Buzea en el Aprendizaje Profundo

Escuela de Verano 2024

Feliú Sagols Troncoso

Departamento de Matemáticas
Cinvestav-IPN, Mexico City

Julio 25 2024

Tabla de Contenido

Introducción

Introducción

- La Inteligencia Artificial y áreas afines como el Aprendizaje de Máquina, la Ciencia de Datos o el Aprendizaje Reforzado, por mencionar algunas; son actividades que están transformado a la humanidad en su conjunto. Prácticamente ninguna de nuestras actividades escapa a sus alcances. Lo mismo la encontramos en la producción de piezas de arte en todas sus manifestaciones, que en el desarrollo de la ciencia, la política, el entretenimiento, las relaciones comerciales, el desarrollo tecnológico, etc. La liberación de herramientas como ChatGPT 4.0 es solo el inicio de una gran invasión y seguramente habrán muchos cambios y adaptaciones. Se avecina sin lugar a dudas una nueva revolución industrial y tendremos que aprender a convivir con esta nueva forma de tecnología

Introducción

- La Inteligencia Artificial y áreas afines como el Aprendizaje de Máquina, la Ciencia de Datos o el Aprendizaje Reforzado, por mencionar algunas; son actividades que están transformado a la humanidad en su conjunto. Prácticamente ninguna de nuestras actividades escapa a sus alcances. Lo mismo la encontramos en la producción de piezas de arte en todas sus manifestaciones, que en el desarrollo de la ciencia, la política, el entretenimiento, las relaciones comerciales, el desarrollo tecnológico, etc. La liberación de herramientas como ChatGPT 4.0 es solo el inicio de una gran invasión y seguramente habrán muchos cambios y adaptaciones. Se avecina sin lugar a dudas una nueva revolución industrial y tendremos que aprender a convivir con esta nueva forma de tecnología
- La pregunta es ¿por qué ahora?

Introducción

- La Inteligencia Artificial y áreas afines como el Aprendizaje de Máquina, la Ciencia de Datos o el Aprendizaje Reforzado, por mencionar algunas; son actividades que están transformado a la humanidad en su conjunto. Prácticamente ninguna de nuestras actividades escapa a sus alcances. Lo mismo la encontramos en la producción de piezas de arte en todas sus manifestaciones, que en el desarrollo de la ciencia, la política, el entretenimiento, las relaciones comerciales, el desarrollo tecnológico, etc. La liberación de herramientas como ChatGPT 4.0 es solo el inicio de una gran invasión y seguramente habrán muchos cambios y adaptaciones. Se avecina sin lugar a dudas una nueva revolución industrial y tendremos que aprender a convivir con esta nueva forma de tecnología
- La pregunta es ¿por qué ahora?
- En realidad esto se ha venido gestando desde hace varias décadas, pero hay un hecho que marcó un parte aguas en la historia reciente.

Deep Blue derrota al campeón del mundo de ajedrez

Deep Blue derrota al campeón del mundo de ajedrez

- Deep Blue, un supercomputador creado por IBM, venció al campeón mundial de ajedrez Garry Kasparov en una partida el 10 de febrero de 1996. Sin embargo, Kasparov ganó el match de 6 partidas con un marcador de 4-2.

Deep Blue derrota al campeón del mundo de ajedrez

- Deep Blue, un supercomputador creado por IBM, venció al campeón mundial de ajedrez Garry Kasparov en una partida el 10 de febrero de 1996. Sin embargo, Kasparov ganó el match de 6 partidas con un marcador de 4-2.
- El evento que es más recordado ocurrió el siguiente año, en 1997. En un rematch muy anticipado, Deep Blue y Kasparov jugaron un nuevo match de 6 partidas del 3 al 11 de mayo. En esta ocasión, Deep Blue salió victorioso con un marcador de 3.5-2.5 (dos victorias para Deep Blue, una para Kasparov y tres empates). Este fue un hito histórico, ya que fue la primera vez que una máquina derrotó a un campeón mundial en un match a ritmo estándar bajo condiciones de torneo.

Deep Blue derrota al campeón del mundo de ajedrez

- Deep Blue, un supercomputador creado por IBM, venció al campeón mundial de ajedrez Garry Kasparov en una partida el 10 de febrero de 1996. Sin embargo, Kasparov ganó el match de 6 partidas con un marcador de 4-2.
- El evento que es más recordado ocurrió el siguiente año, en 1997. En un rematch muy anticipado, Deep Blue y Kasparov jugaron un nuevo match de 6 partidas del 3 al 11 de mayo. En esta ocasión, Deep Blue salió victorioso con un marcador de 3.5-2.5 (dos victorias para Deep Blue, una para Kasparov y tres empates). Este fue un hito histórico, ya que fue la primera vez que una máquina derrotó a un campeón mundial en un match a ritmo estándar bajo condiciones de torneo.
- La victoria de Deep Blue fue un logro impresionante de la inteligencia artificial y de la computación en general. No solo demostró que las máquinas podían igualar e incluso superar las habilidades de los humanos en tareas complejas como el ajedrez, sino que también demostró los avances en la capacidad de las máquinas para explorar y analizar una vasta cantidad de posibilidades y para seleccionar la mejor acción basándose en esa exploración.

Deep Blue

Características técnicas de Deep Blue

Deep Blue fue una supercomputadora creada por IBM específicamente para jugar ajedrez a un alto nivel. Su diseño y funcionamiento eran notablemente diferentes a los sistemas de computación general que conocemos hoy.
El sistema estaba basado en una arquitectura paralela de procesamiento masivo, que utilizaba 30 nodos de procesamiento, cada uno con un chip de propósito general RS/6000 y 480 chips especializados para el procesamiento del ajedrez. Estos chips especializados eran capaces de procesar hasta 2 millones de posiciones por segundo cada uno, lo que permitía a Deep Blue explorar hasta 200 millones de posiciones por segundo en total.
El papel del chip RS/6000 era coordinar el trabajo de los chips especializados, decidiendo qué posiciones debían ser examinadas y enviando estas posiciones a los chips especializados para su análisis. Una vez que los chips especializados analizaban las posiciones, los resultados se enviaban de vuelta al RS/6000 para su evaluación final.
En términos de software, Deep Blue utilizaba una combinación de algoritmos de búsqueda y una base de datos de aperturas de ajedrez para planificar sus movimientos. También podía "aprender" de las partidas anteriores para mejorar su rendimiento en partidas futuras.

En otras palabras

En otras palabras

- Deep Blue era una basura.

En otras palabras

- Deep Blue era una basura.
- No desde un punto de vista de Ingeniería. Lo era desde el punto de vista matemático y algorítmico.

En otras palabras

- Deep Blue era una basura.
- No desde un punto de vista de Ingeniería. Lo era desde el punto de vista matemático y algorítmico.
- Esto de analizar 200 millones de posiciones por segundo no es otra cosa sino FUERZA BRUTA.

En otras palabras

- Deep Blue era una basura.
- No desde un punto de vista de Ingeniería. Lo era desde el punto de vista matemático y algorítmico.
- Esto de analizar 200 millones de posiciones por segundo no es otra cosa sino FUERZA BRUTA.
- Obviamente Kasparov pidió la revancha pero IBM de inmediato des-ensamblo Deep Blue. Las acciones de IBM subieron y el futuro de la Inteligencia Artificial quedó asegurado.

La Inteligencia Artificial hoy

La Inteligencia Artificial hoy

- La creciente explosión en el desarrollo de GPUs (Graphical Processing Units) y TPUs (Tensor Processing Units) ha potenciado la producción de supercomputadoras. A su vez estos equipos han hecho posible que muchos algoritmos y métodos puedan ser ejecutados en tiempos razonablemente cortos.

La Inteligencia Artificial hoy

- La creciente explosión en el desarrollo de GPUs (Graphical Processing Units) y TPUs (Tensor Processing Units) ha potenciado la producción de supercomputadoras. A su vez estos equipos han hecho posible que muchos algoritmos y métodos puedan ser ejecutados en tiempos razonablemente cortos.
- Muchos algoritmos y estructuras diseñadas para resolver problemas en Inteligencia Artificial se han podido ejecutar exitosamente para resolver problemas cada vez más desafiantes.

La Inteligencia Artificial hoy

- La creciente explosión en el desarrollo de GPUs (Graphical Processing Units) y TPUs (Tensor Processing Units) ha potenciado la producción de supercomputadoras. A su vez estos equipos han hecho posible que muchos algoritmos y métodos puedan ser ejecutados en tiempos razonablemente cortos.
- Muchos algoritmos y estructuras diseñadas para resolver problemas en Inteligencia Artificial se han podido ejecutar exitosamente para resolver problemas cada vez más desafiantes.
- Sin embargo, lo que realmente está en fondo de la Inteligencia Artificial es la Optimización Combinatoria. Todo problema en IA tiene implícita la solución de algún problema de optimización. Así, si ustedes desean dominar las técnicas de la Inteligencia Artificial deben preocuparse por obtener una formación razonable en Optimización Combinatoria.

Sobre este curso

Sobre este curso

- Es pretencioso intentar dar un curso sobre un tema tan amplio como es la Inteligencia Artificial. En esta sesión veremos una selección de algunos de los temas más relevantes.

Sobre este curso

- Es pretencioso intentar dar un curso sobre un tema tan amplio como es la Inteligencia Artificial. En esta sesión veremos una selección de algunos de los temas más relevantes.
- Para que realmente aprendan los fundamentos del IA es importante practicar con problemas reales y hacer programas. Un curso gratuito y disponible en línea con el que podrían iniciar es "Dive into Deep Learning" y la liga es https://d2l.ai/index.html. En este curso encontrarán prácticas en python, bibliotecas de programas de gran utilidad creadas por los autores del curso y una gran cantidad de repositorios de datos para practicar.

Sobre este curso

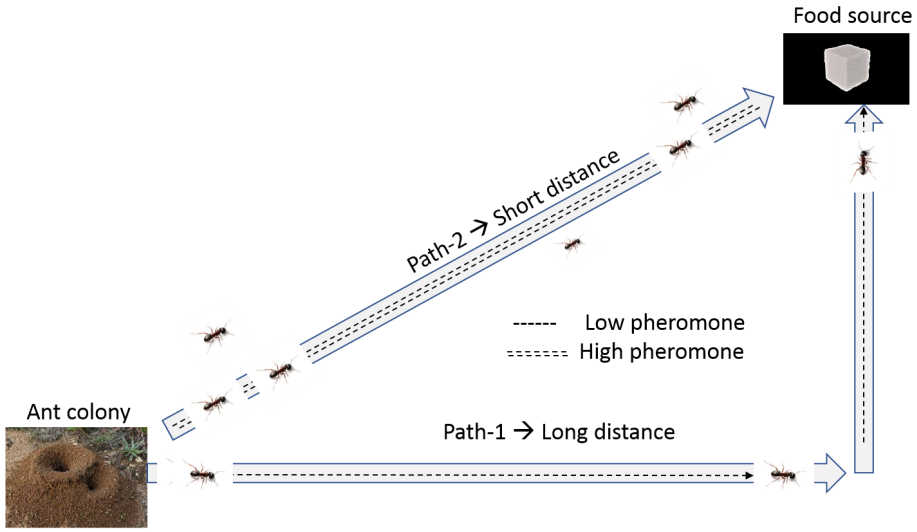
- Es pretencioso intentar dar un curso sobre un tema tan amplio como es la Inteligencia Artificial. En esta sesión veremos una selección de algunos de los temas más relevantes.
- Para que realmente aprendan los fundamentos del IA es importante practicar con problemas reales y hacer programas. Un curso gratuito y disponible en línea con el que podrían iniciar es "Dive into Deep Learning" y la liga es https://d2l.ai/index.html. En este curso encontrarán prácticas en python, bibliotecas de programas de gran utilidad creadas por los autores del curso y una gran cantidad de repositorios de datos para practicar.
- Un curso mucho más formal y avanzado es "Machine Learning with Python-From Linear Models to Deep Learning" que ofrece el MIT por medio de la plataforma EDX. Es un curso que pueden tomar gratuito o pueden pagarlo. Normalmente se abre durante los meses de marzo de cada año. La liga es https://learning.edx.org/course/course-v1:MITx+6.86x+1T2023/home

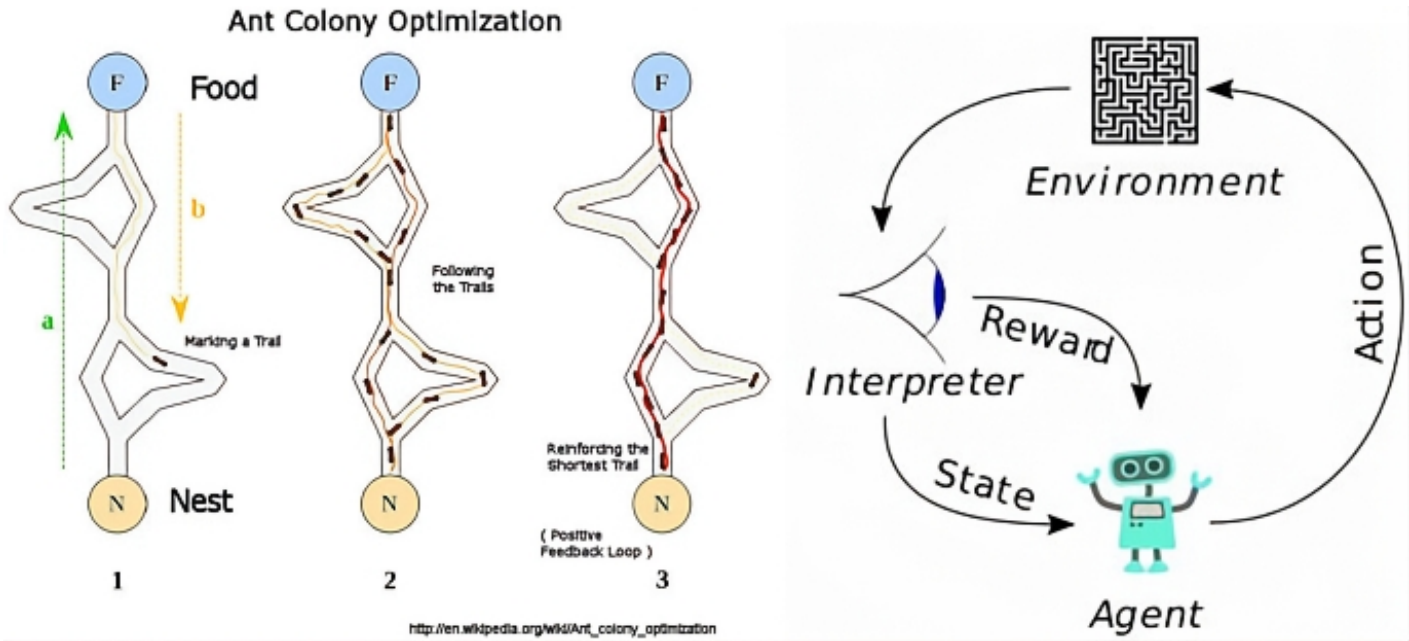
Tabla de Contenido

El Perceptrón

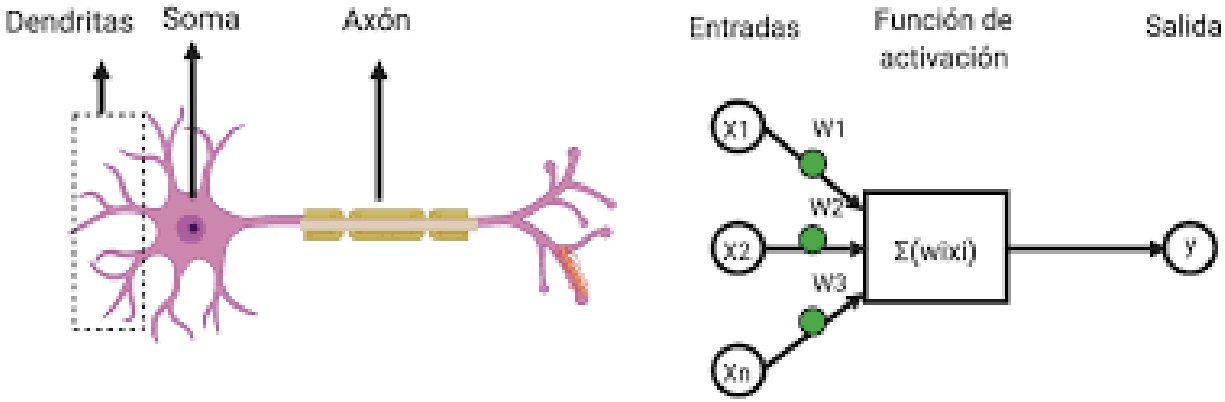
- La forma como aprendemos los seres vivos está sujeta a estudio constante. Su análisis, emulación y estudio lo llevan a cabo sistemáticamente disciplinas como la pedagogía o la lingüística, y es importante por el papel primordial que juegan los mecanismos de aprendizaje como manifestación de la inteligencia, Legg y Hutter 2007. La capacidad de aprender es una cualidad inherente a la inteligencia y no exclusiva de los seres humanos, muchas otras criaturas muestran comportamientos basados en procesos de aprendizaje que les ayudan a resolver problemas.
- Un caso interesante se presenta con los llamados animales gregarios, es decir, aquellos que viven en grupos. En ellos es posible observar comportamientos sociales que evidencian procesos de aprendizaje colectivo que originan comportamientos complejos para resolver los problemas de la comunidad. Tomemos a las hormigas como ejemplo, sus grandes comunidades realizan tareas conjuntas, como la búsqueda de comida. En esta actividad todas ellas participan. Quienes cumplen exitosamente su misión recorren muchas veces el mismo camino. En cada vuelta acarrean su alimento a la vez que impregnan la ruta con sus feromonas. Así, los caminos con mayor densidad de feromonas resultan mucho más atractivo a la comunidad en su conjunto, y a un grupo exitoso poco a poco se unen otras hormigas haciéndolo crecer.

Optimización en las colonias de hormigas

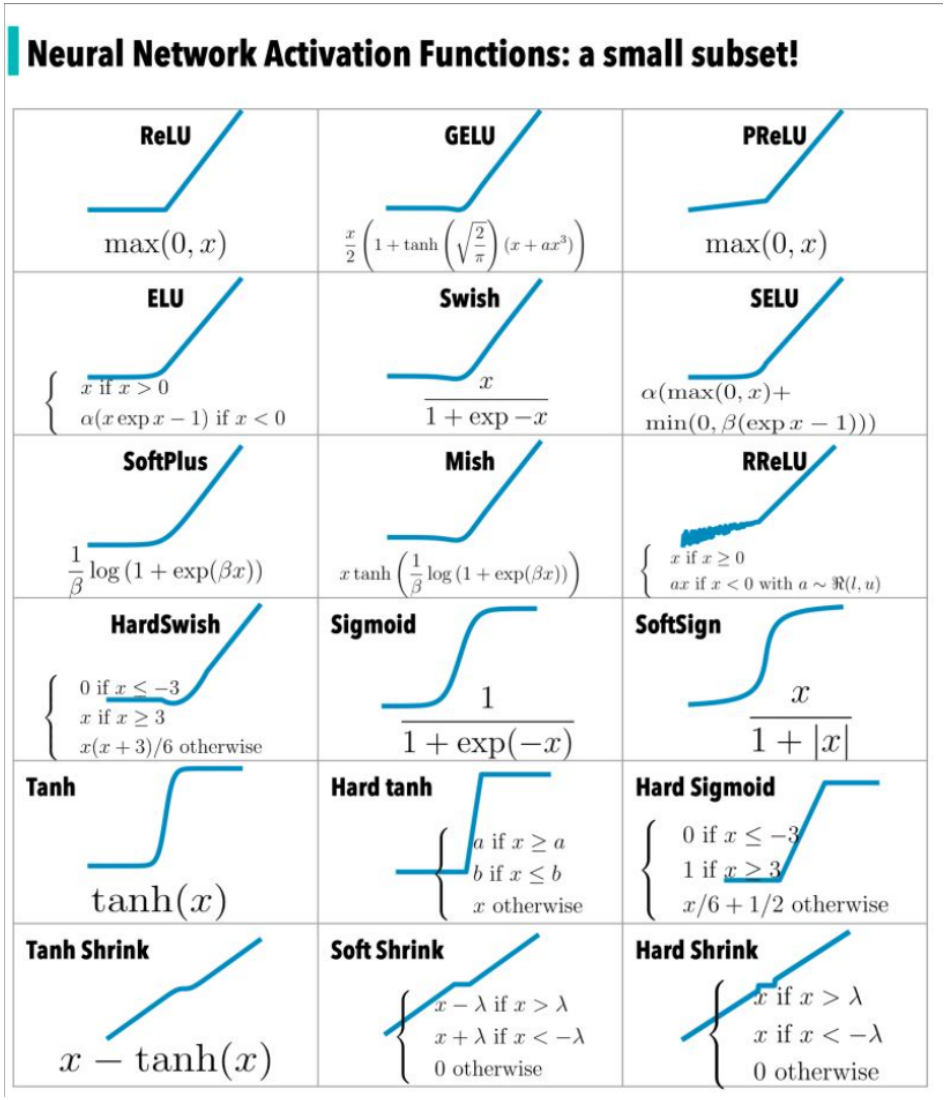




Estructura del Perceptrón



Funciones de activación



Entrenamiento del Perceptrón

- Inicialización: Se establecen los pesos y el umbral a cero o valores pequeños aleatorios.
- Para cada ejemplo de entrenamiento la neurona calcula su salida. Si ésta es incorrecta, se ajustan los pesos y el umbral según la regla del perceptrón.
- Los pesos y el umbral se actualizan usando la fórmula:

$$w_i = w_i + \Delta w_i,$$

donde Δw_i se calcula como:

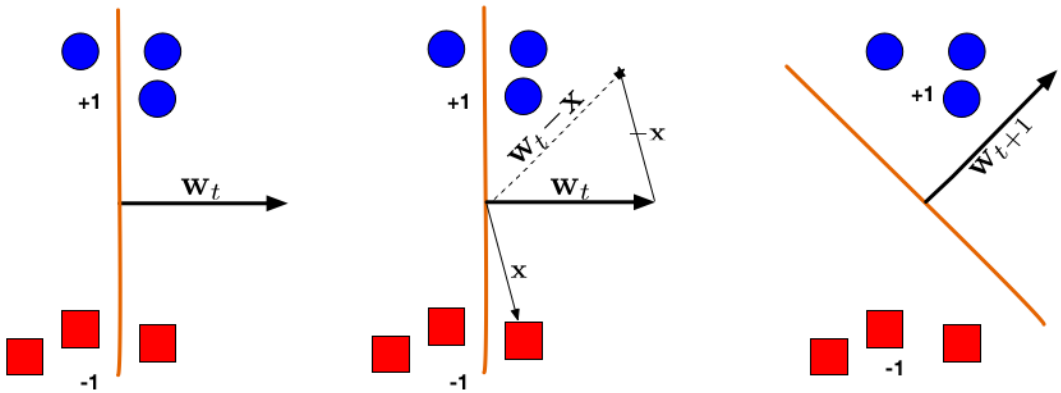
$$\Delta w_i = \eta(y - \hat{y})x_i,$$

con w_i denotando el peso i-ésimo, x_i la entrada i-ésima, y la salida deseada, $\hat{y}$ la salida de la neurona, y η la tasa de aprendizaje, un número real positivo pequeño.

- El proceso se repite hasta que el error sea bajo o se alcance un número de iteraciones preestablecido.

Teorema de Convergencia del Perceptrón

- Durante el entrenamiento, cada vez que el perceptrón hace una predicción incorrecta, ajusta sus pesos en una dirección que se acerca al hiperplano de separación correcto.
- Este ajuste reduce la distancia entre el vector de pesos del perceptrón y el vector de pesos que define el hiperplano de separación correcto.
- Debido a que el espacio de configuraciones es finito, el perceptrón eventualmente encontrará una configuración de pesos y sesgos que clasifica correctamente todos los ejemplos de entrenamiento.
- La convergencia se logra siempre y cuando exista un hiperplano que separe los conjuntos de datos de las salidas deseadas, es decir, si los datos son **linealmente separables**.



Función de Pérdida

Bajo otra perspectiva, el objetivo del entrenamiento del perceptrón es minimizar una función de pérdida.

$$L = \sum_{i=1}^{n} \text{máx}(0, -y_i \cdot f(x_i))$$

Donde x_i representa la observación, y_i la etiqueta esperada y $f(x_i)$ el valor predicho por el perceptrón.

- Esta función de pérdida penaliza incorrectas clasificaciones con un valor de 1 y clasificaciones correctas con un valor de 0.
- El algoritmo del perceptrón garantiza encontrar el mínimo de esta función de pérdida.
- Si el mínimo es cero, el perceptrón clasifica correctamente todas las entradas del entrenamiento.

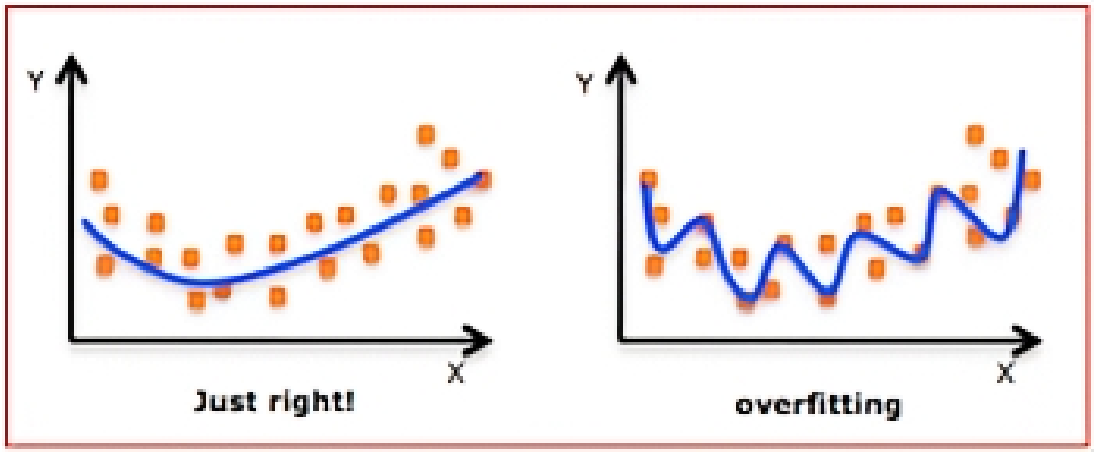
Regularización en el Perceptrón

El sobreajuste es un problema común en clasificadores como el perceptrón. Para mitigarlo, la **Regularización** agrega un término de penalización a la función de pérdida.
Regularización L_2 (Decaimiento del Peso):

$$L_{\text{reg}} = L + \lambda \|w\|^2$$

Donde $\|w\|^2 = w_1^2 + w_2^2 + \cdots + w_n^2$ es la suma de los cuadrados de los pesos, y λ es un parámetro de regularización que controla la importancia de la penalización de los pesos en relación con la pérdida original.

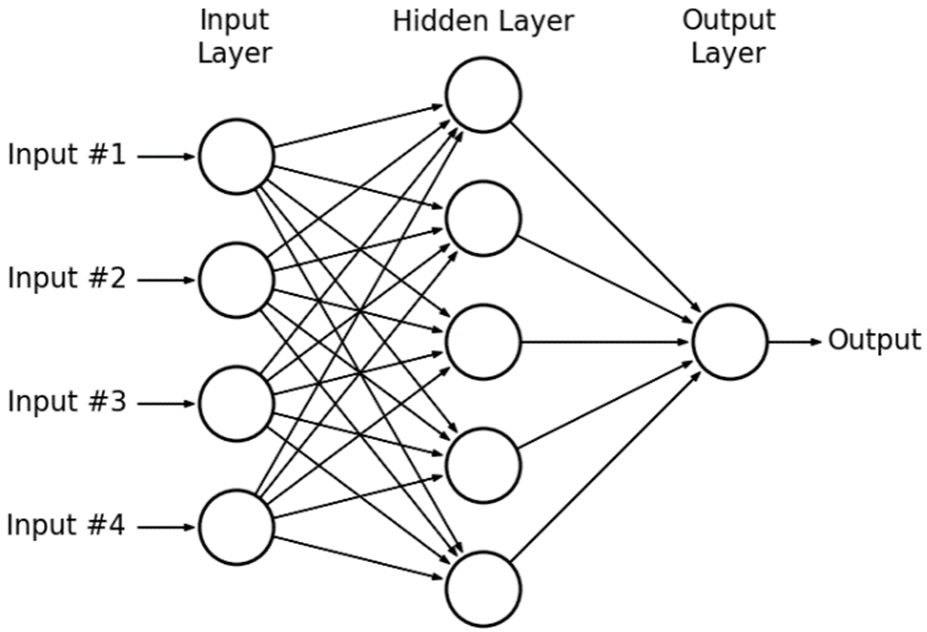
La regularización L_2 ayuda a mantener los pesos pequeños, lo que hace que el modelo sea más simple y menos propenso al sobreajuste.



El Perceptrón Multicapa (MLP)

El Perceptrón Multicapa (MLP) es una extensión del perceptrón simple que utiliza múltiples capas de neuronas.

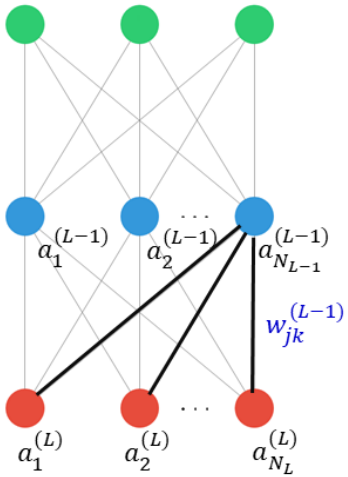
- Cada neurona en una capa está conectada a todas las neuronas de la capa siguiente, formando una estructura en red totalmente conectada.
- El MLP consta de al menos tres capas de nodos: una capa de entrada, una de salida y una o más capas ocultas.
- Introduce funciones de activación no lineales en las neuronas de las capas ocultas para aprender y modelar relaciones no lineales entre las entradas y las salidas.



Proceso de Aprendizaje del MLP: Retropropagación

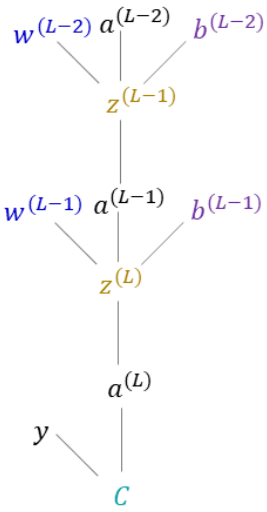
El proceso de aprendizaje del MLP se realiza a través de un algoritmo llamado **retropropagación**.

- **Paso hacia adelante**: La entrada se propaga hacia adelante hasta generar la salida y se compara con la etiqueta verdadera para calcular la pérdida.
- **Paso hacia atrás (retropropagación)**: El error de la salida se propaga hacia atrás hasta las capas ocultas. Se calculan las derivadas parciales del error con respecto a los pesos y sesgos.
- **Actualización de pesos**: Los pesos y sesgos se actualizan en dirección opuesta al gradiente para minimizar la función de pérdida. La tasa de aprendizaje controla el tamaño del paso.



$$\frac{\partial C}{\partial w_{jk}^{(L-1)}} = \frac{\partial z_j^{(L)}}{\partial w_{jk}^{(L-1)}} \frac{\partial a_j^{(L)}}{\partial z_j^{(L)}} \frac{\partial C}{\partial a_j^{(L)}}$$

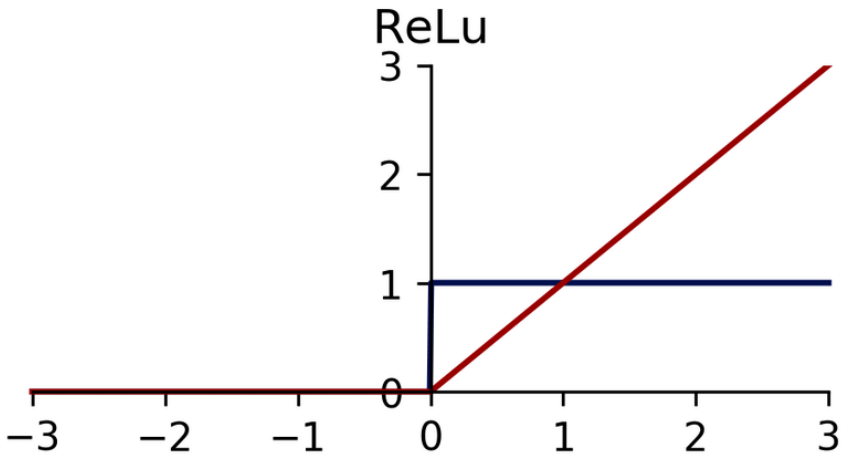
$$\frac{\partial C}{\partial a_k^{(L-1)}} = \sum_{j=1}^{N_L} \frac{\partial z_j^{(L)}}{\partial a_k^{(L-1)}} \frac{\partial a_j^{(L)}}{\partial z_j^{(L)}} \frac{\partial C}{\partial a_j^{(L)}}$$



Importancia de la Función de Activación ReLU

ReLU (Rectified Linear Unit)

- ReLU se define como $f(x) = \text{máx}(0, x)$, asignando valores negativos a 0 y manteniendo valores positivos.
- La simplicidad computacional de ReLU no afecta la velocidad de entrenamiento y añade la no linealidad necesaria al modelo.
- Durante el entrenamiento de redes neuronales profundas, las actualizaciones de los pesos pueden llegar a ser muy pequeñas, casi cercanas a cero. Esto se conoce como desvanecimiento del gradiente y puede hacer que la red deje de aprender. Debido a que la función ReLU tiene una derivada constante de 1 para todos los valores positivos, no sufre tanto de este problema.



Notas Finales sobre el Perceptrón

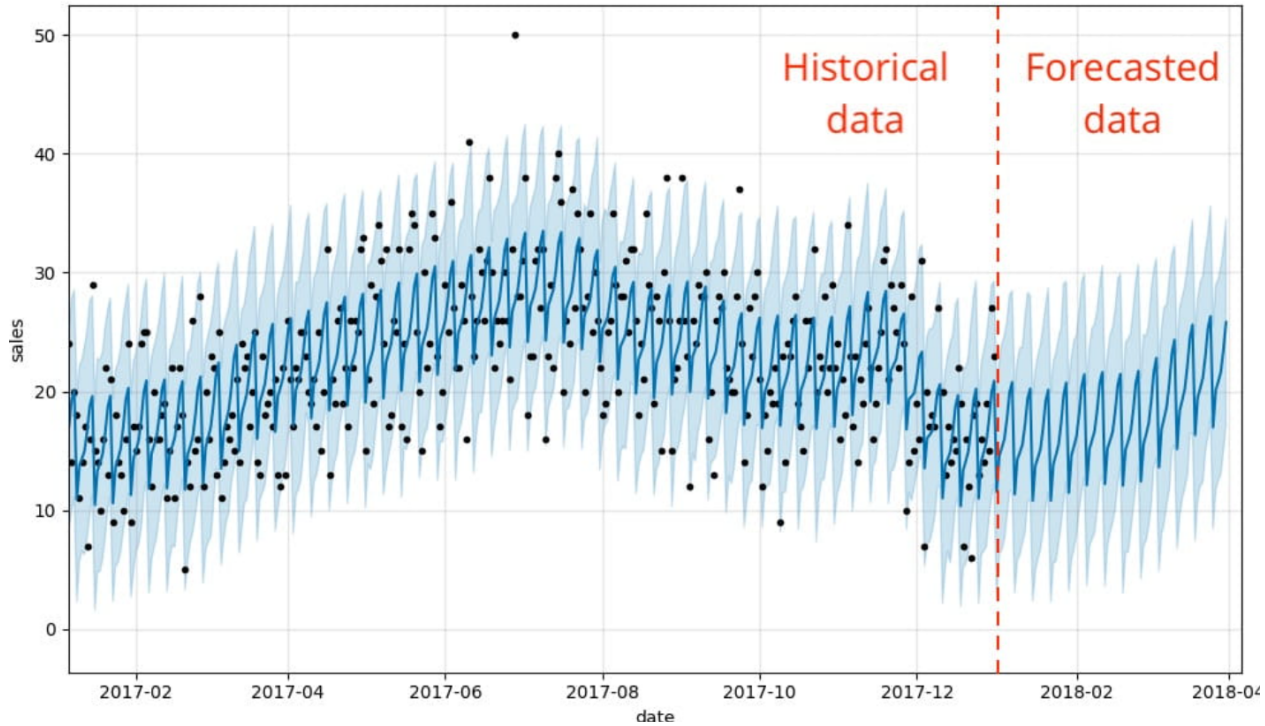
- El perceptrón es la arquitectura de red neuronal que detona el desarrollo de las redes neuronales modernas.
- Su adaptabilidad para extenderse hacia el perceptrón multicapa permite resolver problemas no linealmente separables.
- La introducción de funciones de pérdida y regularización lo llevó al campo de la optimización combinatoria. Esto ha permitido estudiarlo matemáticamente y ha dado lugar a arquitecturas más avanzadas como las redes recurrentes y convolucionales.
- Problemas como el sobreajuste y el desvanecimiento del gradiente se han abordado a partir de la estructura básica del perceptrón.

Tabla de Contenido

Redes neuronales recurrentes y LSTM

Una introducción

- Red neuronal diseñada para procesar datos secuenciales.
- Caracterizada por introducir estados ocultos que se calculan en términos de la entrada actual y recurrentemente de todas las entradas previas.
- Se pueden utilizar para una variedad de tareas, incluyendo predicción, clasificación y generación.



Ecuación para el estado oculto

La ecuación para el estado oculto se expresa como:

$$h_t = \sigma(W_{ih}x_t + W_{hh}h_{t-1} + b_h) \tag{1}$$

donde:

- h_t representa el estado oculto en el paso de tiempo t.
- x_t es la entrada en el paso de tiempo t.
- W_{ih} sirve para conectar la entrada con el primer estado oculto.
- W_{hh} contiene los pesos para conectar el estado oculto anterior con el estado oculto actual.
- b_h es el vector de sesgo.
- σ es una función de activación no lineal, como la sigmoide o la tangente hiperbólica.

Ecuación para la salida

Y la ecuación de la salida de la red a partir del estado oculto es:

$$y_t = \sigma(W_{hy} h_t + b_y) \tag{2}$$

donde:

- y_t representa la salida conforme transcurre el tiempo t.
- W_{hy} conecta el estado oculto con la salida.
- b_y se aplica directamente a la salida.

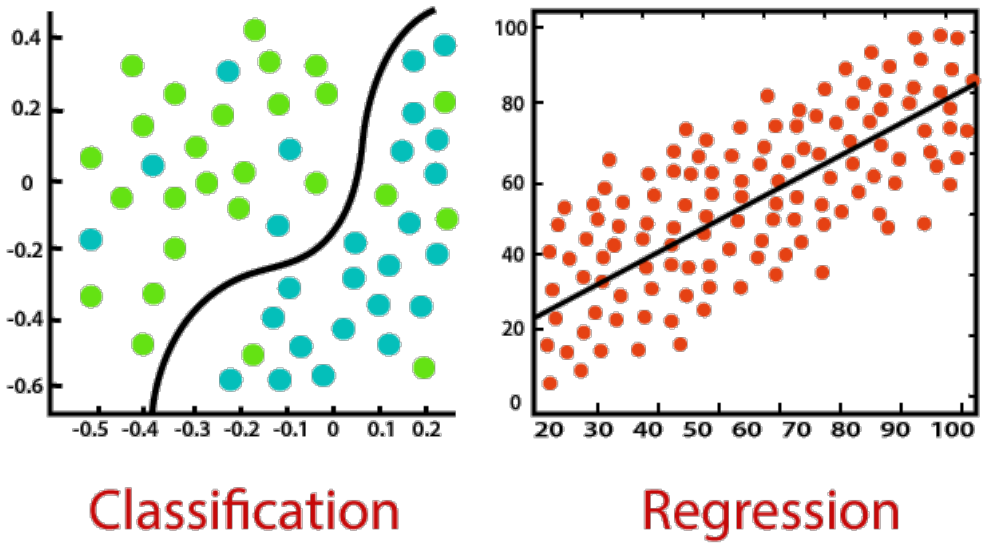
Las capas se distribuyen en la red, y en su conjunto forman una memoria capaz de recordar el pasado que ha analizado la red

Funciones Objetivo en RNN

En RNN, se utilizan funciones objetivo para guiar el entrenamiento y ajustar los parámetros de la red. En cada caso es común usar las siguientes funciones:

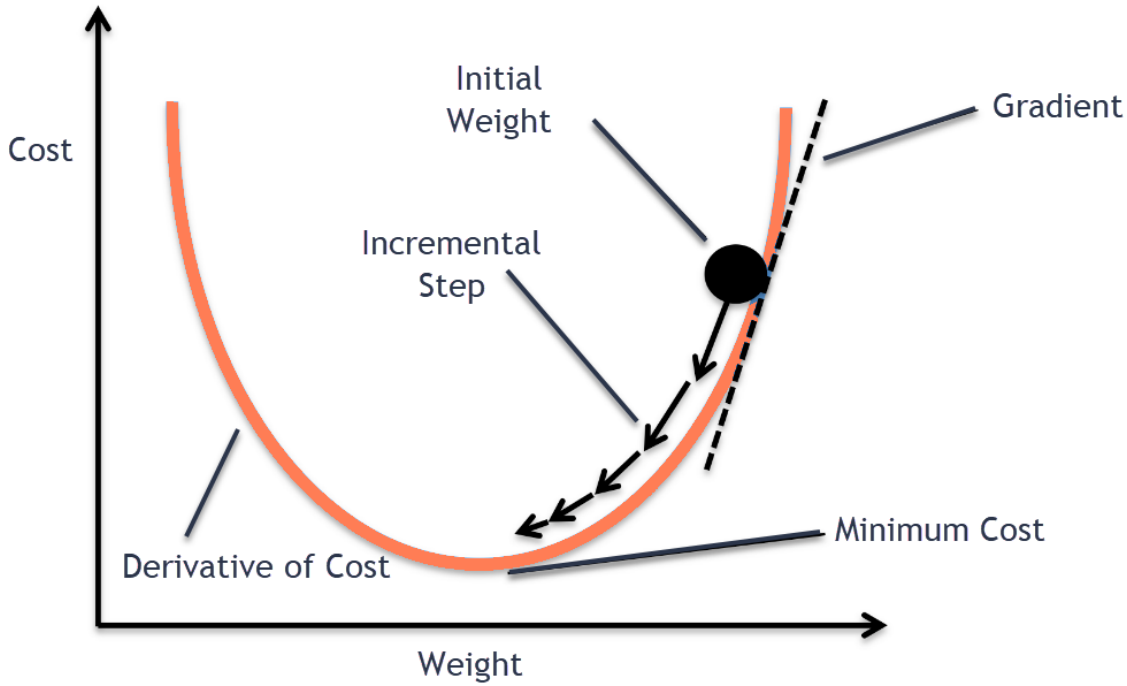
- **Regresión:** Error Cuadrático Medio (MSE). Se calcula la diferencia al cuadrado entre las salidas predichas y las salidas esperadas y se promedia sobre todos los ejemplos de entrenamiento esperadas.
- **Clasificación:** Entropía Cruzada (cross-entropy). Esta función mide la discrepancia entre las distribuciones de probabilidad de las salidas predichas y las salidas esperadas.
- **Generación de secuencias:** La función objetivo puede ser una medida de similitud entre la secuencia generada y la secuencia objetivo, como la distancia de Levenshtein o la pérdida de la perplejidad.

Tipos de tareas



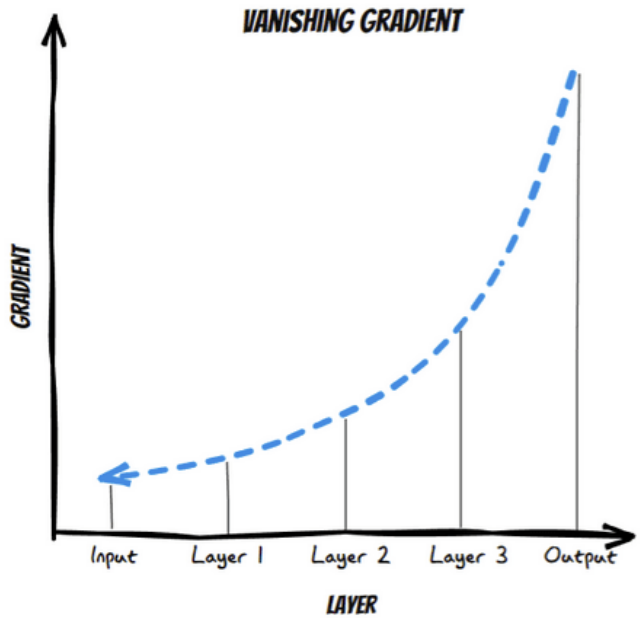
Métodos de Entrenamiento en RNN

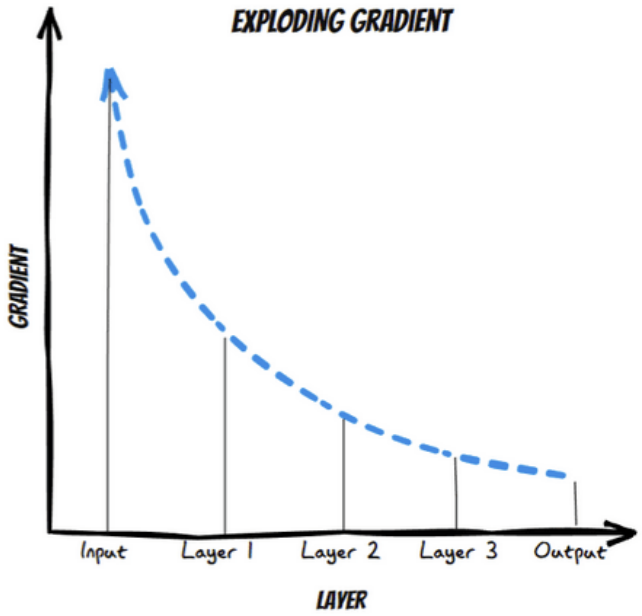
El método de gradiente estocástico (SGD) es común para entrenar RNN. Actualiza los pesos usando el gradiente de la función de pérdida en muestras individuales o lotes pequeños.
SGD ajusta gradualmente la red a los datos, optimizando parámetros para minimizar la pérdida.



Redes LSTM (Long Short-Term Memory)

- Las redes LSTM (Long Short-Term Memory) son una variante de las RNN diseñadas para superar las limitaciones de las RNN tradicionales:
  - Desvanecimiento o explosión del gradiente.
  - Dificultad para capturar dependencias a largo plazo.
- El propósito de las redes LSTM es mejorar el aprendizaje de dependencias a largo plazo en secuencias.





Compuertas en las LSTM

Las redes LSTM utilizan células de memoria y compuertas para regular el flujo de información.
Cada célula de memoria en una LSTM tiene tres compuertas principales:

- Compuerta de entrada: Actualiza la célula de memoria con la entrada actual y el estado anterior.
- Compuerta de olvido: Permite a la LSTM olvidar información irrelevante o no deseada.
- Compuerta de salida: Controla la información que se envía como salida de la célula de memoria.

Esta arquitectura permite que la red aprenda a mantener y actualizar información a largo plazo y controlar el flujo de información en función del contexto y la relevancia.

Notas Finales sobre Redes Recurrentes y Redes LSTM

- Las redes neuronales recurrentes son una herramienta poderosa para trabajar con datos secuenciales y capturar dependencias a largo plazo.
- Las redes LSTM son una variante diseñada para abordar el problema del desvanecimiento/explosión del gradiente. Son especialmente efectivas en tareas que involucran secuencias largas y dependencias a largo plazo, como el procesamiento del lenguaje natural, la traducción automática y la generación de texto.

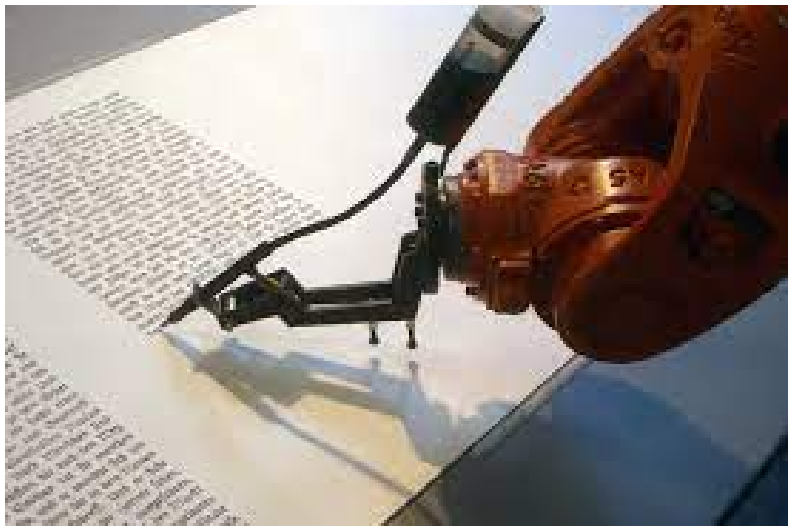


Tabla de Contenido

Reducción de la Dimensión - Definición e Importancia

Definición: La reducción de la dimensión es una técnica que busca disminuir la cantidad de características aleatorias en los datos.

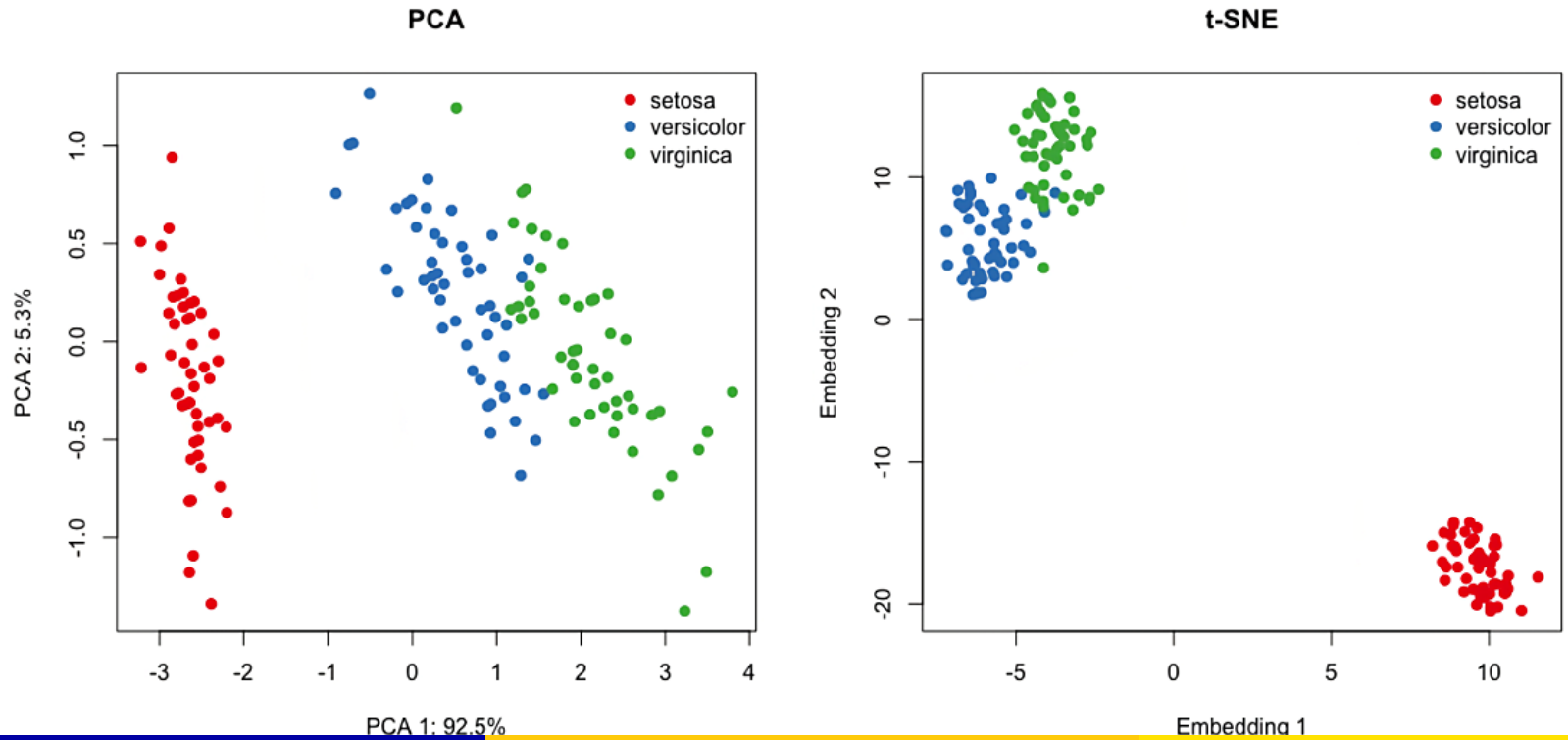
Importancia:

- Visualización: Permite proyectar conjuntos de datos de alta dimensión a 2D o 3D para facilitar la visualización.
- Eficiencia Computacional: Reduce el tiempo de cálculo y la memoria necesaria en algoritmos de aprendizaje automático.
- Mitigación de la Maldición de la Dimensionalidad: Ayuda a evitar problemas causados por la alta dimensionalidad de los datos.
- Resolución de Multicolinealidad: Identifica nuevas características independientes en conjuntos de datos con correlaciones.

Enfoques y Ejemplo Clásico

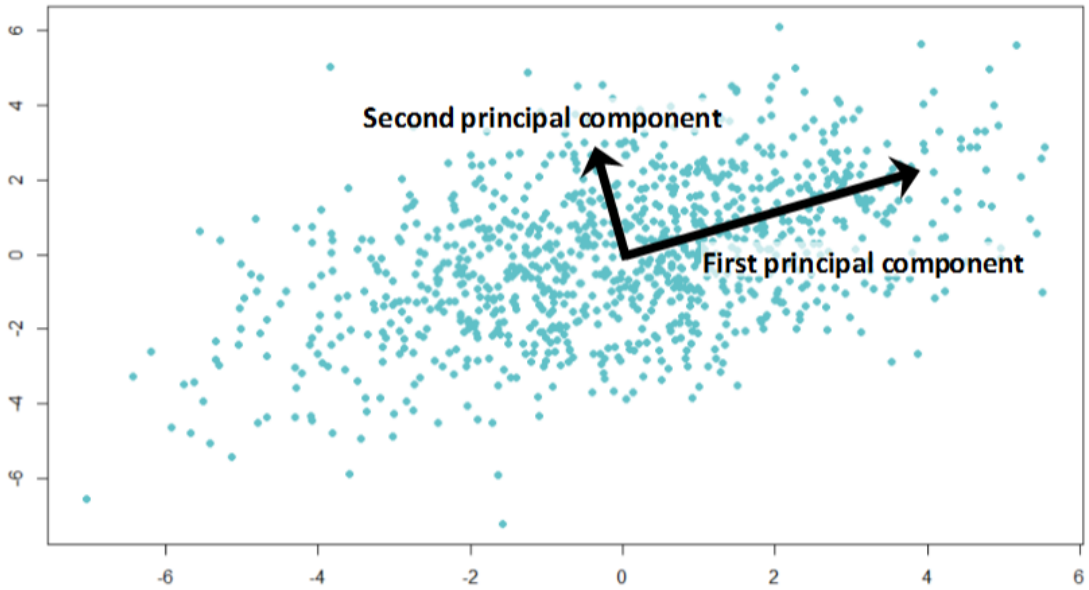
- **Lineal:** Métodos como el Análisis de Componentes Principales (PCA) proyectan datos a un subespacio de menor dimensión preservando la mayor varianza.
- **No Lineal:** Métodos como el Encaje de Vecinos más Cercanos Localmente Lineales (LLE) y t-SNE permiten proyecciones más complejas para preservar relaciones no lineales.

Ejemplo Clásico: Iris - Un conjunto de datos con 4 características (longitud y ancho de sépalo y pétalo) proyectado a 2D.



Análisis de Componentes Principales (PCA)

- PCA es un método clásico de reducción de la dimensión.
- Transforma un conjunto de observaciones de variables posiblemente correlacionadas en un conjunto de valores de variables no correlacionadas linealmente llamadas componentes principales.



Cálculo de Componentes Principales

El cálculo de los componentes principales busca encontrar el vector $\mathbf{w}$ que maximiza la varianza de los datos proyectados en su dirección.

$$V(\mathbf{w}) = \mathbf{w}^T \mathbf{S} \mathbf{w} \tag{3}$$

Donde $\mathbf{S}$ es la matriz de covarianza de los datos.
Para esto, se forma la función Lagrangiana:

$$\mathcal{L}(\mathbf{w}, \lambda) = \mathbf{w}^T \mathbf{S} \mathbf{w} - \lambda(\mathbf{w}^T \mathbf{w} - 1) \tag{4}$$

Donde λ es el multiplicador de Lagrange.
Luego, se toma la derivada con respecto a $\mathbf{w}$ e iguala a cero:

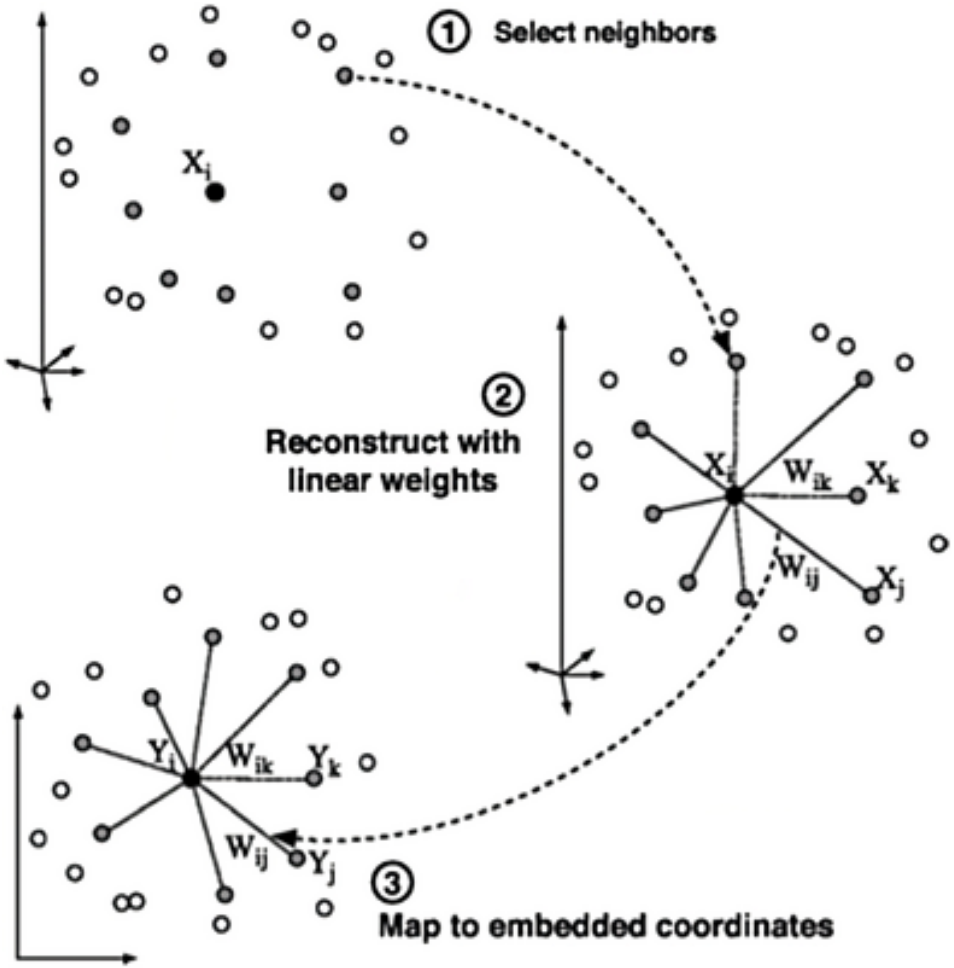
$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}} = 2\mathbf{S}\mathbf{w} - 2\lambda\mathbf{w} = 0 \tag{5}$$

Resolviendo la ecuación, se encuentra que $\mathbf{S}\mathbf{w} = \lambda\mathbf{w}$, es decir, $\mathbf{w}$ es un vector propio de $\mathbf{S}$ y λ es el valor propio correspondiente.
El primer componente principal es el vector propio correspondiente al mayor valor propio de $\mathbf{S}$.

Encaje de Vecinos más Cercanos Localmente Lineales (LLE)

LLE es un método no lineal de reducción de la dimensión que se basa en la estructura geométrica local de los datos de alta dimensión.

- El algoritmo LLE consta de dos pasos principales:
  - Reconstrucción local: Se identifican los k vecinos más cercanos para cada punto de datos y se reconstruye como una combinación lineal de ellos.
  - Encaje global: Utiliza la matriz de pesos obtenida en el paso anterior para mapear los datos a una dimensión menor.



Paso de Reconstrucción Local

El objetivo es representar cada punto de datos de alta dimensión x_i como una combinación lineal de sus k vecinos más cercanos.

$$\min_{w_{i1},\ldots,w_{ik}} ||x_i - \sum_{j=1}^{k} w_{ij} x_{nj}||^2 \tag{6}$$

- x_i es el punto de datos que estamos intentando reconstruir.
- x_{nj} son los vecinos más cercanos de x_i.
- w_{ij} son los pesos que estamos intentando aprender, es decir, cuánto de cada vecino deberíamos "mezclar" para obtener una buena aproximación de x_i.
- Se impone la restricción: $\sum_{j=1}^{k} w_{ij} = 1$.

Para resolver este problema de optimización, se puede utilizar cualquier algoritmo estándar, como el descenso de gradiente o un enfoque de mínimos cuadrados.

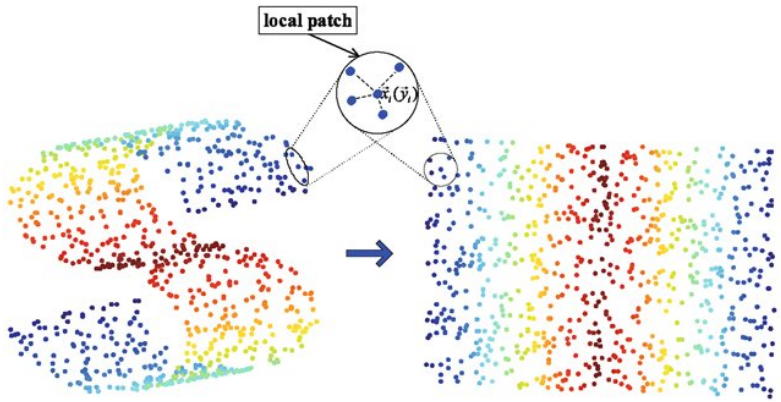
Paso de Encaje Global

Objetivo: Encontrar una representación de baja dimensión que minimice la discrepancia Φ entre las representaciones de alta (X) y baja (Y) dimensión.

$$\Phi(Y) = \sum_i ||y_i - \sum_j w_{ij} y_j||^2 \tag{7}$$

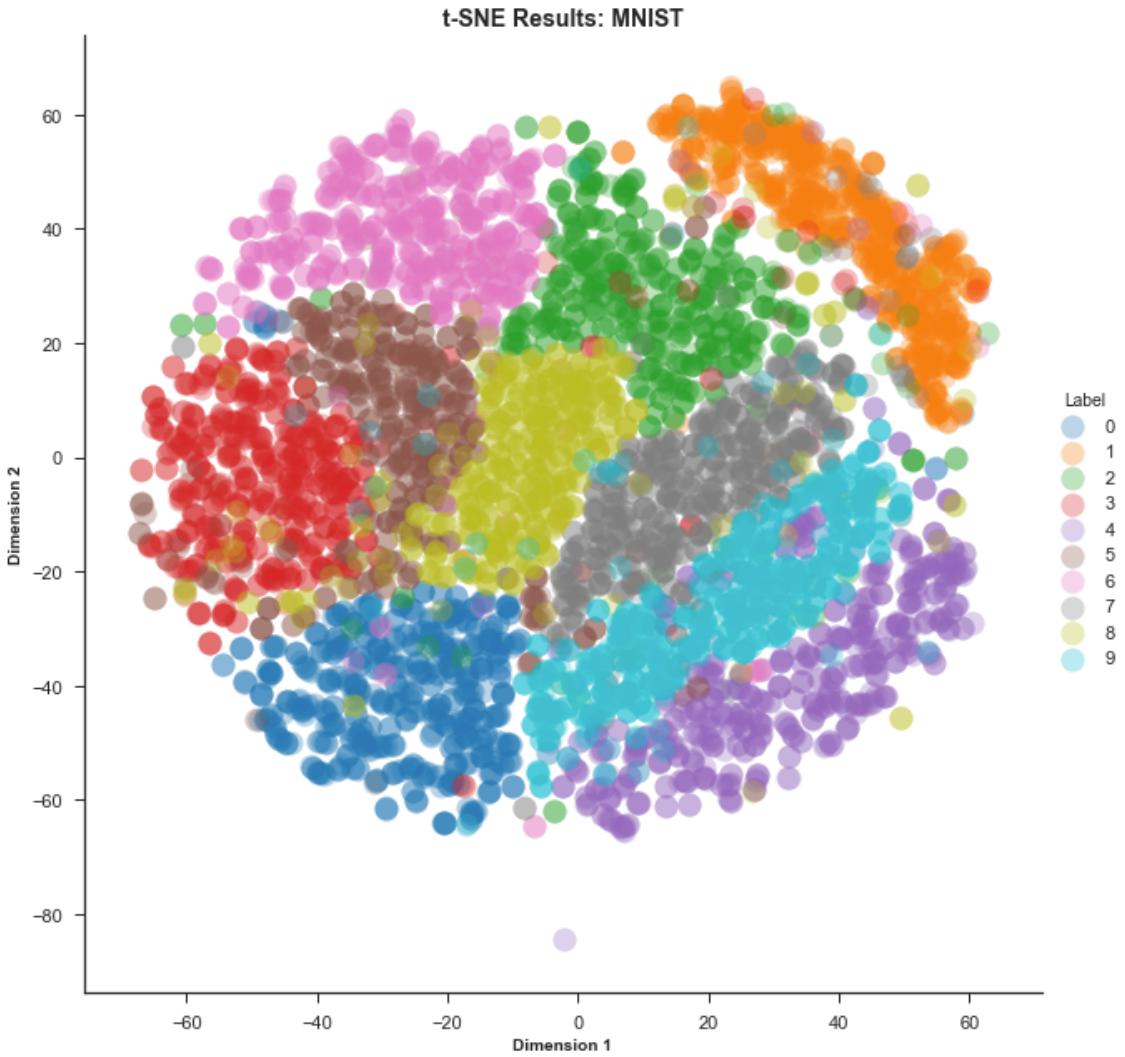
- y_i son los puntos de datos en la representación de baja dimensión.
- w_{ij} son los pesos de reconstrucción obtenidos en el paso de reconstrucción local.

La solución al problema de optimización del encaje global generalmente implica la solución de un problema de autovalores. Los puntos de datos en la representación de baja dimensión son los vectores propios correspondientes a los autovalores más pequeños de la matriz de pesos de reconstrucción.
El resultado final es una representación de baja dimensión que preserva la estructura geométrica local capturada por los pesos de reconstrucción.



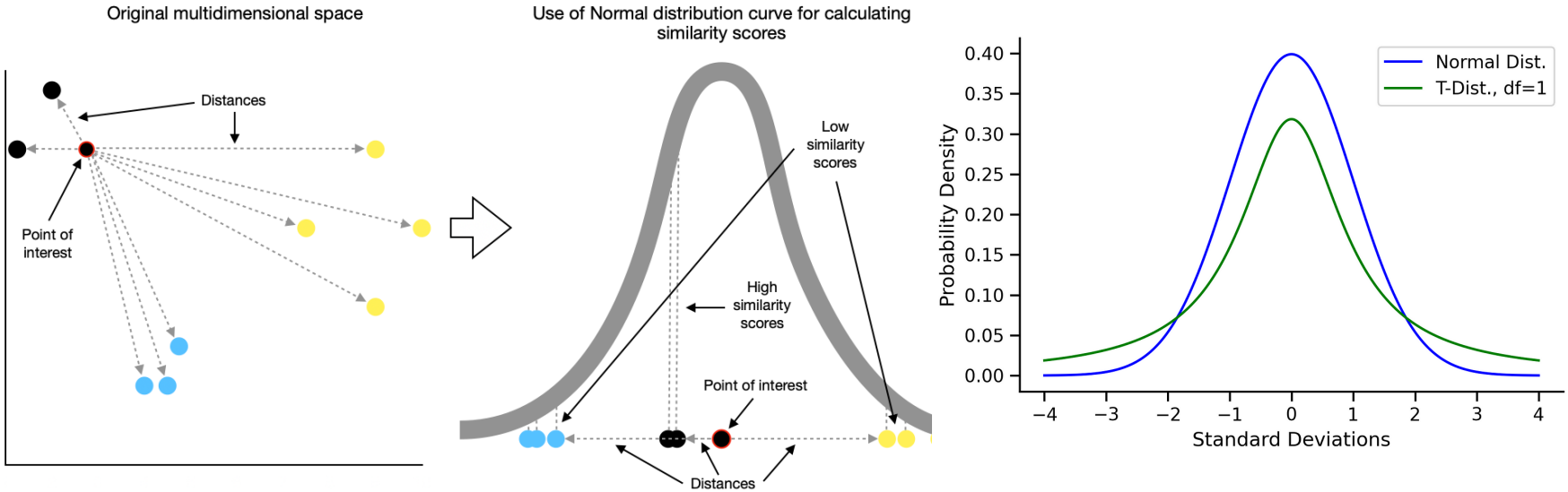
Encaje del Vecino Estocástico (t-SNE)

- Método ampliamente utilizado para la visualización de datos de alta dimensión.
- Desarrollado por van der Maaten y Hinton en 2008, basado en técnicas de reducción de dimensionalidad estocástica.
- Objetivo: Preservar la estructura de vecindad de los datos de alta dimensión en un espacio de baja dimensión.



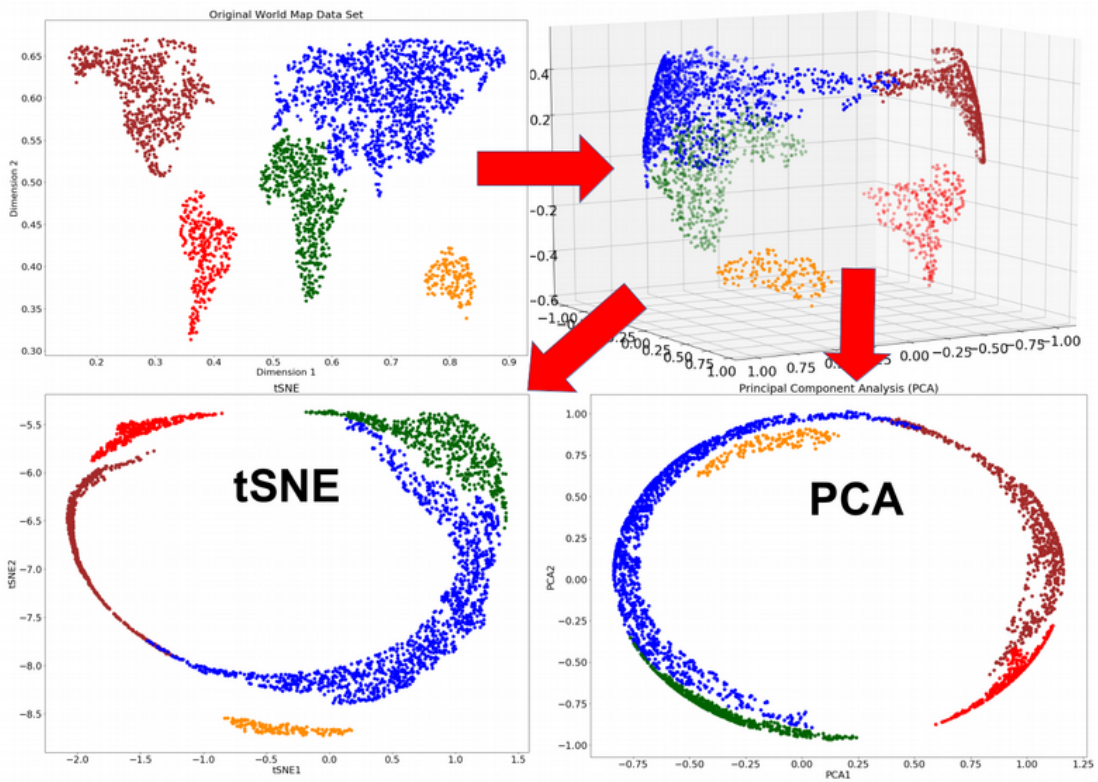
Pasos del Algoritmo:

- Calcula las similitudes entre las observaciones en el espacio de alta dimensión como probabilidades condicionales.
- Calcula las similitudes entre las observaciones en el espacio de baja dimensión utilizando una distribución de t-Student con un grado de libertad.
- Minimiza la discrepancia entre las probabilidades de alta y baja dimensión utilizando la divergencia de Kullback-Leibler.



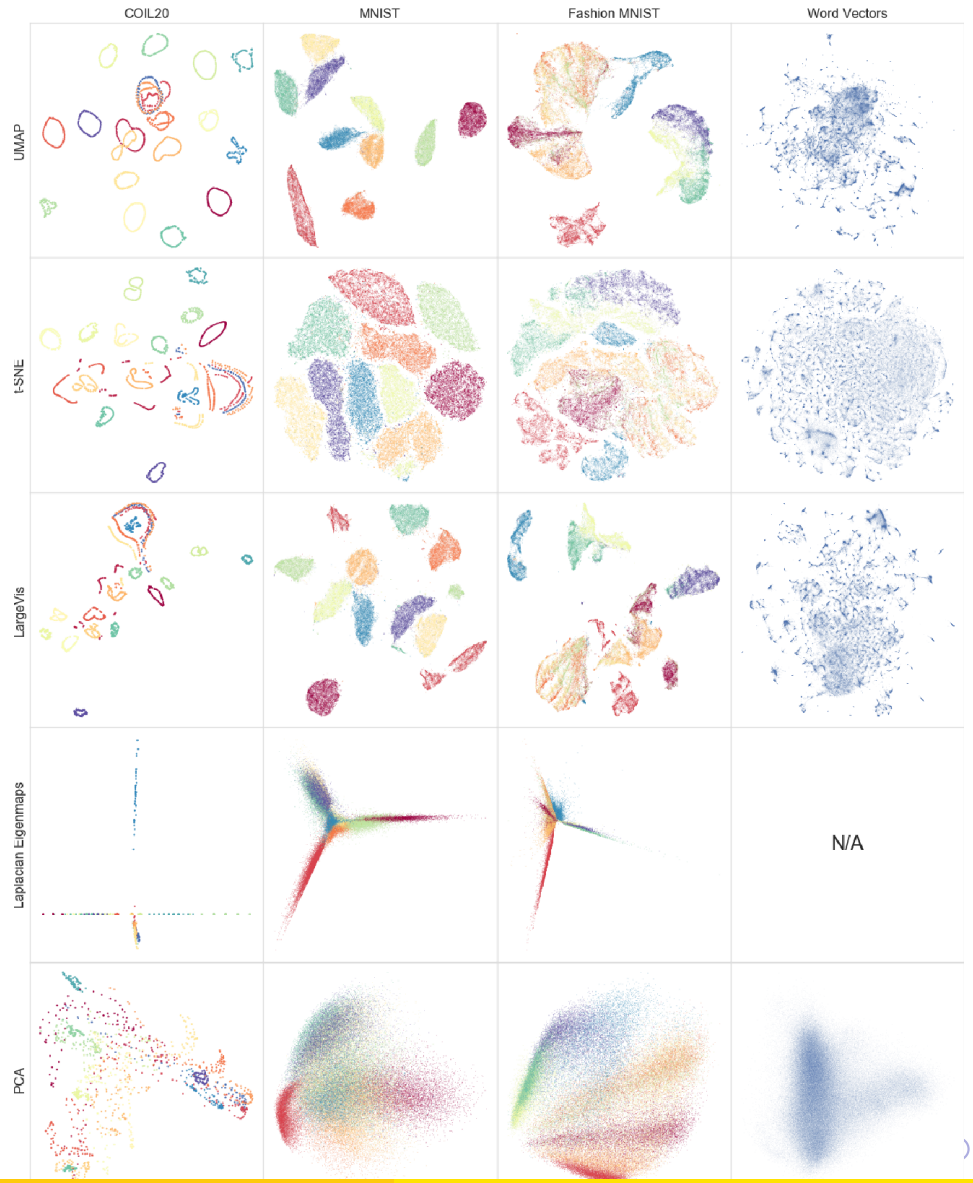
Limitaciones del t-SNE:

- Optimización no convexa: Puede obtener resultados diferentes en diferentes ejecuciones.
- Puede no preservar distancias globales entre observaciones, aunque preserva relaciones locales.
- Interpretación de distancias en el espacio de baja dimensión puede ser difícil, ya que t-SNE se enfoca en preservar relaciones de vecindad, no distancias absolutas.



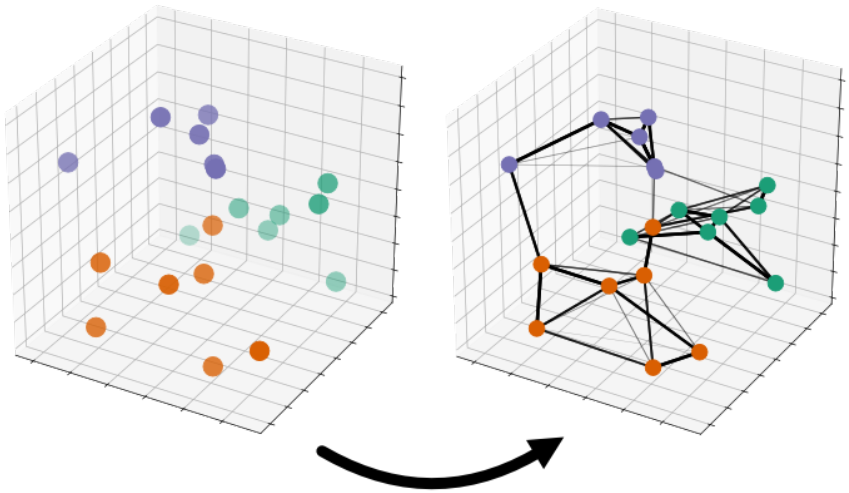
UMAP - Uniform Manifold Approximation and Projection

- Técnica novedosa para reducción de dimensiones y visualización de datos de alta dimensión.
- Basada en fundamentos matemáticos sólidos de topología y geometría.
- Captura tanto la estructura local como global de los datos complejos en altas dimensiones.
- Eficiente computacionalmente y compatible con scikit-learn.



Funcionamiento básico de UMAP

- **Construcción del Grafo de Vecinos:** UMAP crea un gráfico de vecinos más cercanos en el espacio de alta dimensión. Calcula las distancias a los vecinos más cercanos para cada punto, capturando la estructura local de los datos.
- **Optimización Basada en el Grafo:** UMAP busca optimizar el grafo en el espacio de baja dimensión, preservando la estructura local y global de los datos. Minimiza una función de costo que mide la diferencia entre las distancias en los grafos de alta y baja dimensión, logrando que las distancias en el espacio de baja dimensión reflejen fielmente las distancias en el espacio original.



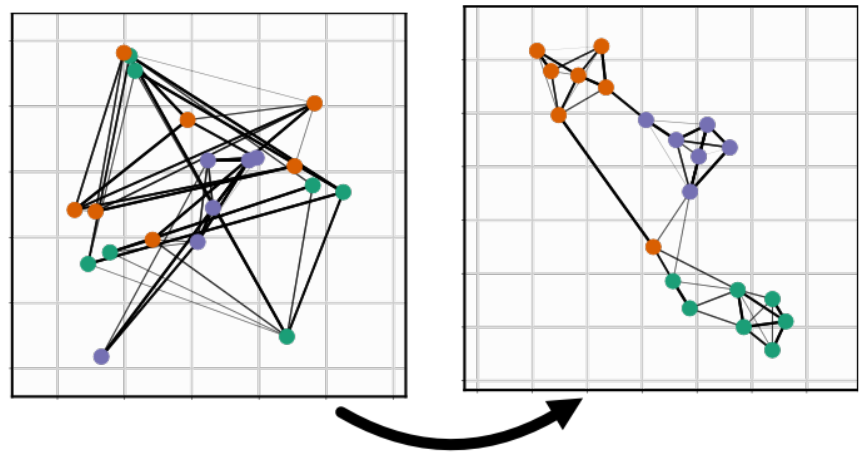
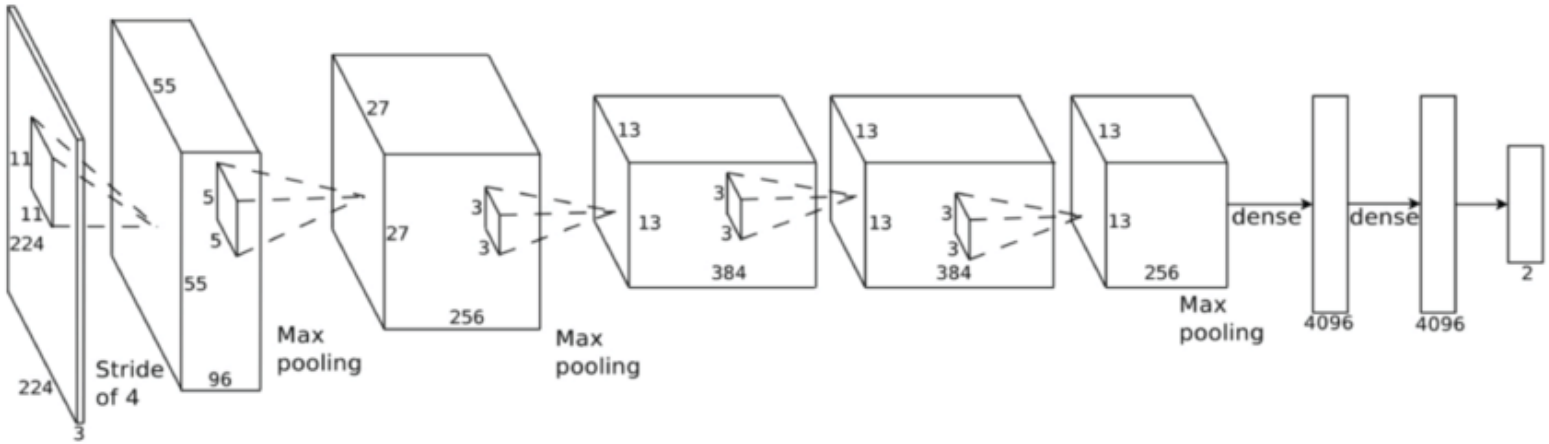


Tabla de Contenido

Redes Neuronales Convolucionales (CNN)

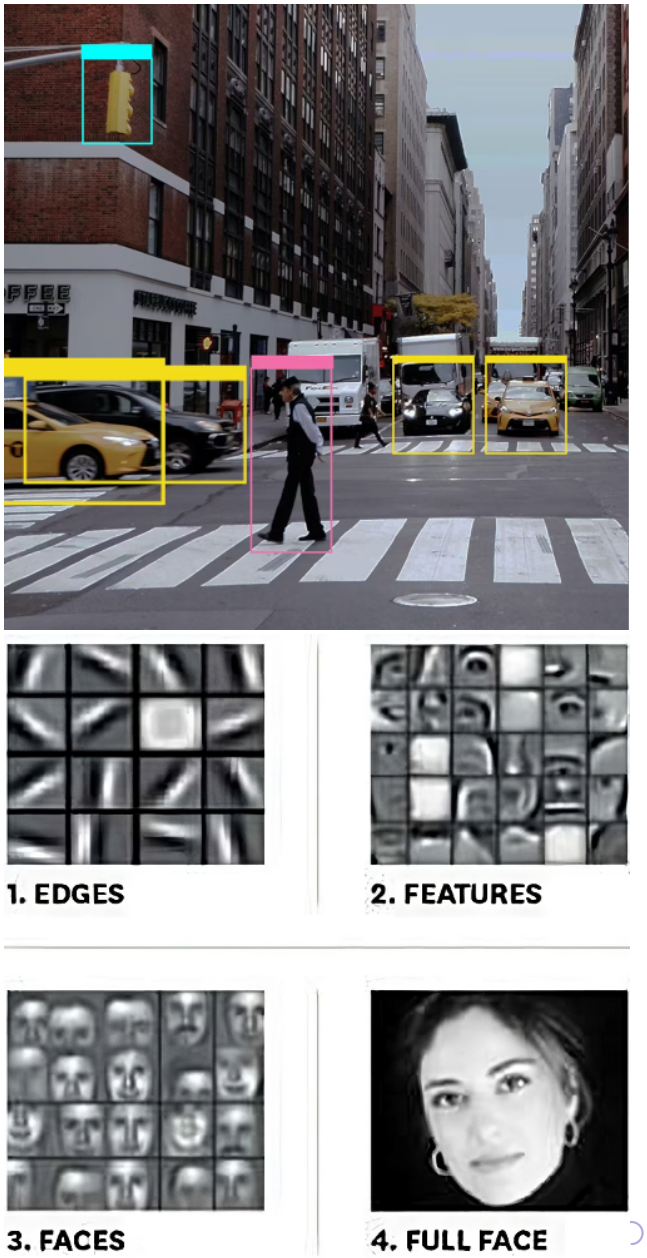
- Las Redes Neuronales Convolucionales (CNN) han revolucionado la visión por computadora con un rendimiento de vanguardia en diversas tareas, como clasificación de imágenes, detección de objetos y segmentación de imágenes.
- Inspiradas en la percepción visual de los seres vivos, las CNN se basan en la idea de que las células en la corteza visual reconocen lo que miran partiendo de pequeños campos receptivos que barren el entorno.
- En 2012, AlexNet (mostrada abajo) marcó un hito en el campo de visión por computadora con su impactante eficiencia y resultados sobresalientes.



Aplicaciones y Tipos de CNN

- CNN bidimensionales: Ampliamente utilizadas en el reconocimiento de patrones en imágenes y en espacios de más de dos dimensiones en procesamiento del lenguaje natural y visión por computadora.

- CNN lineales: Especiales para analizar series de tiempo en áreas como climatología, control de procesos industriales, economía y otras áreas relacionadas con la Teoría de Procesos Estocásticos.

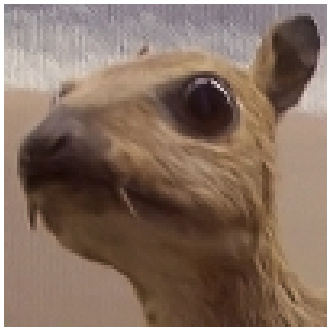


Redes Neuronales Convolucionales bidimensionales

Las CNN son arquitecturas poderosas diseñadas para procesar datos estructurados en matrices o tensores, como imágenes y videos. Características clave:

- Aprendizaje Implícito de Características: Utilizan la convolución para detectar patrones relevantes en las imágenes, como bordes, esquinas y texturas.
- Reducción de Dimensiones: Las capas de agrupación reducen las dimensiones espaciales y hacen que las características sean más invariantes a las traslaciones y distorsiones.

Input image



Convolution Kernel

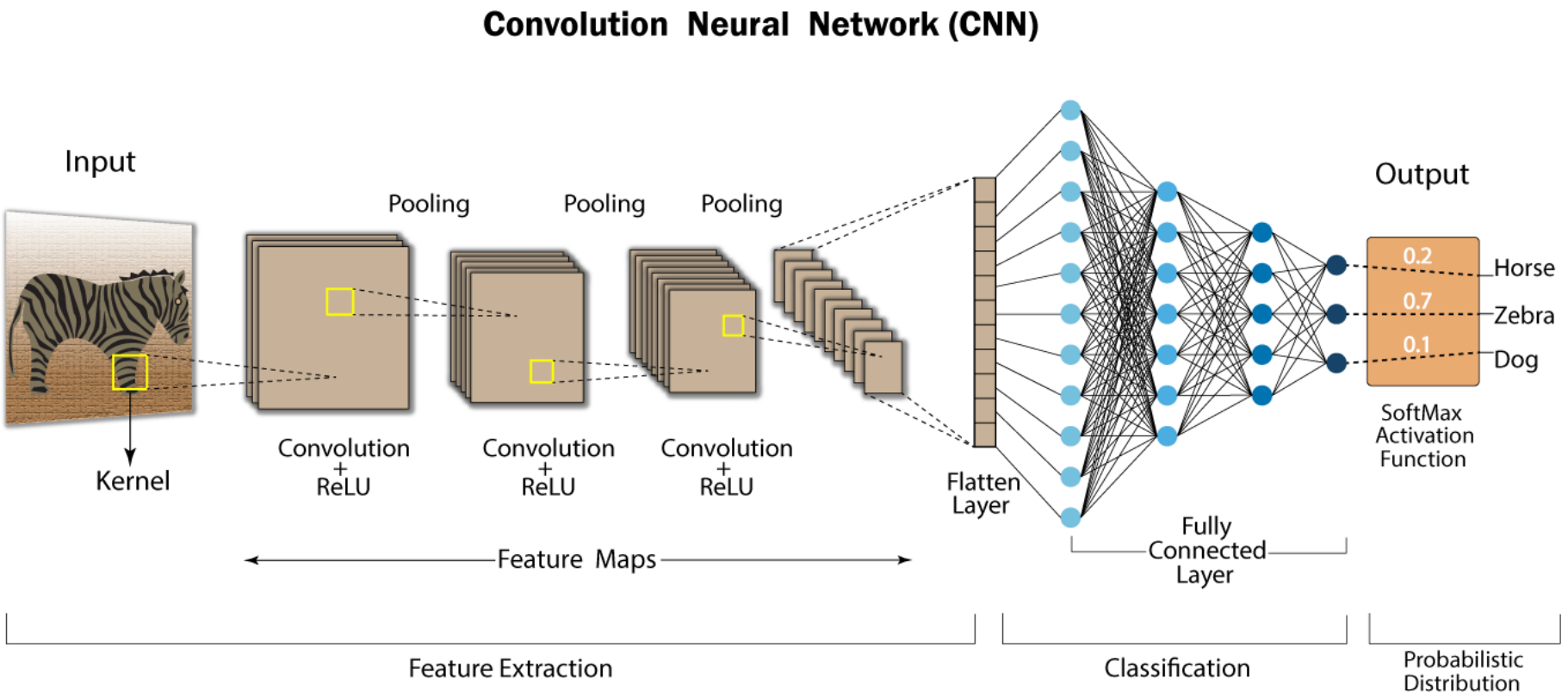
$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Feature map

Arquitectura de una CNN

- Capas Convolucionales: La salida de cada capa convolucional es un conjunto de mapas de características, donde cada mapa corresponde a un filtro. Luego se pasan a través de funciones de activación no lineales.
- Capas de Agrupación (Pooling): Reducen las dimensiones espaciales de los mapas de características, preservando información relevante. Ayudan a que las características aprendidas sean invariantes a traslaciones y distorsiones espaciales.
- Capas Totalmente Conectadas: Transforman los mapas de características finales en un vector para realizar predicciones de alto nivel, como probabilidades de clase en tareas de clasificación de imágenes.

Ejemplo de CNN



Función de Costo y Optimización en CNN

- Problema de Optimización en CNN: Entrenar los pesos para minimizar una función de costo.
- Función de Costo: En clasificación de imágenes, se utiliza comúnmente la función de costo de entropía cruzada (crossentropy):

$$J(\theta) = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} y_{i,c}\log(\hat{y}_{i,c})$$

  donde θ representa los pesos de la red, $y_{i,c}$ es la etiqueta verdadera para la muestra i en la clase c, y $\hat{y}_{i,c}$ es la salida predicha por la red para la muestra i en la clase c.
- Algoritmo de Optimización: Descenso de Gradiente Estocástico (SGD) con retropropagación:

$$\theta \leftarrow \theta - \alpha\frac{\partial J(\theta)}{\partial\theta}$$

  donde α es la tasa de aprendizaje, que controla el tamaño de los pasos de actualización de los pesos.

Ventajas y Desventajas de las CNN Lineales

- Ventajas:
  - Estructura simplificada y mayor interpretabilidad.
  - Menor cantidad de parámetros y mayor estabilidad numérica.
  - Mayor facilidad de implementación y menor demanda de cómputo.

- Desventajas:
  - Limitaciones en la capacidad de representación de datos complejos.

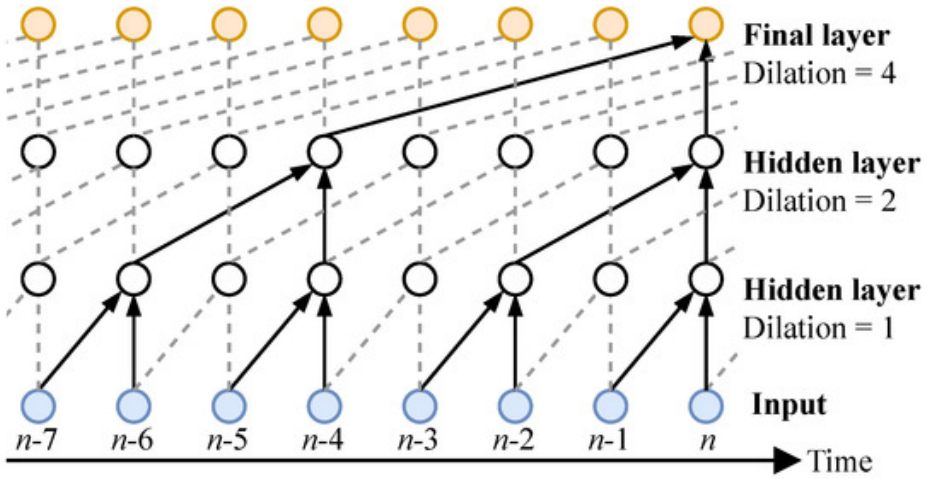
Las CNN lineales evitan la aplicación de funciones de activación no lineales que pueden introducir problemas de estabilidad numérica, como el desvanecimiento o explosión del gradiente.

Convolución Dilatada Causal (DCC)

La **Convolución Dilatada Causal** (DCC) es una técnica utilizada en las **CNN lineales** para capturar relaciones de largo alcance en una secuencia de entrada sin perder la propiedad causal de la convolución.

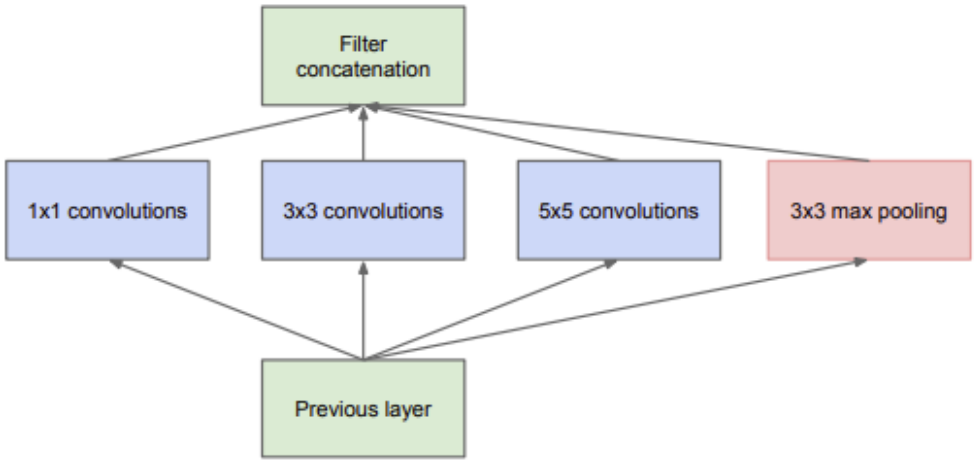
- La DCC expande el campo receptivo del filtro mediante **dilataciones** (espacios vacíos), sin aumentar su tamaño.
- Realiza operaciones de convolución en saltos regulares, abarcando un campo receptivo más amplio.

La DCC es especialmente útil para capturar relaciones no locales en una secuencia sin perder la eficiencia y la interpretación lineal de las CNN lineales.



Instauración: Crecimiento en Amplitud

En las CNN, la operación de convolución apila múltiples capas para obtener características precisas. La **DCC** reduce capas para el mismo campo receptivo
Cuando la longitud de la serie de tiempo crece, la red se vuelve inevitablemente más profunda, lo que puede conducir a sobreajuste.
Para resolver este problema, se propone el concepto de **"Instauración"** (Inception), donde se reemplaza cada núcleo de convolución por varios núcleos pequeños que amplían el campo receptivo sin volverse demasiado profundos.



(a) Inception module, naïve version

Tabla de Contenido

Referencias I


9. Recurrent Neural Networks — Dive into Deep Learning 1.0.0-beta0 documentation (2023). URL: https://d2l.ai/chapter_recurrent-neural-networks/index.html (visitado 29-05-2023).


A Beginner's Guide to Word2Vec and Neural Word Embeddings — Pathmind (2023). URL: https://wiki.pathmind.com/word2vec (visitado 03-06-2023).


Abdi, Hervé y Lynne J Williams (2010). "Principal component analysis". En: *Wiley Interdisciplinary Reviews: Computational Statistics* 2.4, págs. 433-459.


Ahuja, Ravindra K, Thomas L Magnanti y James B Orlin (1993). *Network flows: theory, algorithms, and applications*. Prentice hall Englewood Cliffs.


ALPAC (1966). *Language and Machines: Computers in Translation and Linguistics*. 1416. Washington DC: National Academic of Sciences / National Research Council. URL: https://web.archive.org/web/20110409070141/http://www.mt-archive.info/ALPAC-1966.pdf (visitado 09-07-2023).


An Interactive Node-Link Visualization of Convolutional Neural Networks (2023). URL: https://adamharley.com/nn_vis/ (visitado 24-05-2023).

Referencias II

Angluin, Dana (1976). "Computational complexity of learning with queries". En: *Yale University, Department of Computer Science* 111.5, pág. 22.

Applegate, David y col. (2006). *The Traveling Salesman Problem: A Computational Study*. Princeton University Press.

Bachmann, Samuel, Rene Schuldt y Stefan Lerch (2020). "Linear convolutional neural networks for weather prediction". En: *arXiv preprint arXiv:2003.14177*.

Bahdanau, Dzmitry y col. (2016). "End-to-end attention-based large vocabulary speech recognition". En: *arXiv preprint arXiv:1609.06773*.

Balafoutis, Chrysoula, Andreas Matzarakis y Ioannis X Tsiros (2020). "Machine learning summer climate classes with linear convolutional neural networks". En: *2020 8th International Conference on Renewable Energy Research and Applications (ICRERA)*. IEEE, págs. 726-731.

Bello, Irwan y col. (2017). "Neural combinatorial optimization with reinforcement learning". En: *arXiv preprint arXiv:1611.09940*.

Bengio, Yoshua, Aaron Courville y Ian Goodfellow (2020). *Deep Learning*. MIT Press.

Bishop, Christopher M. (2006a). *Pattern Recognition and Machine Learning*. Springer.

Referencias III


Bishop, Christopher M. (2006b). *Pattern Recognition and Machine Learning*. Springer.


Bottou, Léon (2010). "Large-scale machine learning with stochastic gradient descent". En: *Proceedings of COMPSTAT*, págs. 177-186.


Boykov, Yuri y Vladimir Kolmogorov (2004). "An Experimental Comparison of Min-Cut/Max-Flow Algorithms for Energy Minimization in Vision". En: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 26.9, págs. 1124-1137.


Boykov, Yuri, Olga Veksler y Ramin Zabih (2001). "Fast approximate energy minimization via graph cuts". En: *IEEE Transactions on pattern analysis and machine intelligence* 23.11, págs. 1222-1239.


Breiman, Leo (2001). "Random forests". En: *Machine learning* 45.1, págs. 5-32.


Burke, Edmund y Sanja Petrovic (2010). "The practice and theory of automated timetabling". En: *International Series in Operations Research & Management Science* 140, págs. 3-21.


Cardei, Mihaela y col. (2005). "Improved wireless sensor network lifetime with the CDS-based virtual backbone". En: *Wireless Communications and Networking Conference, 2005 IEEE* 2, págs. 1302-1307.

Referencias IV


Carter, Daniel Smilkov {and} Shan (2023). *Tensorflow — Neural Network Playground*. URL: http://playground.tensorflow.org (visitado 24-05-2023).


Cerf, Vinton (21 de ene. de 1973). *PARRY Encounters the DOCTOR*. URL: https://datatracker.ietf.org/doc/html/rfc439 (visitado 23-06-2023).


Cho, Kyunghyun y col. (2014). "Learning phrase representations using RNN encoder-decoder for statistical machine translation". En: *arXiv preprint arXiv:1406.1078*.


Chollet, François (2018). *Deep Learning with Python*. Manning Publications Co.


Chung, Junyoung y col. (2014). "Empirical evaluation of gated recurrent neural networks on sequence modeling". En: *arXiv preprint arXiv:1412.3555*.


Colbourn, Charles J. y Jeffrey H. Dinitz (2006). *Handbook of Combinatorial Designs, Second Edition*. Chapman & Hall/CRC.


Colby, Kenneth Mark, Sylvia Weber y Franklin Dennis Hilf (1971). "Artificial Paranoia". En: *Artificial Intelligence* 2.1, págs. 1-25. URL: https://www.sciencedirect.com/science/article/abs/pii/0004370271900026.

Referencias V

- Conte, Donatello y col. (2004). "Thirty years of graph matching in pattern recognition". En: *International journal of pattern recognition and artificial intelligence* 18.03, págs. 265-298.
- *ConvNet Playground* (2023). ConvNet Playground. URL: https://convnetplayground.fastforwardlabs.com (visitado 24-05-2023).
- Cortes, Corinna y Vladimir Vapnik (1995). "Support-vector networks". En: *Machine learning* 20.3, págs. 273-297.
- Denning, Peter J. (jun. de 2023). "Can Generative AI Bots Be Trusted?" En: *Communications of the ACM* 66.6, págs. 24-27. URL: https://cacm.acm.org/magazines/2023/6/273236-can-generative-ai-bots-be-trusted/fulltext.
- Diestel, Reinhard (2017). *Graph Theory*. Springer.
- *Dive into Deep Learning — Dive into Deep Learning 1.0.0-beta0 documentation* (2023). URL: https://d2l.ai/index.html (visitado 29-05-2023).
- Du, Jie y col. (2021). "Linear convolutional neural network-based modeling and control for industrial processes". En: *IEEE Access* 9, págs. 111785-111797.

Referencias VI

- Easley, David y Jon Kleinberg (2010). *Networks, Crowds, and Markets: Reasoning About a Highly Connected World*. Cambridge University Press.
- Fisher, Ronald A (1936). "The use of multiple measurements in taxonomic problems". En: *Annals of eugenics* 7.2, págs. 179-188.
- Fortunato, Santo (2010). "Community detection in graphs". En: *Physics Reports* 486.3-5, págs. 75-174.
- Fukushima, Kunihiko (abr. de 1980). "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position". En: *Biological Cybernetics* 36.4, págs. 193-202.
- Gale, David y Lloyd S. Shapley (1962). "College Admissions and the Stability of Marriage". En: *The American Mathematical Monthly* 69.1, págs. 9-15.
- Gallagher, Andrew C (2008). "Using ghost pairings to solve the permutation problem in jigsaw puzzles". En: *2008 19th International Conference on Pattern Recognition*. IEEE, págs. 1-4.
- Gao, Chao y col. (2019). "Linear convolutional neural network for industrial process control". En: *2019 IEEE 10th International Conference on Software Engineering and Service Science (ICSESS)*. IEEE, págs. 1113-1116.

Referencias VII


Gemetto, Jorge (28 de mayo de 2023). *Inteligencia artificial generativa y derechos culturales (parte 1): ¿alguien quiere pensar en las personas usuarias?* Ártica Centro Cultural. URL: https://www.articaonline.com/2023/05/inteligencia-artificial-generativa-y-derechos-culturales-parte-1-alguien-quiere-pensar-en-las-personas-usuarias/.


Gers, Felix A, Jürgen Schmidhuber y Fred Cummins (2000). "Learning to forget: Continual prediction with LSTM". En: *Neural computation* 12.10, págs. 2451-2471.


Ghosh, Anirudha y col. (2020). "Fundamental Concepts of Convolutional Neural Network". en. En: *Recent Trends and Advances in Artificial Intelligence and Internet of Things*. Ed. por Valentina E. Balas, Raghvendra Kumar y Rajshree Srivastava. Vol. 172. Series Title: Intelligent Systems Reference Library. Cham: Springer International Publishing, págs. 519-567. ISBN: 978-3-030-32643-2 978-3-030-32644-9. DOI: 10.1007/978-3-030-32644-9_36. URL: http://link.springer.com/10.1007/978-3-030-32644-9_36 (visitado 12-06-2023).


Glorot, Xavier, Antoine Bordes y Yoshua Bengio (2011). "Deep sparse rectifier neural networks". En: *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, págs. 315-323.

Referencias VIII


Golden, Bruce y col. (1980). "Computational aspects of the planar Euclidean and planar rectilinear traveling salesman problems". En: *Omega* 8.2, págs. 173-181.


Goodfellow, Ian, Yoshua Bengio y Aaron Courville (2016). *Deep Learning*. MIT Press.


Goodman, Joshua T (2001). "A bit of progress in language modeling". En: *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. Vol. 1. IEEE, págs. 6-10.


Gould, Stephen y col. (2011). "Learning Tree Conditional Random Fields". En: *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, págs. 81-88.


Graves, Alex (2012). "Supervised sequence labelling with recurrent neural networks". En: *Studies in computational intelligence*. Vol. 385. Springer, págs. 1-13.


Graves, Alex, Abdel-rahman Mohamed y Geoffrey Hinton (2013). "Speech recognition with deep recurrent neural networks". En: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, págs. 6645-6649.


Greff, Klaus y col. (2015). "LSTM: A search space odyssey". En: *arXiv preprint arXiv:1503.04069*.

Referencias IX


Gupta, Saurabh y Rakesh Agrawal (2018). "Forecasting economic time series using stacked linear convolutional neural networks". En: *Applied Soft Computing* 68, págs. 61-72.


Guyon, Isabelle y André Elisseeff (2003). "An introduction to variable and feature selection". En: *Journal of machine learning research* 3.Mar, págs. 1157-1182.


Hastie, Trevor, Robert Tibshirani y Jerome Friedman (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Science & Business Media.


Haykin, Simon (2009). *Neural networks and learning machines*. Pearson Upper Saddle River, NJ, USA:


Henrickson, Leah (16 de jul. de 2018). "The Policeman's Beard is Algorithmically Constructed". En: *3:AM Magazine*. URL: https://www.3ammagazine.com/3am/the-policemans-beard-is-algorithmically-constructed/ (visitado 13-07-2023).


Hinton, Geoffrey E y Sam T Roweis (2002). "Stochastic Neighbor Embedding". En: *Advances in Neural Information Processing Systems* 15, págs. 857-864.


Hochreiter, Sepp y Jürgen Schmidhuber (1997). "Long short-term memory". En: *Neural computation* 9.8, págs. 1735-1780.

Referencias X


How Did Scientists Succumb to Aunt Edna? The Dangers of a Superintelligent AI is Fiction — blog@CACM — Communications of the ACM (2023). URL: https://cacm.acm.org/blogs/blog-cacm/273207-how-did-scientists-succumb-to-aunt-edna-the-dangers-of-a-superintelligent-ai-is-fiction/fulltext (visitado 28-05-2023).


Hubel, David H y Torsten N Wiesel (1968). "Receptive fields and functional architecture of monkey striate cortex". En: *Journal of Physiology (London)* 195, págs. 215-243.


Hugging Face – The AI community building the future. (2023). URL: https://huggingface.co/ (visitado 24-05-2023).


Hutchins, John (2006). *The first public demonstration of machine translation: the Georgetown-IBM system, 7th January 1954*. Págs. 1-39.

Referencias XI

- Hutchison, David y col. (2010). "Evaluation of Pooling Operations in Convolutional Architectures for Object Recognition". en. En: *Artificial Neural Networks – ICANN 2010*. Ed. por Konstantinos Diamantaras, Wlodek Duch y Lazaros S. Iliadis. Vol. 6354. Series Title: Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, págs. 92-101. ISBN: 978-3-642-15824-7 978-3-642-15825-4. DOI: 10.1007/978-3-642-15825-4_10. URL: http://link.springer.com/10.1007/978-3-642-15825-4_10 (visitado 14-06-2023).
- Idury, Ramana M y Michael S Waterman (1995). "A new algorithm for DNA sequence assembly". En: *Journal of Computational Biology* 2.2, págs. 291-306.
- Jain, Anil K (2010). "Data clustering: 50 years beyond K-means". En: *Pattern recognition letters*. Vol. 31. 8. Elsevier, págs. 651-666.
- Jeon, Seung-Hyun y col. (2020). "A review of deep learning in multidimensional brain MRI". En: *Frontiers in Neuroscience* 14, pág. 235.
- Jolliffe, Ian (2002). *Principal component analysis*. Springer Series in Statistics. Springer.
- Jolliffe, Ian y Jorge Cadima (2016). *Principal Component Analysis*. Springer.

Referencias XII


Kempe, David, Jon Kleinberg y Eva Tardos (2003). "Maximizing the spread of influence through a social network". En: *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, págs. 137-146.


Kool, Wouter y col. (2019). "Attention, learn to solve routing problems!" En: *arXiv preprint arXiv:1803.08475*.


Krizhevsky, Alex, Ilya Sutskever y Geoffrey E. Hinton (2012). "Imagenet classification with deep convolutional neural networks". En: *Advances in Neural Information Processing Systems 25*. Ed. por Francisco Pereira y col. Curran Associates, Inc., págs. 1097-1105.


Kullback, Solomon (1959). *Information Theory and Statistics*. Dover Publications.


Kumar, Mitesh M. y Martial Hebert (2016). "Weighting with stochastic gradient decent". En: *European Conference on Computer Vision (ECCV)*. Springer, págs. 468-484.


Kumar, Prachi (21 de oct. de 2017). *An Introduction to N-grams: What Are They and Why Do We Need Them?* XRDS - Crossroads: The ACM Magazine for Students. URL: https://blog.xrds.acm.org/2017/10/introduction-n-grams-need/ (visitado 09-07-2023).

Referencias XIII


Langdell, James (25 de dic. de 1984). "People in the news: Bill Chamberlain". En: *PC Magazine*, pág. 64. URL: https://archive.org/details/PC-Mag-1984-12-25/page/n63/mode/2up (visitado 13-07-2023).


Lawler, Eugene L y col. (1985). *The traveling salesman problem: A guided tour of combinatorial optimization*. John Wiley & Sons, Inc.


LeCun, Yann y col. (nov. de 1998). "Gradient-based learning applied to document recognition". En: *Proceedings of the IEEE* 86.11, págs. 2278-2324.


Lee, Dembarg (8 de mayo de 1977). "Experts Argue Whether Computers Could Reason, and if They Should". En: *The New York Times*. URL: https://www.nytimes.com/1977/05/08/archives/experts-argue-whether-computers-could-reason-and-if-they-should.html (visitado 28-06-2023).


Legg, Shane y Marcus Hutter (2007). "A Collection of Definitions of Intelligence". En: *Proceedings of the 2007 Conference on Advances in Artificial General Intelligence: Concepts, Architectures and Algorithms: Proceedings of the AGI Workshop 2006*. NLD: IOS Press, págs. 17-24. ISBN: 9781586037581.


Levenshtein, Vladimir I (1966). "Binary codes capable of correcting deletions, insertions, and reversals". En: *Soviet Physics Doklady* 10.8, págs. 707-710.

Referencias XIV


Lewis, Peter H. (14 de mayo de 1985). "A new brand of lunacy for sale". En: *The New York Times*, pág. 4. URL: https://www.nytimes.com/1985/05/14/science/peripherals-a-new-brand-of-lunacy-for-sale.html (visitado 13-07-2023).


Libri, Michael y David E Rummelhart (1993). *An introduction to recurrent neural networks*. Inf. téc. 1142. AI memo, págs. 1-43.


Lin, Jie y et al. (2019). "Eulerian path-based robot and vehicle routing for autonomous transportation systems". En: *IEEE Transactions on Intelligent Transportation Systems* 20.3, págs. 1166-1177.


Liu, Bing, Jie Wang y Shuiwang Zhu (2020). "Graph Neural Networks: A Review of Methods and Applications". En: *arXiv preprint arXiv:1812.08434*.


Llinás, Rodolfo R. (ene. de 2003). "The contribution of Santiago Ramon y Cajal to functional neuroscience". en. En: *Nature Reviews Neuroscience* 4.1, págs. 77-80. ISSN: 1471-003X, 1471-0048. DOI: 10.1038/nrn1011. URL: https://www.nature.com/articles/nrn1011 (visitado 11-07-2023).

Referencias XV

López Michelone, Manuel (3 de ene. de 2015). *Un juego como Eliza llamado Parry*. UnoCero. URL: https://www.unocero.com/ciencia/un-juego-como-eliza-llamado-parry/ (visitado 30-06-2023).

Maaten, Laurens van der y Geoffrey Hinton (2008). "Visualizing data using t-SNE". En: *Journal of Machine Learning Research* 9.Nov, págs. 2579-2605.

Martinez, Aleix M y Avinash C Kak (2001). "PCA versus LDA". En: *IEEE transactions on pattern analysis and machine intelligence* 23.2, págs. 228-233.

Matzarakis, Andreas, Chrysoula Balafoutis y Ioannis X Tsiros (2019). "Machine learning approach using linear convolutional neural networks for summer climate classification". En: *International Journal of Biometeorology* 63.2, págs. 247-256.

McCulloch, Warren S. y Walter Pitts (dic. de 1943). "A logical calculus of the ideas immanent in nervous activity". en. En: *The Bulletin of Mathematical Biophysics* 5.4, págs. 115-133. ISSN: 0007-4985, 1522-9602. DOI: 10.1007/BF02478259. URL: http://link.springer.com/10.1007/BF02478259 (visitado 11-07-2023).

McInnes, Leland, John Healy y James Melville (2018). "UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction". En: *arXiv preprint arXiv:1802.03426*.

Referencias XVI

- Miller, Colin y col. (1995). "Vehicle routing in practice". En: *Interfaces* 25.1, págs. 116-130.
- Mislove, Alan y col. (2007). "Measurement and analysis of online social networks". En: *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*. ACM, págs. 29-42.
- Molnar, Christoph (2020). *Interpretable Machine Learning*. Lulu.com.
- Mu, Andreas C (s.f.). "Introduction to Machine Learning with Python". En: ().
- Müller, Andreas C. y Sarah Guido (2016). *Introduction to Machine Learning with Python: A Guide for Data Scientists*. O'Reilly Media, Inc.
- Nair, Vinod y Geoffrey E. Hinton (2010). "Rectified linear units improve restricted boltzmann machines". En: *Proceedings of the 27th International Conference on Machine Learning (ICML)*. ACM, págs. 807-814.
- Niu, Rui y col. (2018). "Multidimensional convolutional neural networks for time series classification". En: *Proceedings of the 2018 SIAM International Conference on Data Mining*. SIAM, págs. 393-401.

Referencias XVII


Norberg, Arthur L. (dic. de 1991). *An Interview with Terry Allen Winograd*. English. URL: https://conservancy.umn.edu/bitstream/handle/11299/107717/oh237taw.pdf (visitado 30-06-2023).


Novikoff, Albert B (1962). "On the convergence proofs on perceptrons". En: *Proceedings of the symposium on the mathematical theory of automata* 12, págs. 615-622.


Nwankpa, Chigozie y col. (2018). "Activation Functions: Comparison of trends in Practice and Research for Deep Learning". En: *arXiv preprint arXiv:1811.03378*.


Oliver, Whang (30 de mayo de 2023). "The Race to Make A.I. Smaller (and Smarter)". En: *The New York Times*. URL: https://www.nytimes.com/2023/05/30/science/ai-chatbots-language-learning-models.html (visitado 04-06-2023).


Oord, Aaron van den y col. (2016). "WaveNet: A Generative Model for Raw Audio". En: *arXiv preprint arXiv:1609.03499*.


Papadimitriou, Christos H. y Kenneth Steiglitz (1982). *Combinatorial Optimization: Algorithms and Complexity*. Dover Publications.

Referencias XVIII


Pedregosa, Fabian y col. (2011). "Scikit-learn: Machine learning in Python". En: *the Journal of machine Learning research* 12, págs. 2825-2830.


Pentico, David W (2007). "Assignment problems: A golden anniversary survey". En: *European Journal of Operational Research* 176.2, págs. 774-793.


Pinedo, Michael L (2012). *Scheduling: theory, algorithms, and systems*. Springer.


Racter, William Chaberlain y Joan Hall (1984). *The policeman's beard is half constructed: computer prose and poetry by Racter*. New York, NY: Warner Books. ISBN: 0-446-38051-2. URL: https://archive.org/details/policemansbeardi0000unse/.


Ravi, R y col. (1993). "Many faces of multicommodity network flow". En: *Networks* 23.6, págs. 539-558.


Ringnér, Markus (2008). "What is principal component analysis?" En: *Nature biotechnology* 26.3, págs. 303-304.


Rosenblatt, Frank (1957). "The Perceptron–a perceiving and recognizing automaton". En: *Report*.


— (1958a). "The perceptron: a probabilistic model for information storage and organization in the brain.". En: *Psychological review* 65.6, pág. 386.

Referencias XIX


Rosenblatt, Frank (1958b). "The perceptron: a probabilistic model for information storage and organization in the brain.". En: *Psychological review* 65.6, pág. 386.


Roweis, Sam T y Lawrence K Saul (2000). "Nonlinear dimensionality reduction by locally linear embedding". En: *Science* 290.5500, págs. 2323-2326.


Rumelhart, David E., Geoffrey E. Hinton y Ronald J. Williams (1986). "Learning representations by back-propagating errors". En: *Nature* 323.6088, págs. 533-536.


Runway (2023). Runway. URL: https://app.runwayml.com (visitado 24-05-2023).


Russakovsky, Olga y col. (2015). "ImageNet Large Scale Visual Recognition Challenge". En: *International Journal of Computer Vision (IJCV)* 115.3, págs. 211-252.


Schölkopf, Bernhard y Alexander J. Smola (2002). *Learning with kernels: support vector machines, regularization, optimization, and beyond*. Adaptive computation and machine learning. Cambridge, Mass: MIT Press. 626 págs. ISBN: 978-0-262-19475-4.


Schuster, Mike y Kuldip K Paliwal (1997). "Bidirectional recurrent neural networks". En: *IEEE Transactions on Signal Processing* 45.11, págs. 2673-2681.

Referencias XX


Sharp, Michael (26 de oct. de 2022). *Teaching Computers to Read 'Industry Lingo' — Technical vs. Natural Language Processing*. Just a Standard Blog. URL: https://www.nist.gov/blogs/taking-measure/teaching-computers-read-industry-lingo-technical-vs-natural-language-processing (visitado 22-06-2023).


Shi, Hongzhi y col. (2021). "Economic time series forecasting with a linear convolutional neural network". En: *International Journal of Forecasting* 37.4, págs. 1575-1592.


Simonyan, Karen y Andrew Zisserman (2014). "Very Deep Convolutional Networks for Large-Scale Image Recognition". En: *arXiv preprint arXiv:1409.1556*.


Smith, Jason (2017). "A Gentle Introduction to Cross-Entropy for Machine Learning". En: *Journal of Machine Learning Research* 18.67, págs. 1-25.


Su, Xiaoyuan y Taghi M Khoshgoftaar (2009). "A survey of collaborative filtering techniques". En: *Advances in artificial intelligence* 2009, pág. 4.


Szegedy, Christian y col. (2015). "Going deeper with convolutions". En: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, págs. 1-9.

Referencias XXI


Tero, Atsushi y col. (ene. de 2010). "Rules for Biologically Inspired Adaptive Network Design". en. En: *Science* 327.5964, págs. 439-442. ISSN: 0036-8075, 1095-9203. DOI: 10.1126/science.1177894. URL: https://www.science.org/doi/10.1126/science.1177894 (visitado 12-06-2023).


Theodoridis, Sergios y Sergios Theodoridis (2015). *Machine learning: a Bayesian and optimization perspective*. London San Diego: Elsevier Academic Press. ISBN: 978-0-12-801522-3.


Thomas, Haigh (jun. de 2023). "Cojoined Twins: Artificial Intelligence and the Invention of Computing Science". English. En: *Communications of the ACM* 66.6, págs. 33-37. URL: https://cacm.acm.org/magazines/2023/6/273239-conjoined-twins-artificial-intelligence-and-the-invention-of-computer-science/fulltext.


Tian, Ying y Weibin Dong (2019). "Research on linear convolutional neural network and its application in industrial process control". En: *2019 IEEE 3rd Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC)*. IEEE, págs. 1991-1995.


Toth, Paolo y Daniele Vigo (2002). *The vehicle routing problem*. SIAM.

Referencias XXII


Valipour, Mohammad (2017). "Linear CNN model for rainfall-runoff modeling". En: *Journal of Hydrology* 549, págs. 594-603.


Vinyals, Oriol, Meire Fortunato y Navdeep Jaitly (2015). "Pointer networks". En: *Advances in Neural Information Processing Systems* 28, págs. 2692-2700.


Wang, Wencheng y col. (2018). "Linear convolutional neural network for industrial process control based on compressed sensing". En: *Neural Computing and Applications* 30.7, págs. 2221-2230.


Weizenbaum, Joseph (1966). "ELIZA—a computer program for the study of natural language communication between man and machine". En: *Communications of the ACM* 9.1, págs. 36-45. URL: https://dl.acm.org/doi/abs/10.1145/357980.357991.


Willems, Klaas y Gerda Janssens (2020). *Data Science for Business*. O'Reilly Media, Inc.


Winograd, Terry (ene. de 1971). "Procedures as a representation for data in a computer program for understanding natural language". Tesis doct. Massachusetts Institute of Technology. 461 págs. URL: https://dspace.mit.edu/handle/1721.1/7095 (visitado 23-06-2023).

Referencias XXIII


Winogard, Terry y Fernando Flores (1986). *Understanding computers and cognition: a new foundation for design*. Ablex Publishing Corporation. ISBN: 0-89391-050-3. URL: https://archive.org/details/understandingcom00wino/.


Wu, Hao, Mubashir Shah y Yi Qian (2017). "Convolutional recurrent neural networks for dynamic MR image reconstruction". En: *IEEE Transactions on Medical Imaging* 36.7, págs. 1532-1541.


Yan, Peng y col. (2017). "Linear convolutional neural networks for stock price trend prediction". En: *2017 2nd International Conference on Computer Science and Technology (CST)*. IEEE, págs. 359-364.


Zahn, Charles T. (1971). "Graph-theoretical methods for detecting and describing gestalt clusters". En: *IEEE Transactions on Computers* C-20.1, págs. 68-86.


Zeiler, Matthew D. y Rob Fergus (2014). "Visualizing and Understanding Convolutional Networks". En: *European Conference on Computer Vision (ECCV)*.


Zhang, Chao y col. (2018). "Linear convolutional neural networks for economic time series forecasting". En: *2018 37th Chinese Control Conference (CCC)*. IEEE, págs. 2035-2039.


Zhang, Tuo y Yuan Yuan (2019). "Optimization Methods for Large-Scale Machine Learning". En: *Foundations and Trends in Machine Learning* 12.1, págs. 1-153.

Referencias XXIV

 Zhou, Jie y col. (2020). "Graph neural networks: A review of methods and applications". En: *AI Open* 1, págs. 57-81.