamiento. Su eficiencia computacional se destaca gracias a su cálculo simple, asignando cero a valores negativos. Además, introduce esparsidad en las activaciones, mejorando la eficiencia y generalización del modelo al inactivar unidades. **ReLU** puede tener propiedades de equivariancia a la intensidad, particularmente útiles en el reconocimiento de objetos. Además, tanto ReLU como sus variantes demuestran un mejor desempeño en diversas tareas, como la clasificación de imágenes, lo que contribuye al éxito de las arquitecturas de redes neuronales profundas.

Podemos destacar que también encontramos algunas desventajas como la posibilidad de introducir solo valores "**0**" (***dying ReLU***) y la sensibilidad a valores atípicos.

- **Nair, V., & Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)* (pp. 807-814).**

- **Lu, L., Shin, Y., Su, Y., & Karniadakis, G. E. (2019). Dying relu and initialization: Theory and numerical examples. *arXiv preprint arXiv:1903.06733*.**

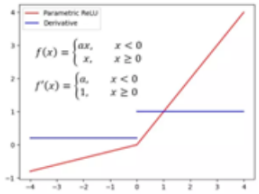
Demetrio Muñoz Álvarez
11:54

Buenos días, aquí dejo mi aportación:

La función **ReLU** (Rectified Linear Unit) es una función de activación común en redes neuronales y otros modelos de aprendizaje automático. Su definición y comportamiento es simple, si la entrada es positiva, devuelve la misma entrada; si es negativa, devuelve cero:

f(x)=max (0, x)

Devuelve "x" si "x" es positivo o cero, y cero si "x" es negativo. Gráficamente, se puede visualizar como una rampa que pasa a través del origen:



En general, **ReLU** activa las neuronas cuando la entrada es positiva y las desactiva cuando la entrada es negativa. Lo que ayuda a introducir no linealidades en el modelo y es una función efectiva en el entrenamiento de redes neuronales profundas.

La utilidad de la función **ReLU** viene dade de su capacidad de no saturar en las regiones positivas, lo que evita imponer penalizaciones en los gradientes positivos y facilita el entrenamiento. Su eficiencia computacional se destaca gracias a su cálculo simple, asignando cero a valores negativos. Además, introduce esparsidad en las activaciones, mejorando la eficiencia y generalización del modelo al inactivar unidades. **ReLU** puede tener propiedades de equivariancia a la intensidad, particularmente útiles en el reconocimiento de objetos. Además, tanto ReLU como sus variantes demuestran un mejor desempeño en diversas tareas, como la clasificación de imágenes, lo que contribuye al éxito de las arquitecturas de redes neuronales profundas.

Podemos destacar que también encontramos algunas desventajas como la posibilidad de introducir solo valores "**0**" (***dying ReLU***) y la sensibilidad a valores atípicos.

Referencias:

- **Nair, V., & Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)* (pp. 807-814).**
- **Lu, L., Shin, Y., Su, Y., & Karniadakis, G. E. (2019). Dying relu and initialization: Theory and numerical examples. *arXiv preprint arXiv:1903.06733*.**

Respuesta

¿Qué son las Memory Neural Networks? Descripción. Propósito y Aplicaciones.

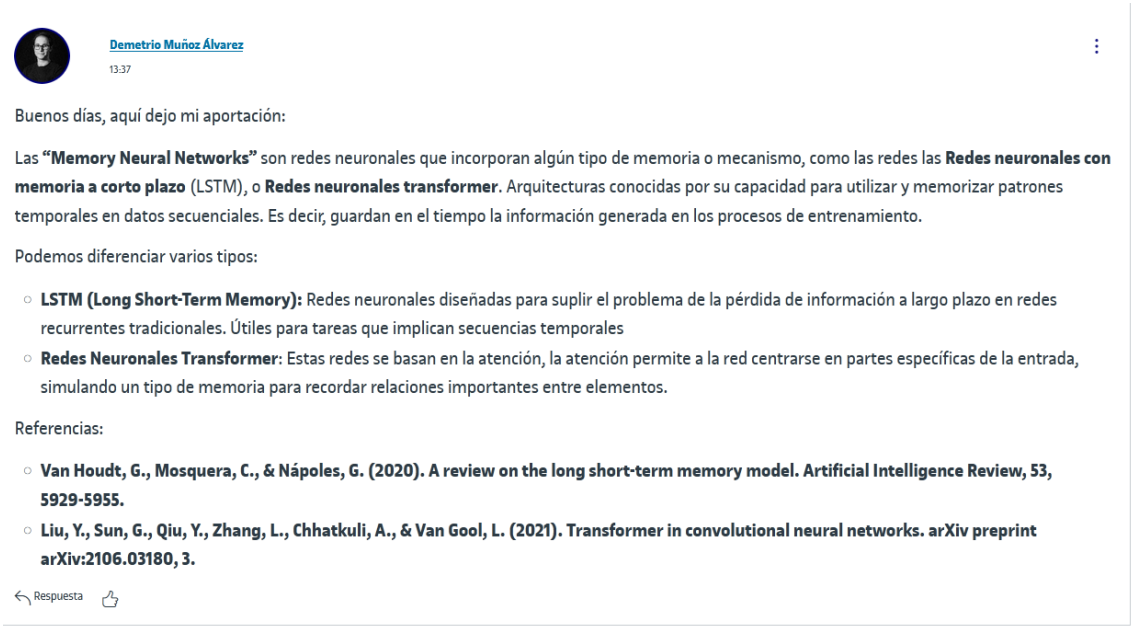
Las **"Memory Neural Networks"** son redes neuronales que incorporan algún tipo de memoria o mecanismo, como las redes las **Redes neuronales con memoria a corto plazo** (LSTM), o **Redes neuronales transformer**. Arquitecturas conocidas por su capacidad para utilizar y memorizar patrones temporales en datos secuenciales. Es decir, guardan en el tiempo la información generada en los procesos de entrenamiento.

Podemos diferenciar varios tipos:

- **LSTM (Long Short-Term Memory):** Redes neuronales diseñadas para suplir el problema de la pérdida de información a largo plazo en redes recurrentes tradicionales. Útiles para tareas que implican secuencias temporales
- **Redes Neuronales Transformer**: Estas redes se basan en la atención, la atención permite a la red centrarse en partes específicas de la entrada, simulando un tipo de memoria para recordar relaciones importantes entre elementos.

Referencias:

- **Van Houdt, G., Mosquera, C., & Nápoles, G. (2020). A review on the long short-term memory model. Artificial Intelligence Review, 53, 5929-5955.**
- **Liu, Y., Sun, G., Qiu, Y., Zhang, L., Chhatkuli, A., & Van Gool, L. (2021). Transformer in convolutional neural networks. arXiv preprint arXiv:2106.03180, 3.**



Demetrio Muñoz Álvarez
13:37

Buenos días, aquí dejo mi aportación:

Las **"Memory Neural Networks"** son redes neuronales que incorporan algún tipo de memoria o mecanismo, como las redes las **Redes neuronales con memoria a corto plazo** (LSTM), o **Redes neuronales transformer**. Arquitecturas conocidas por su capacidad para utilizar y memorizar patrones temporales en datos secuenciales. Es decir, guardan en el tiempo la información generada en los procesos de entrenamiento.

Podemos diferenciar varios tipos:

- **LSTM (Long Short-Term Memory):** Redes neuronales diseñadas para suplir el problema de la pérdida de información a largo plazo en redes recurrentes tradicionales. Útiles para tareas que implican secuencias temporales
- **Redes Neuronales Transformer**: Estas redes se basan en la atención, la atención permite a la red centrarse en partes específicas de la entrada, simulando un tipo de memoria para recordar relaciones importantes entre elementos.

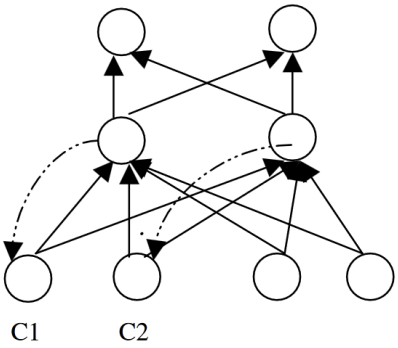
Referencias:

- **Van Houdt, G., Mosquera, C., & Nápoles, G. (2020). A review on the long short-term memory model. Artificial Intelligence Review, 53, 5929-5955.**
- **Liu, Y., Sun, G., Qiu, Y., Zhang, L., Chhatkuli, A., & Van Gool, L. (2021). Transformer in convolutional neural networks. arXiv preprint arXiv:2106.03180, 3.**

Respuesta

¿Qué son las Recurrent Neural Networks? Descripción. Propósito y Aplicaciones.

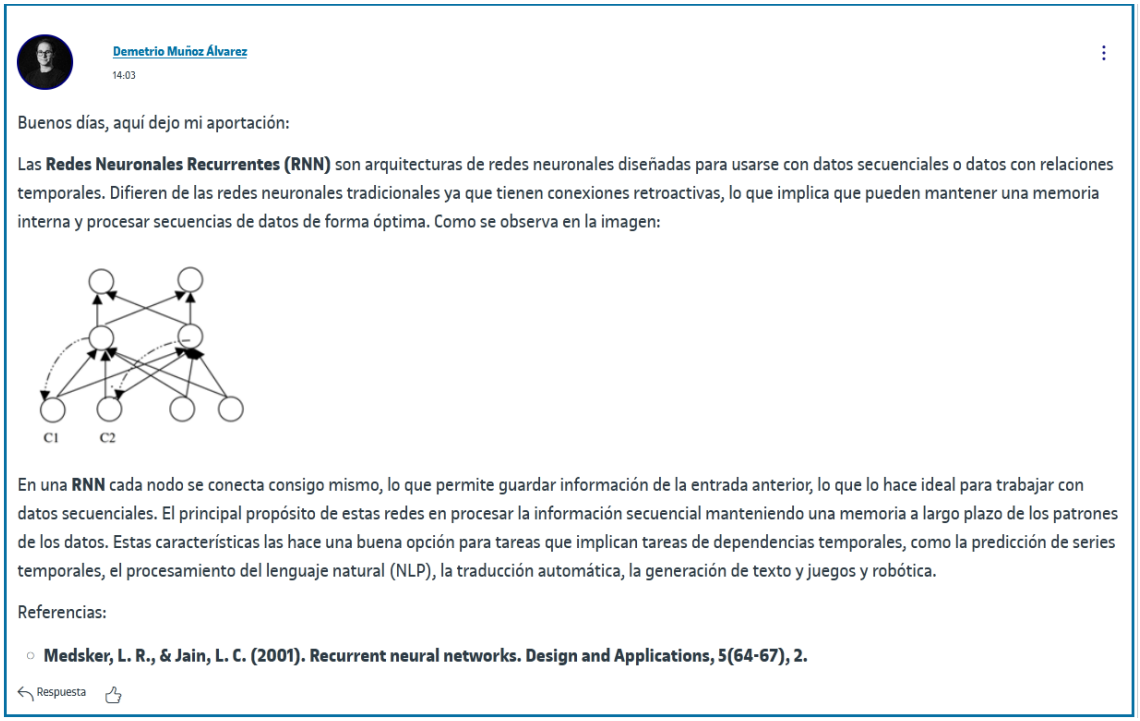
Las **Redes Neuronales Recurrentes (RNN)** son arquitecturas de redes neuronales diseñadas para usarse con datos secuenciales o datos con relaciones temporales. Difieren de las redes neuronales tradicionales ya que tienen conexiones retroactivas, lo que implica que pueden mantener una memoria interna y procesar secuencias de datos de forma óptima. Como se observa en la imagen:



En una **RNN** cada nodo se conecta consigo mismo, lo que permite guardar información de la entrada anterior, lo que lo hace ideal para trabajar con datos secuenciales. El principal propósito de estas redes en procesar la información secuencial manteniendo una memoria a largo plazo de los patrones de los datos. Estas características las hace una buena opción para tareas que implican tareas de dependencias temporales, como la predicción de series temporales, el procesamiento del lenguaje natural **(NLP)**, la traducción automática, la generación de texto y juegos y robótica.

Referencias:

- **Medsker, L. R., & Jain, L. C. (2001). Recurrent neural networks. Design and Applications, 5(64-67), 2.**



Demetrio Muñoz Álvarez
14:03

Buenos días, aquí dejo mi aportación:

Las **Redes Neuronales Recurrentes (RNN)** son arquitecturas de redes neuronales diseñadas para usarse con datos secuenciales o datos con relaciones temporales. Difieren de las redes neuronales tradicionales ya que tienen conexiones retroactivas, lo que implica que pueden mantener una memoria interna y procesar secuencias de datos de forma óptima. Como se observa en la imagen:

En una **RNN** cada nodo se conecta consigo mismo, lo que permite guardar información de la entrada anterior, lo que lo hace ideal para trabajar con datos secuenciales. El principal propósito de estas redes en procesar la información secuencial manteniendo una memoria a largo plazo de los patrones de los datos. Estas características las hace una buena opción para tareas que implican tareas de dependencias temporales, como la predicción de series temporales, el procesamiento del lenguaje natural (NLP), la traducción automática, la generación de texto y juegos y robótica.

Referencias:

- **Medsker, L. R., & Jain, L. C. (2001). Recurrent neural networks. Design and Applications, 5(64-67), 2.**

Respuesta

¿Qué son las Autoencoders Neural Networks? Descripción. Propósito y Aplicaciones.

Los **Autoencoders**, son arquitecturas de aprendizaje profundo utilizadas principalmente para el aprendizaje no supervisado. Se utilizan para la compresión de datos y la obtención de abstracciones útiles de la información. Además, autoencoders pueden aprender modelos con el fin de generar nueva información, como los autoencoders de eliminación de ruido. También pueden aplicarse a datos genéricos o de imágenes, incluso en forma de **Redes Neuronales Convolucionales (CNN)**.

La arquitectura de un **Autoencoder** consta de una red de codificador y decodificador, se asemeja a cuello de botella, donde la entrada se reduce a una representación más compacta (**encoder**) y luego se expande (**decoder**) de nuevo:

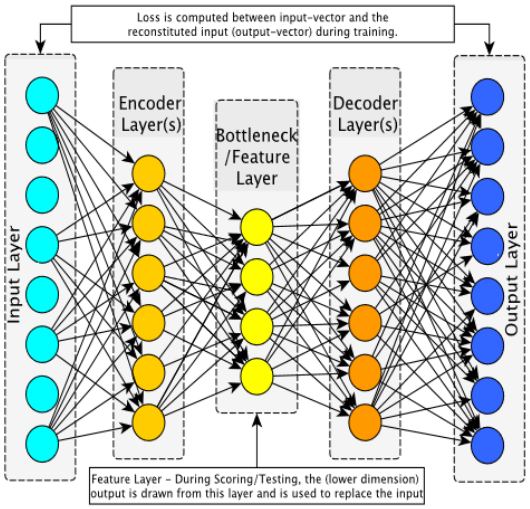


Figure 2: A schematic of an AE showing an encoder connected to the input layer leading to bottleneck layer/ features followed by decoder leading to the reconstituted output layer.

En resumen, los **Autoencoders** tienen como propósito aprender representaciones eficientes de los datos. Forzando al modelo a aprender a reconstruir los datos de entrada a partir de una representación reducida, capturando características importantes y eliminando redundancias.

Se pueden aplicar a procesos como: la reducción de dimensionalidad, eliminación de ruido, detección de datos y patrones anómalos o la generación de datos similares a los del entrenamiento.

Referencias:

- **Sewak, M., Sahay, S. K., & Rathore, H. (2020). An overview of deep learning architecture of deep neural networks and autoencoders. Journal of Computational and Theoretical Nanoscience, 17(1), 182-188.**

Demetrio Muñoz Álvarez

14:23

Buenos días, aquí dejo mi aportación:

Los **Autoencoders**, son arquitecturas de aprendizaje profundo utilizadas principalmente para el aprendizaje no supervisado. Se utilizan para la compresión de datos y la obtención de abstracciones útiles de la información. Además, autoencoders pueden aprender modelos con el fin de generar nueva información, como los autoencoders de eliminación de ruido. También pueden aplicarse a datos genéricos o de imágenes, incluso en forma de **Redes Neuronales Convolucionales (CNN)**.

La arquitectura de un **Autoencoder** consta de una red de codificador y decodificador, se asemeja a cuello de botella, donde la entrada se reduce a una representación más compacta (**encoder**) y luego se expande (**decoder**) de nuevo:

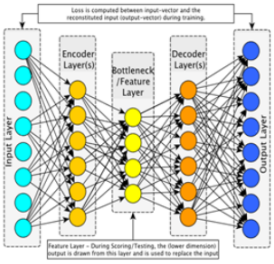


Figure 2: A schematic of an AE showing an encoder connected to the input layer leading to bottleneck layer/ features followed by decoder leading to the reconstituted output layer.

En resumen, los **Autoencoders** tienen como propósito aprender representaciones eficientes de los datos. Forzando al modelo a aprender a reconstruir los datos de entrada a partir de una representación reducida, capturando características importantes y eliminando redundancias.

Se pueden aplicar a procesos como: la reducción de dimensionalidad, eliminación de ruido, detección de datos y patrones anómalos o la generación de datos similares a los del entrenamiento.

Referencias:

- **Sewak, M., Sahay, S. K., & Rathore, H. (2020). An overview of deep learning architecture of deep neural networks and autoencoders. Journal of Computational and Theoretical Nanoscience, 17(1), 182-188.**

Respuesta