

“REDES NEURONALES ARTIFICIALES: CONCEPTOS BÁSICOS Y APLICACIONES”

**Mexican NLP Summer School
2021**

Co-located event with NAACL 2021 (June 2-4, 2021)

DRA. LETICIA C. CAGNINA



Universidad Nacional de
San Luis - Argentina

lcagnina@gmail.com



CONICET
Consejo Nacional de
Investigaciones Científicas
y Técnicas - Argentina

TEMARIO

- ❖ Definición
- ❖ Surgimiento de las Redes Neuronales Artificiales (RNAs)
- ❖ Componentes de una RNA
- ❖ Evolución de las RNAs
- ❖ Arquitecturas de RNAs
- ❖ Funcionamiento de una RNA
- ❖ Entorno de trabajo
- ❖ Aprendizaje Profundo
- ❖ Aplicaciones de las RNAs

DEFINICIÓN

Una red neuronal es una máquina diseñada para modelar la forma en la que el cerebro realiza una determinada tarea o función de interés (...) implementada con componentes electrónicos o simulada en software (Haykin, 2009).

Básicamente es un modelo matemático que toma como entrada un vector de números (datos), calcula la suma pesada de esos valores y los procesa para obtener la salida, utilizando funciones de activación no lineales.

HISTORIA



FIG. 1 — Organization of a biological brain. (Red areas indicate active cells, responding to the letter X.)

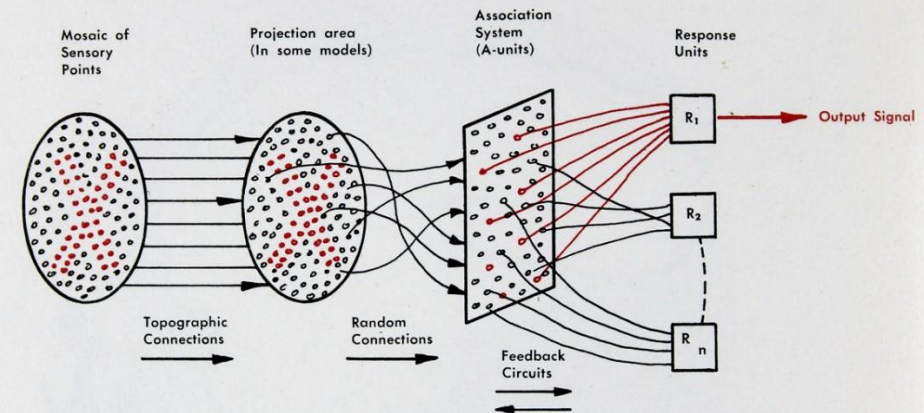
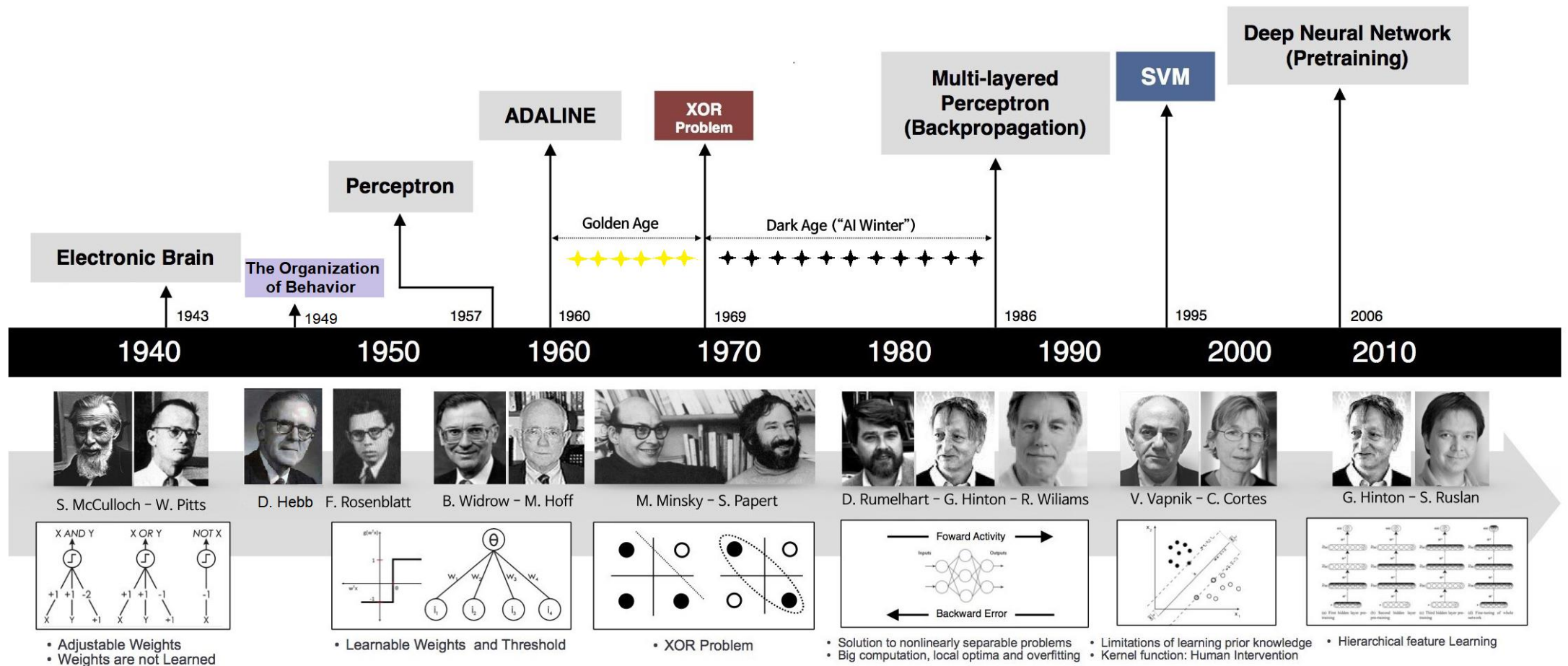


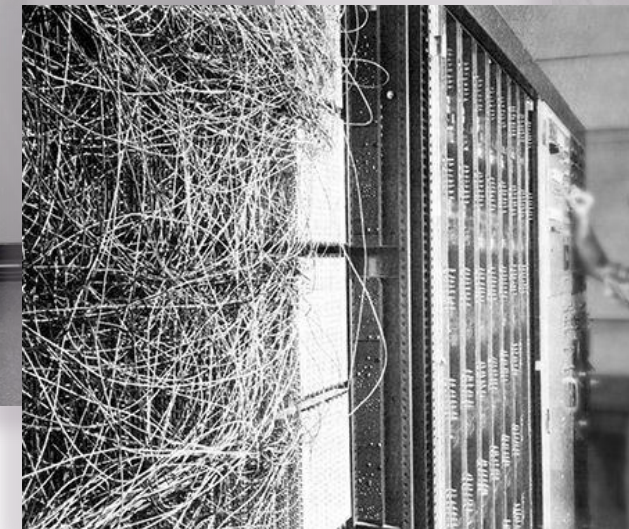
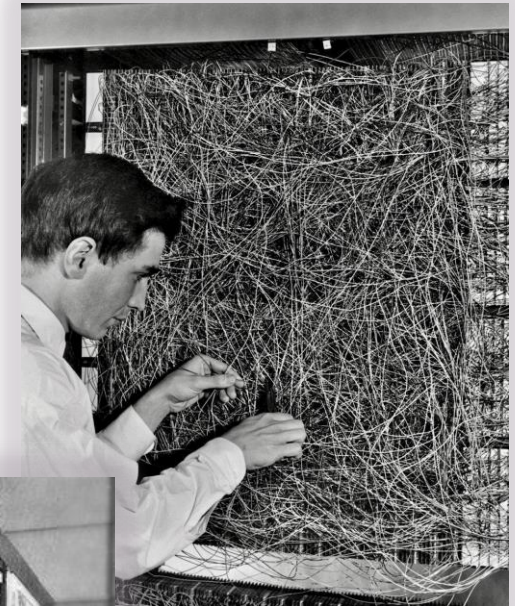
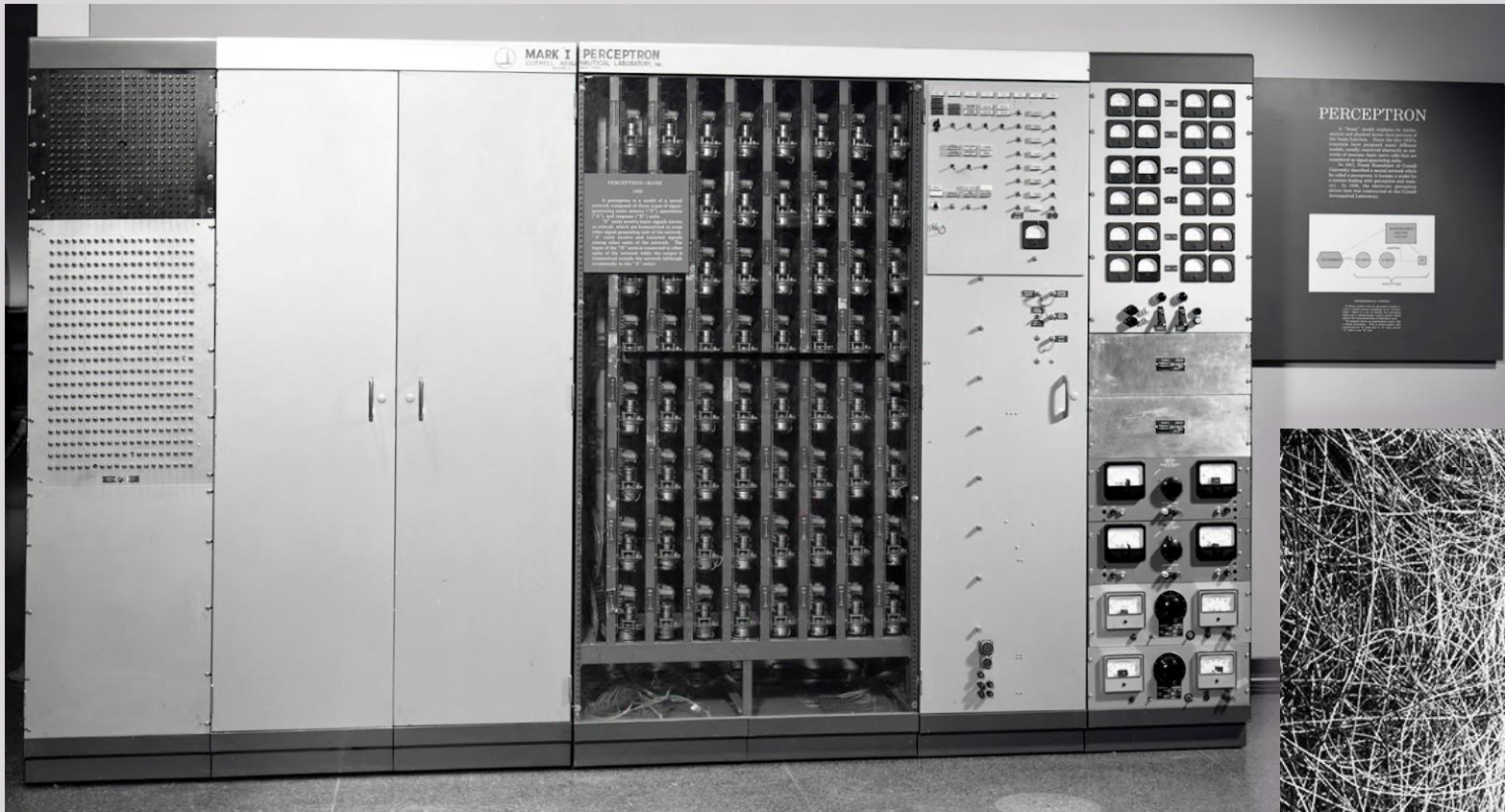
FIG. 2 — Organization of a perceptron.

Imagen de <https://tripleampersand.org/kernelled-connections-perceptron-diagram/>

HISTORIA



HISTORIA — MARK I - PERCEPTRON (1958)



Imágenes de https://americanhistory.si.edu/collections/search/object/nmah_334414
<https://www.nzz.ch/digital/ehre-fuer-die-deep-learning-mafia-ld.1472761?reduced=true>
<https://blogs.umass.edu/comphon/2017/06/15/did-frank-rosenblatt-invent-deep-learning-in-1962/>

ANALOGÍA

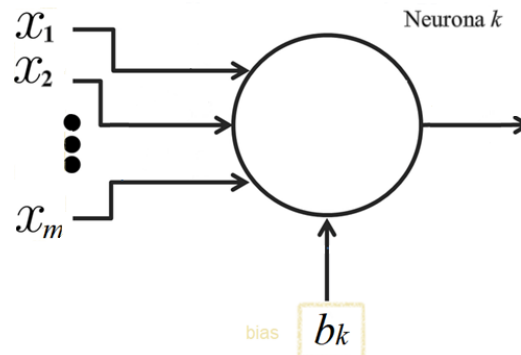
Las neuronas se conectan a otras para estimular o inhibir su actividad, formando redes neuronales que se traducen en circuitos capaces de procesar la información entrante y producir una respuesta...



Imagen de <https://concepto.de/neurona/>

COMPONENTES DE UNA RNA

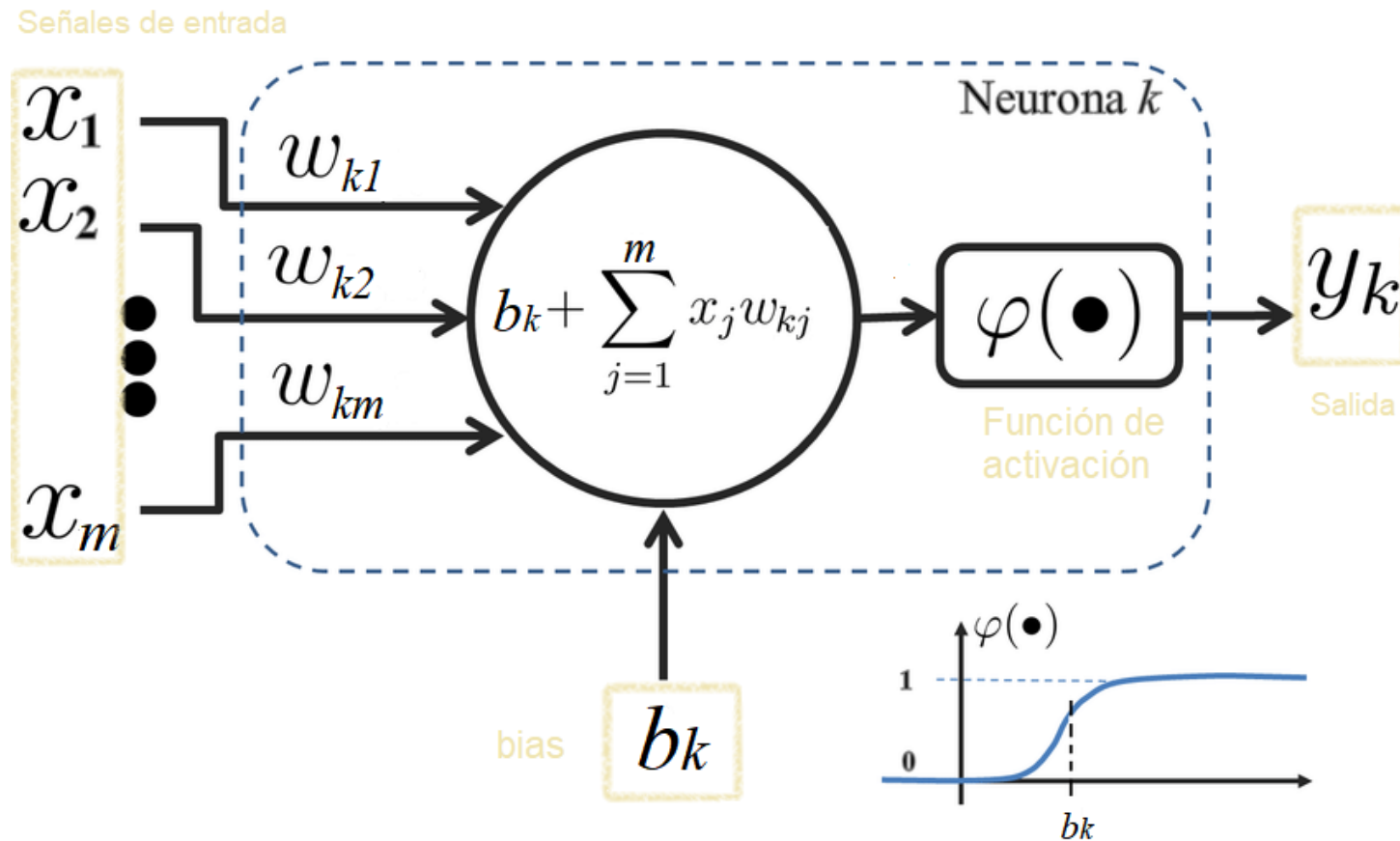
- ❖ Conjunto de valores/estímulos/señales de entrada x_j , cada uno de los cuales representa el estímulo a la neurona k
- ❖ Valor de salida de la neurona k (resultante de la combinación de los valores de entrada) y en base al cual se activará/disparará o no, dependiendo un umbral
- ❖ Valor *bias* b (sesgo) como mecanismo adicional de modificación del resultado de la neurona. Se utiliza para incrementar o decrementar la entrada de la función de activación (transformación)



COMPONENTES DE UNA RNA

- ❖ Conjunto de sinapsis o links de conexiones w que representan pesos (ponderaciones) de las señales de entrada x_j de la neurona k
- ❖ La neurona es la unidad de procesamiento de información de una RNA: básicamente realizará la suma pesada de las entradas y dependiendo si el resultado es mayor a un umbral, la neurona se activará o no
- ❖ El **sumador** (función sumatoria) para todas las señales de entrada pesadas, puede ser tan simple como el producto punto entre dos vectores o bastante compleja
- ❖ La **función de activación** φ que limita (acota) la amplitud de la salida y de la neurona $[0,1]$, por ejemplo)

COMPONENTES DE UNA RNA



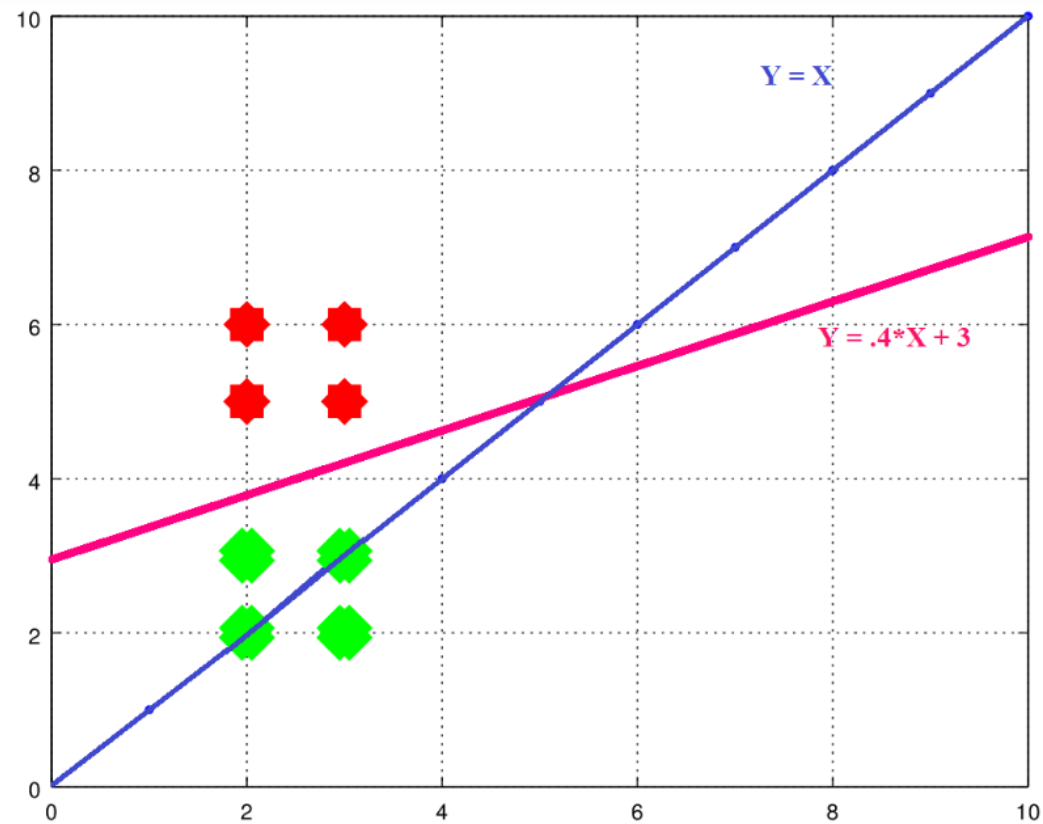
COMPONENTES DE UNA RNA

Matemáticamente...

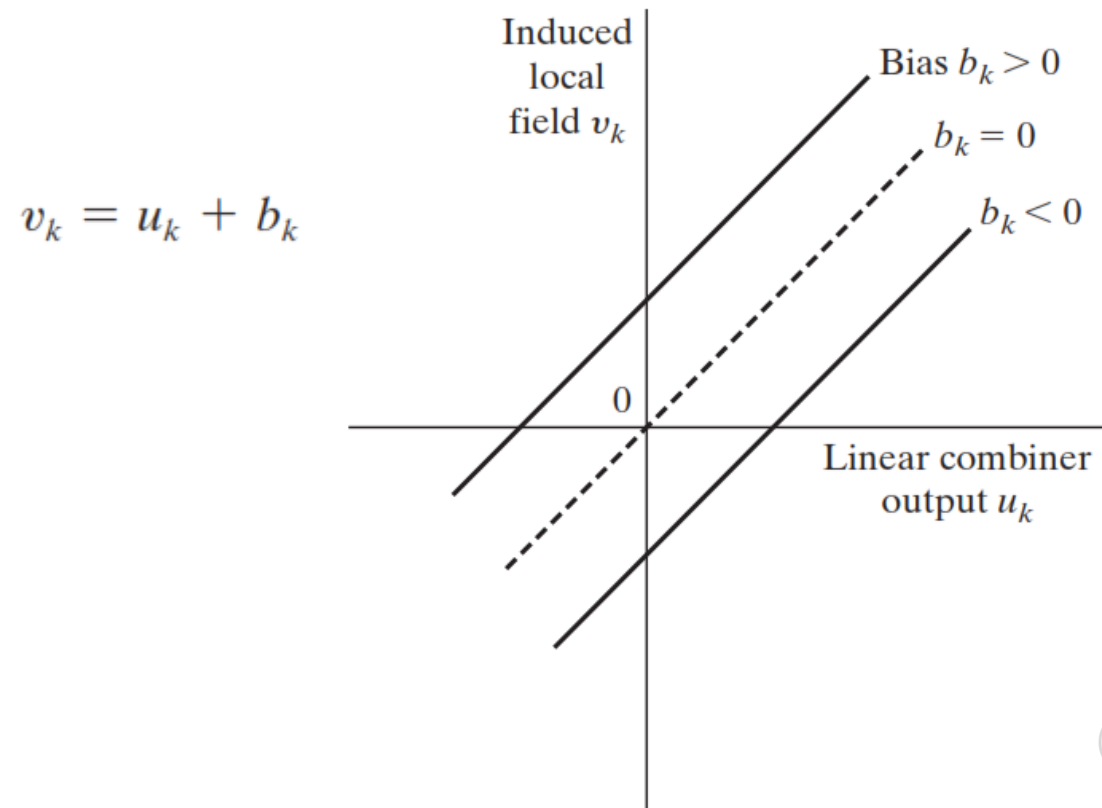
$$u_k = \sum_{j=1}^m w_{kj} x_j$$

$$y_k = \varphi(u_k + b_k)$$

COMPONENTES DE UNA RNA



COMPONENTES DE UNA RNA



(Haykin, 2009)

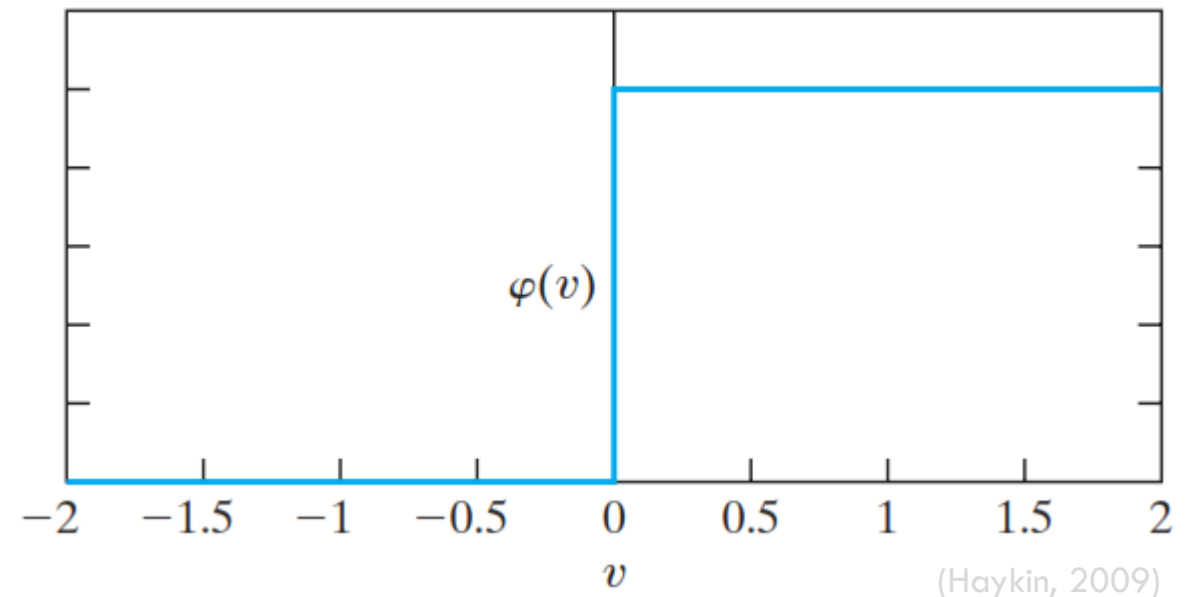
COMPONENTES DE UNA RNA

Función de activación: determina el valor de la salida de una neurona.

❖ **Función umbral/Escalón:**

(McCulloch–Pitts model, 1943)

$$\varphi(v) = \begin{cases} 1 & \text{if } v \geq 0 \\ 0 & \text{if } v < 0 \end{cases}$$



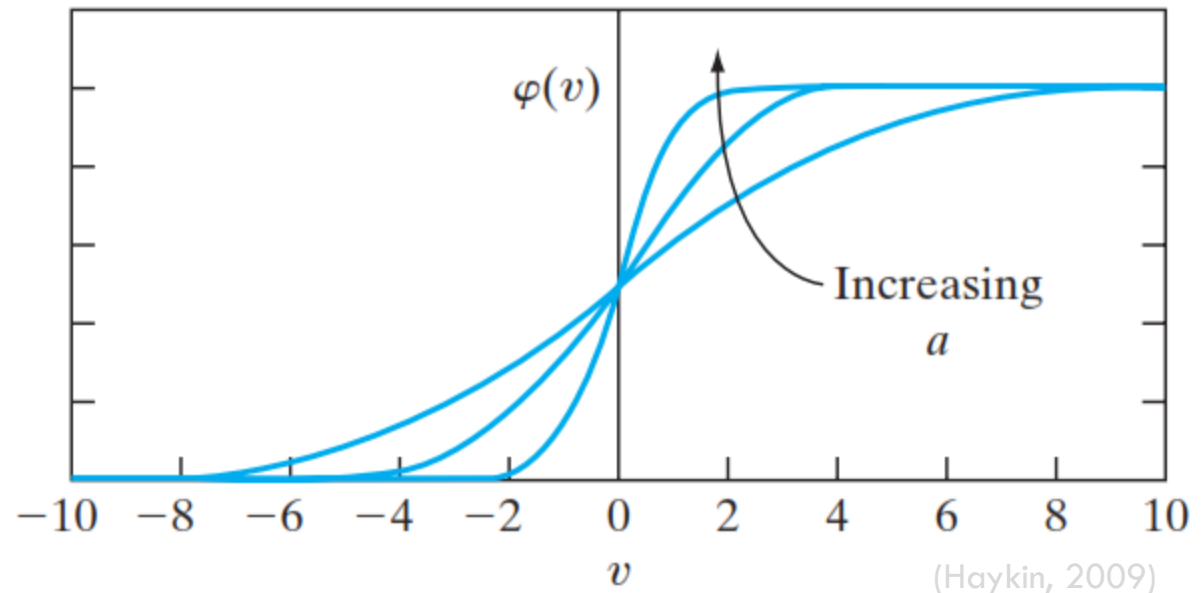
(Haykin, 2009)

COMPONENTES DE UNA RNA

Función de activación: determina el valor de la salida de una neurona.

❖ **Función sigmoide:**

$$\varphi(v) = \frac{1}{1 + \exp(-av)}$$



(Haykin, 2009)

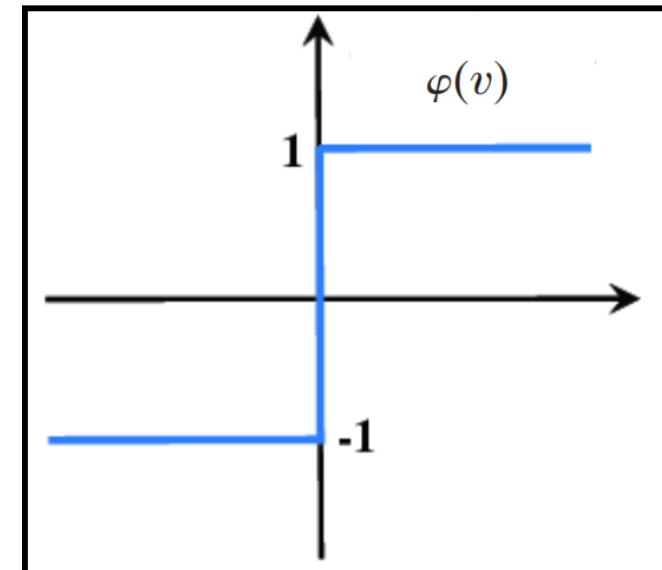
COMPONENTES DE UNA RNA

Función de activación: determina el valor de la salida de una neurona.

❖ **Función signo:**

generalización para $[-1,1]$

$$\varphi(v) = \begin{cases} 1 & \text{if } v > 0 \\ 0 & \text{if } v = 0 \\ -1 & \text{if } v < 0 \end{cases}$$



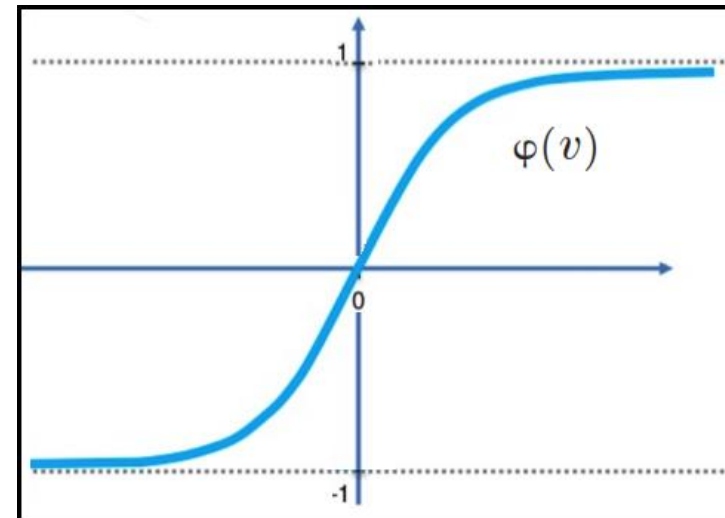
COMPONENTES DE UNA RNA

Función de activación: determina el valor de la salida de una neurona.

❖ **Función tangente hiperbólica:**

generalización para $[-1,1]$

$$\varphi(v) = \tanh(v) = \frac{\exp(v) - \exp(-v)}{\exp(v) + \exp(-v)}$$



LA NEURONA SIGMOIDE (1986)

- ❖ Entradas reales
- ❖ Utiliza pesos para expresar la importancia de las entradas
- ❖ Puede ser entrenada usando la técnica del descenso del gradiente
- ❖ La salida es un valor entre 0 y 1 (probabilidad)
- ❖ Introduce la no-linealidad (en contraste con el perceptrón)

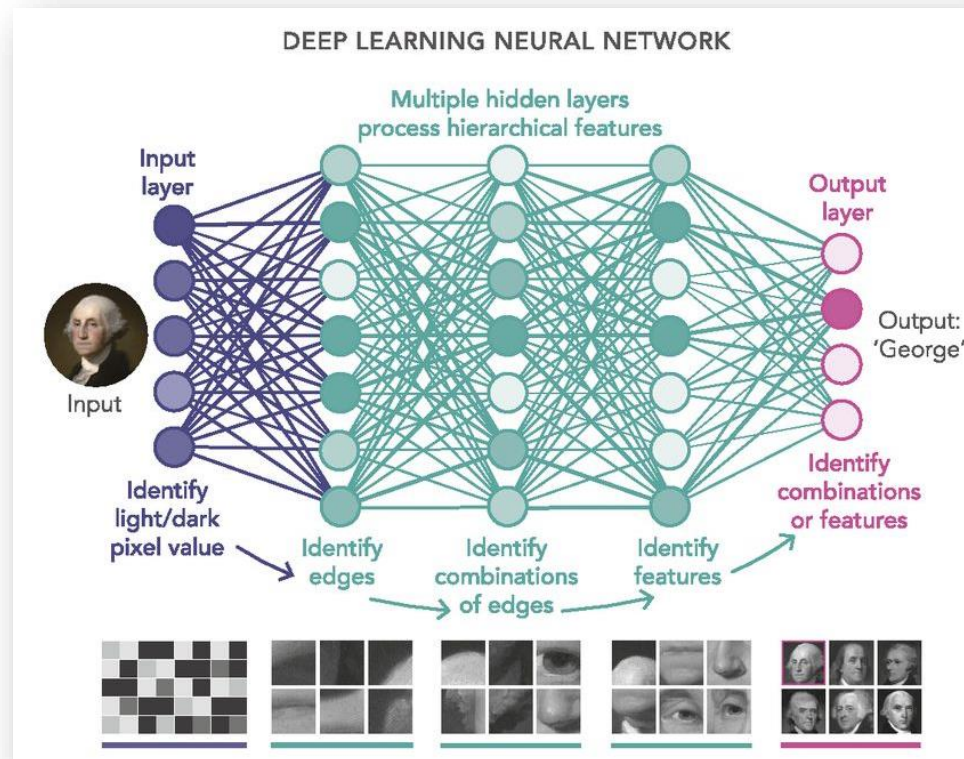
EVOLUCIÓN DE LA RNA

Deep learning

- ❖ RNA con varias capas ocultas
- ❖ La función de activación es una componente importante (no lineales!)
- ❖ Pueden representar funciones muy complejas
- ❖ Arquitecturas profundas son capaces de aprender características en diferentes niveles de abstracción

EVOLUCIÓN DE LA RNA

Deep learning

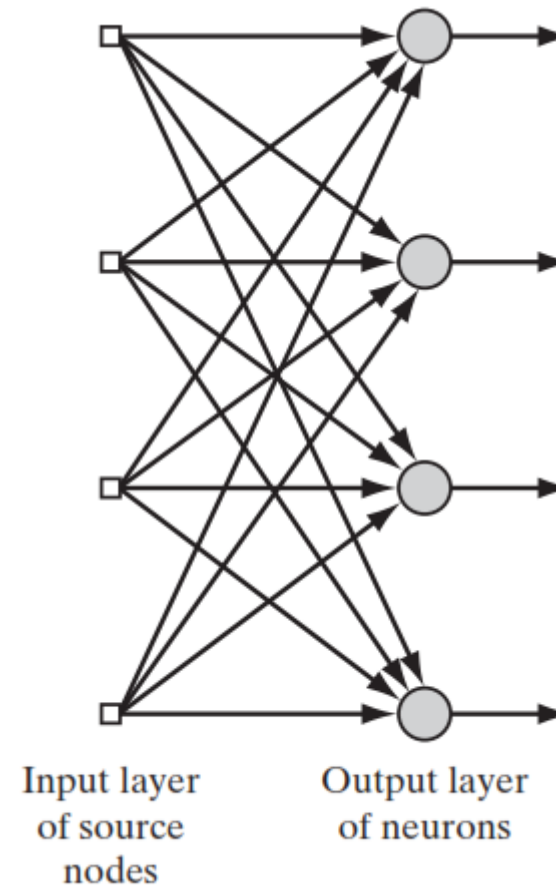


<https://www.pnas.org/content/116/4/1074>

ARQUITECTURAS

Redes *feedforward* de única capa

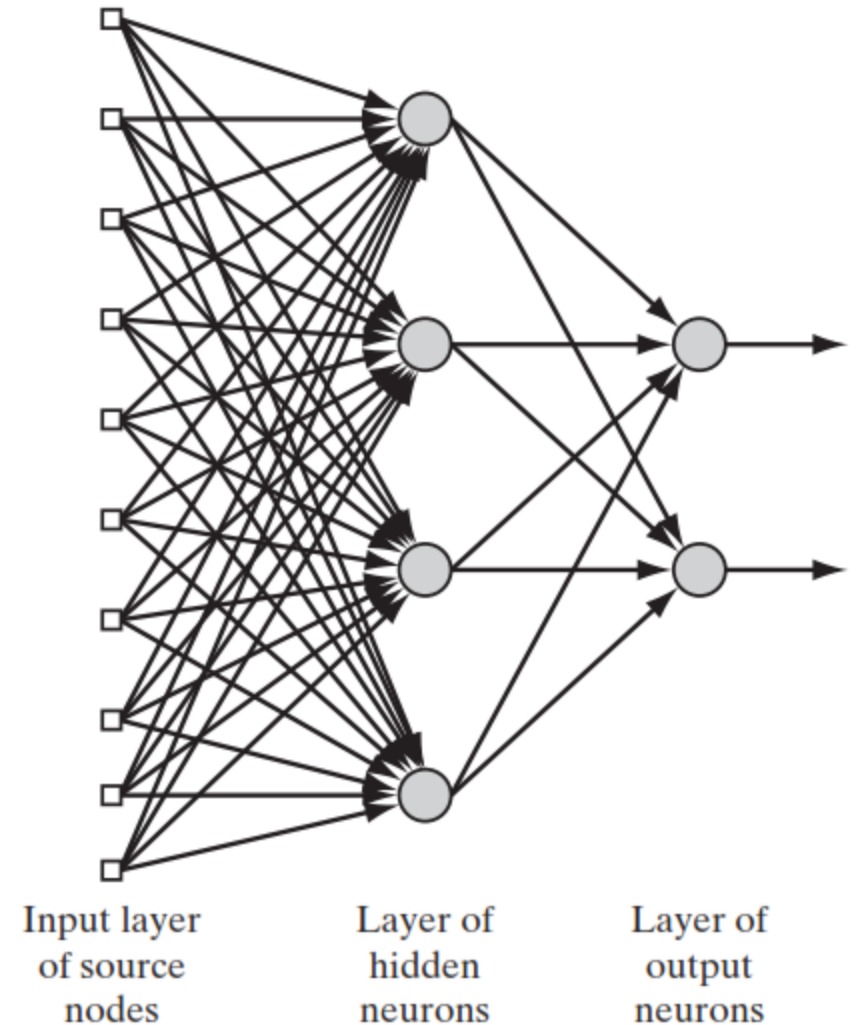
- ❖ Las neuronas están organizadas en 1 capa
- ❖ Es la arquitectura más simple (y acotada)
- ❖ Las entradas (señales) son proyectadas directamente en la capa (única) de salida
- ❖ *feedforward*: la información se procesa en un sólo sentido



ARQUITECTURAS

Redes *feedforward* multicapa

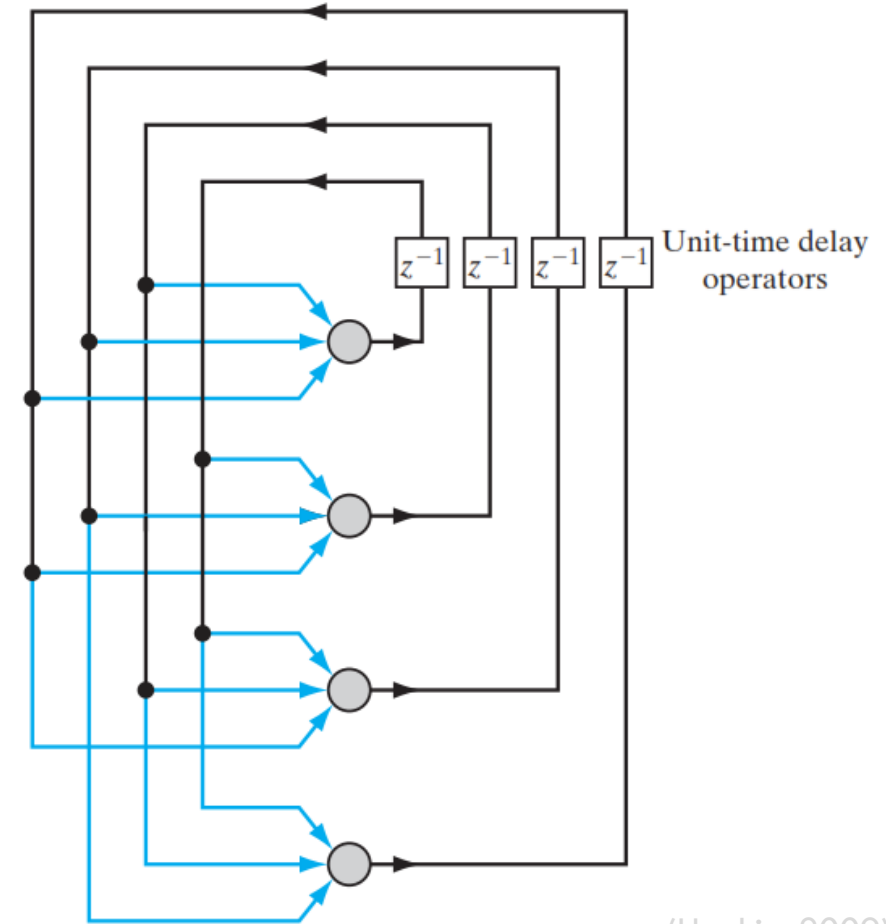
- ❖ Posee una o más *capas ocultas* de neuronas (o unidades) ocultas (no son entrada ni neuronas de salida)
- ❖ Las capas ocultas extraen información de orden más alto (detallada)
- ❖ La salida de una capa (señales) es la entrada de la siguiente



ARQUITECTURAS

Redes recurrentes

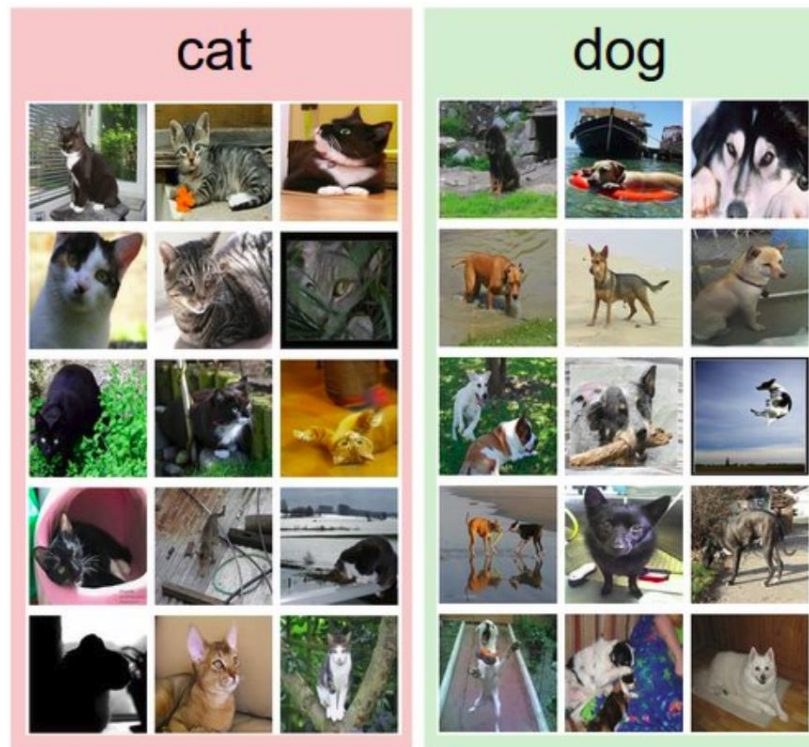
- ❖ Posee al menos un loop *feedback*
- ❖ Puede ser única o multicapa
- ❖ Los loops *feedback* tienen gran impacto en la capacidad de aprendizaje de la red y su performance



(Haykin, 2009)

FUNCIONAMIENTO

Clasificando fotos de perros y gatos...



In a nutshell

FUNCIONAMIENTO

Clasificando fotos de perros y gatos...

In a nutshell

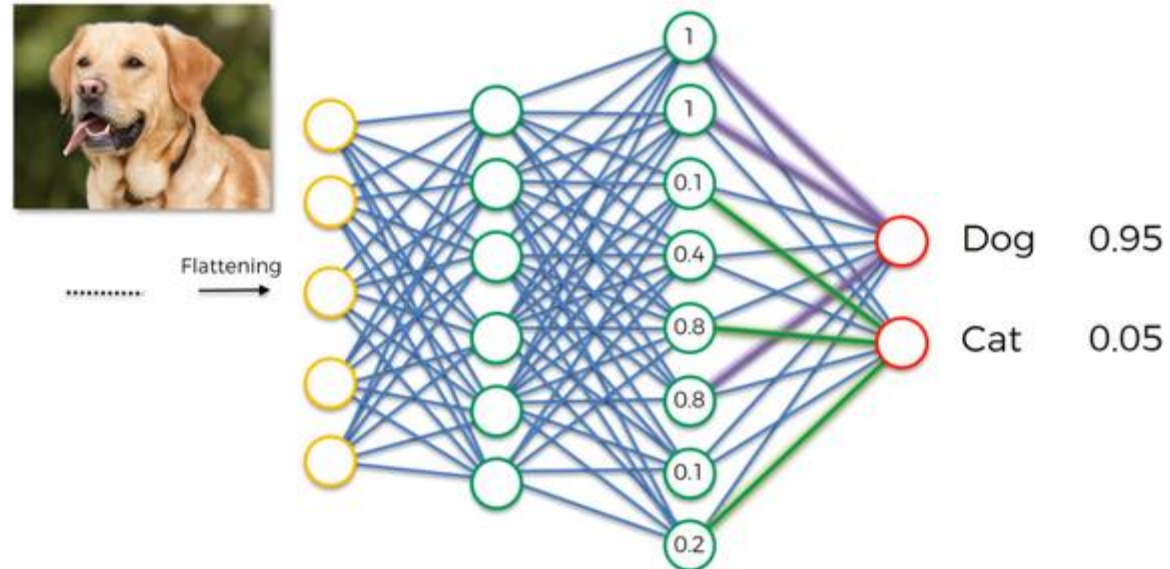


Imagen de <https://vishalj7.github.io/Convolutional-Neural-Networks/>

FUNCIONAMIENTO

- ❖ Proveer datos de entrenamiento (suficientes) para que la red “aprenda”
- ❖ Utilizar la red para clasificar datos nuevos

In a nutshell

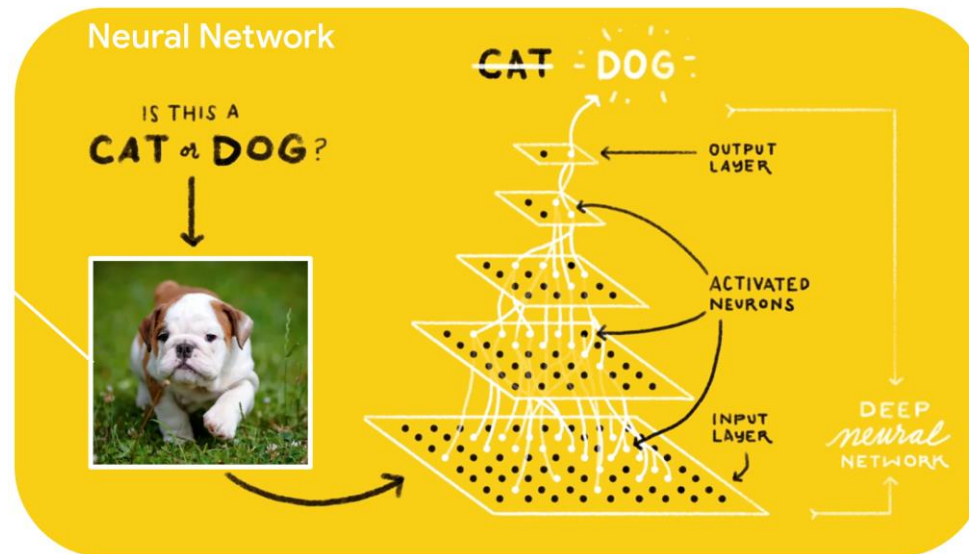
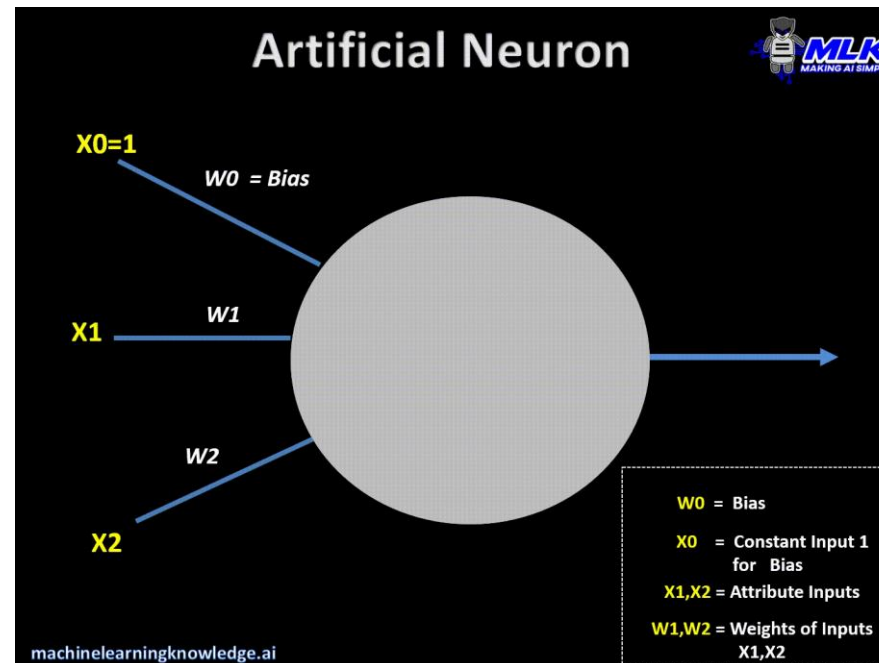


Imagen de https://www.tensorflow.org/neural_structured_learning?hl=es-419

FUNCIONAMIENTO

- ❖ Los datos de entrada son utilizados en la primera capa de la red para efectuar las sumatorias ponderadas y calcular el valor de la función de activación (pasa la señal?, cuánto pasa?) en cada una de las neuronas de la capa

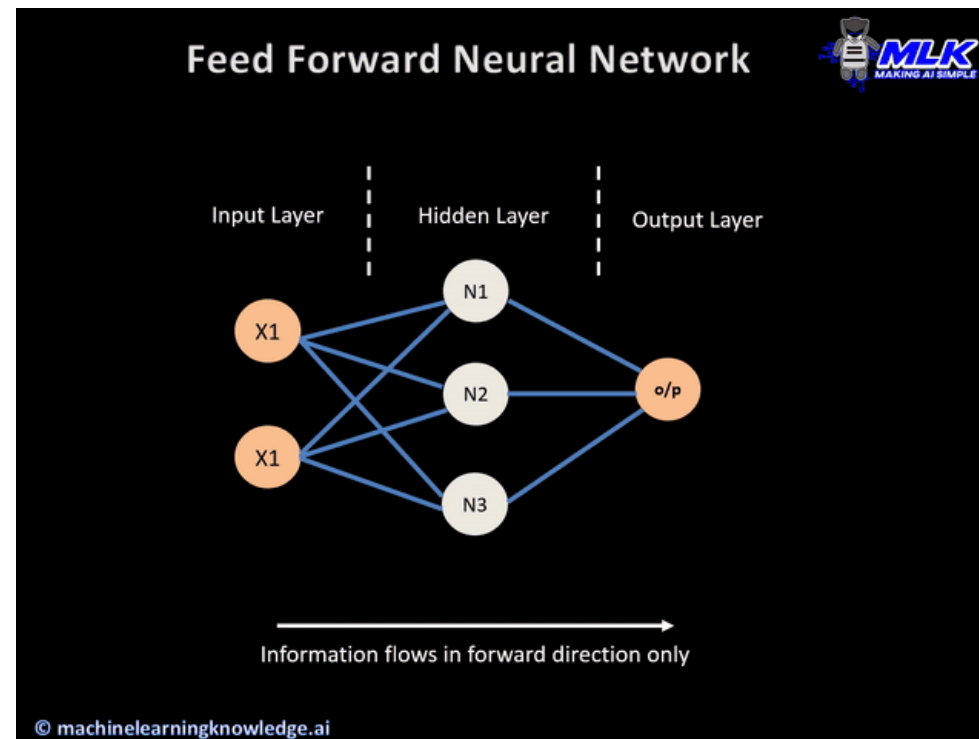
In a nutshell



FUNCIONAMIENTO

- ❖ Las operaciones se repiten hasta llegar a la última capa de la red en la cual se decide el valor de salida de una manera comprensible

In a nutshell



FUNCIONAMIENTO

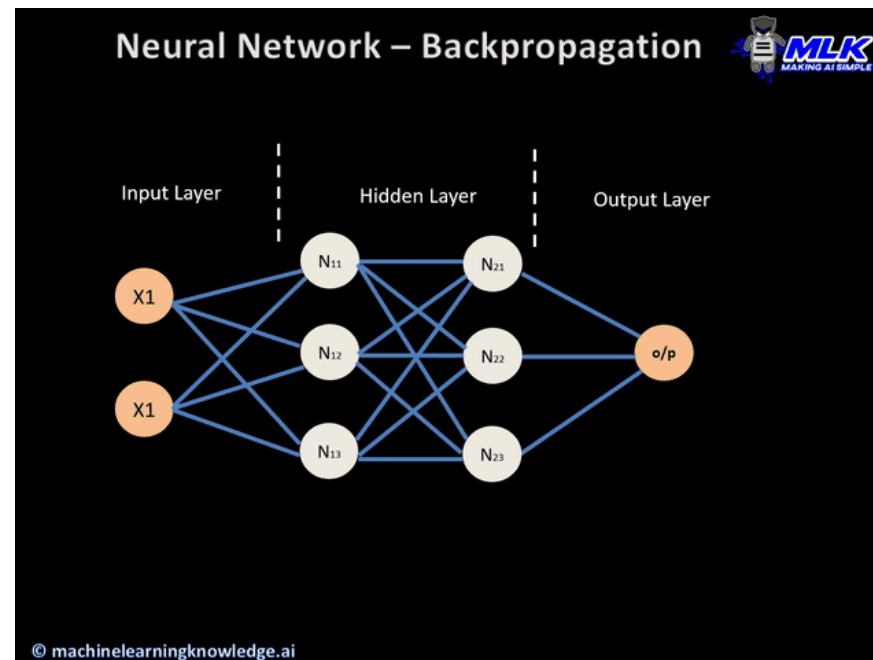
- ❖ La red evalúa la salida con respecto al verdadero valor esperado (ground truth) utilizando una **función de costo** (valor real vs valor predicho por la red) también conocida como **función de pérdida** (**loss function**) la cual debe ser minimizada

In a nutshell

FUNCIONAMIENTO

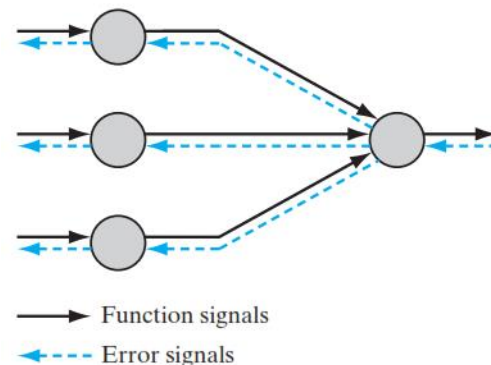
- ❖ La información sobre los errores se propaga a las capas anteriores para que comience un nuevo ciclo de **ajuste de pesos** y así minimizar la función de pérdida: **backpropagation**

In a nutshell



ENTRENAMIENTO: BACKPROPAGATION

- ❖ *Forward phase*: se fijan los pesos y las entradas se propagan capa a capa hasta llegar a la capa de salida. Los cambios son los producidos por las funciones de activación (salidas de las neuronas)
- ❖ *Backward phase*: se genera una señal de error comparando la salida de la red con la real (pérdida). Esta señal es propagada hacia atrás por toda la red, capa a capa. En este proceso, los cambios son los ajustes que se realizan a los pesos



(Haykin, 2009)

ENTRENAMIENTO: BACKPROPAGATION

❖ ¿Cómo evaluamos la performance de la RNA?: existen varias funciones de pérdida/costo que permiten cuantificar el error del modelo...

Mean Squared Error (MSE) $Loss = \frac{1}{n} * \sum_{i=0}^n (Y_i - Y_{pred_i})^2$

[sparse]Categorical Crossentropy ([SCC]CC) $Loss = - \sum_i^C Y * \log(Y_{pred})$

Binary Crossentropy (BCE) $Loss = (Y)(-\log(Y_{pred})) + (1 - Y)(-\log(1 - Y_{pred}))$

Remains when Y = 1

Remains when Y = 0

Removed when Y = 0

Removed when Y = 1

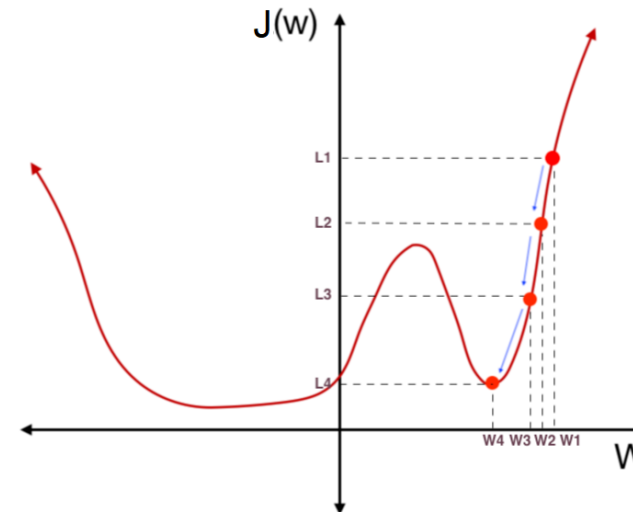
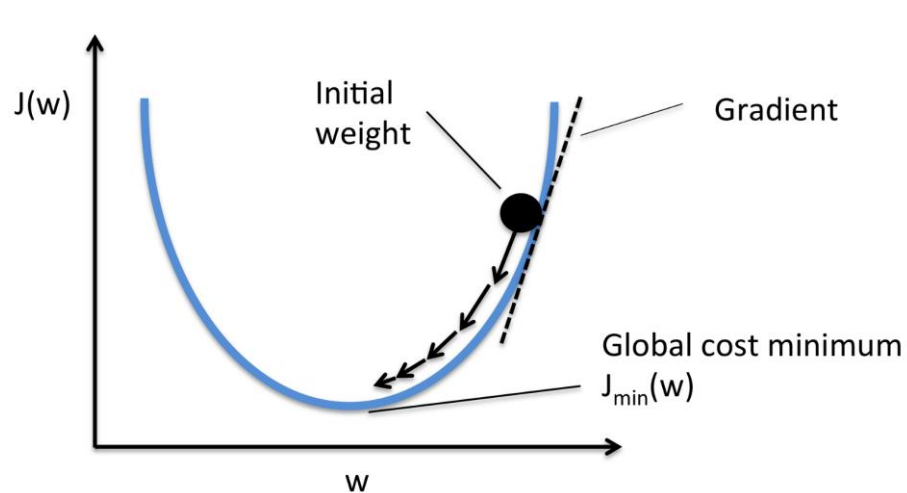
ENTRENAMIENTO: BACKPROPAGATION

- ❖ *¿Cómo evaluamos la performance de la RNA?*: existen varias funciones de pérdida/costo que permiten cuantificar el error del modelo...
- ❖ *¿Cómo podemos minimizar el error de la RNA?*: se necesita encontrar los pesos W que minimizan el error (o pérdida) de la RNA (training), es decir optimizar la función de pérdida cuyas variables serán los pesos...

El descenso del gradiente

ENTRENAMIENTO: BACKPROPAGATION

El descenso del gradiente: Inicializar los pesos de la red de forma aleatoria para calcular la función de pérdida J . Buscar la dirección que produzca la pendiente descendente más pronunciada en el valor de la función. Mover los pesos hacia esos valores (dirección opuesta a la del gradiente).



ENTRENAMIENTO: BACKPROPAGATION

Para una función con dos variables x e y ...

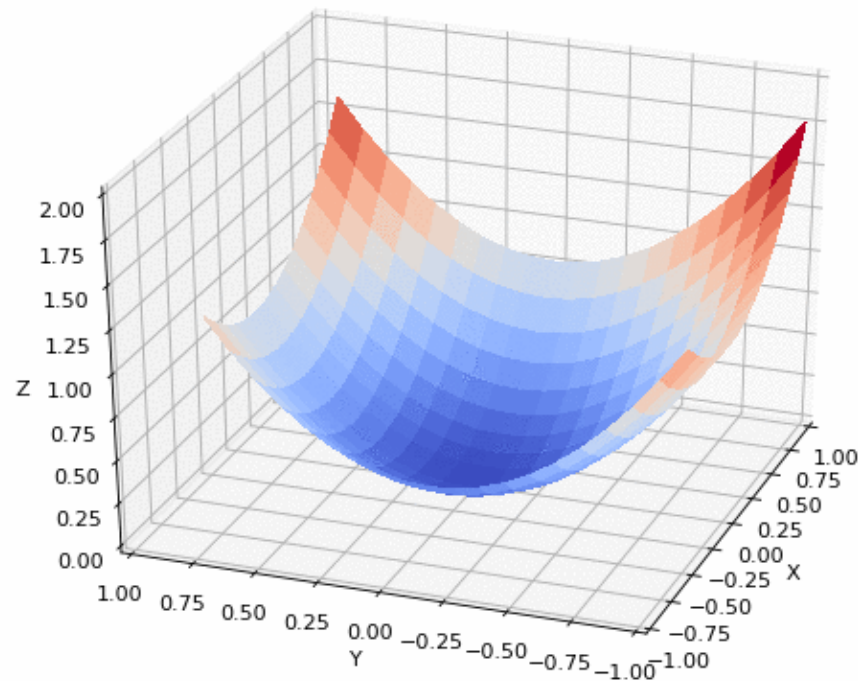
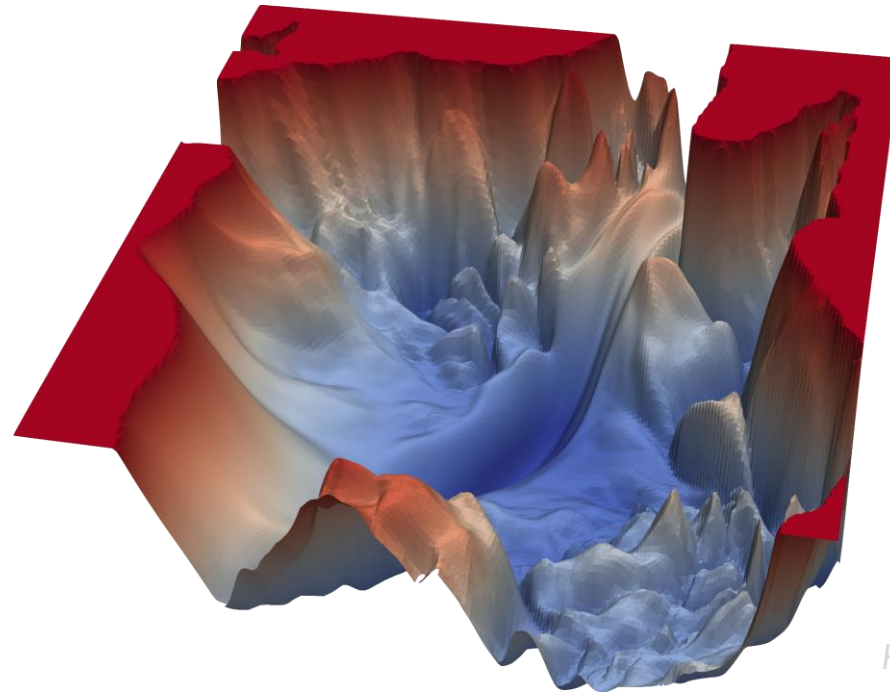


Imagen de <https://blog.paperspace.com/intro-to-optimization-in-deep-learning-gradient-descent/>

ENTRENAMIENTO: BACKPROPAGATION

Para una función de pérdida real con dos variables x e y ...



<https://www.cs.umd.edu/~tomg/projects/landscapes/>

ENTRENAMIENTO: BACKPROPAGATION

El tamaño del paso... importa

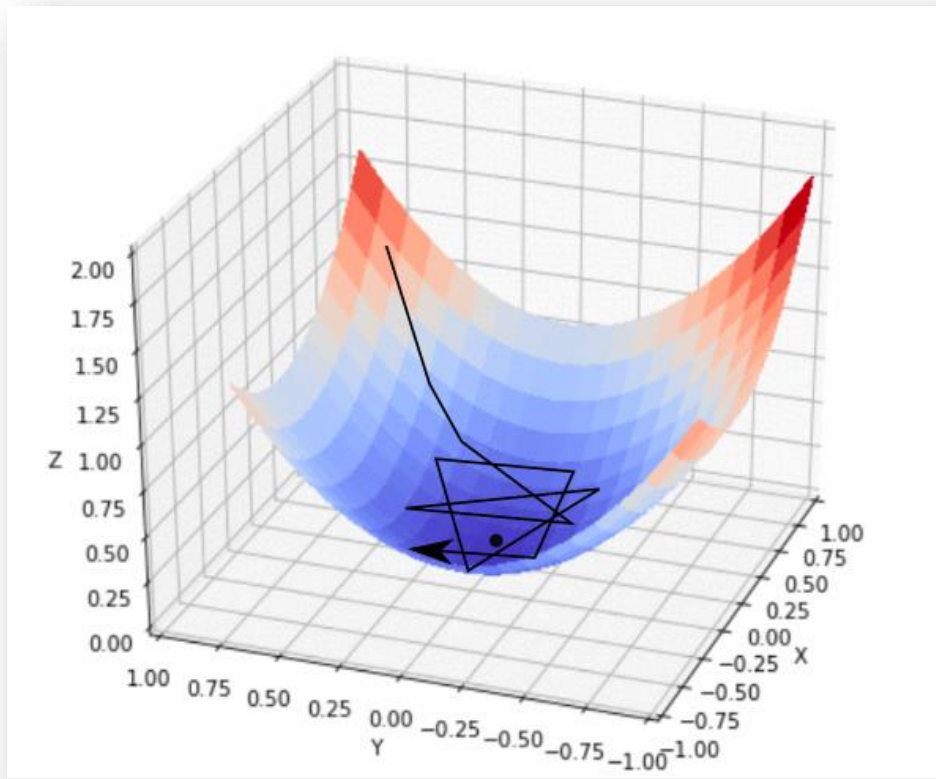


Imagen de <https://blog.paperspace.com/intro-to-optimization-in-deep-learning-gradient-descent/>

ENTRENAMIENTO: BACKPROPAGATION

El tamaño del paso: **Tasa de aprendizaje η**

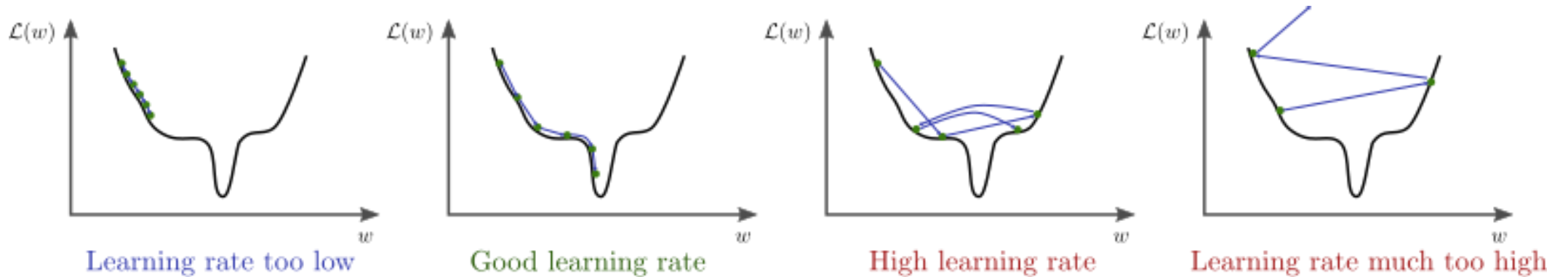
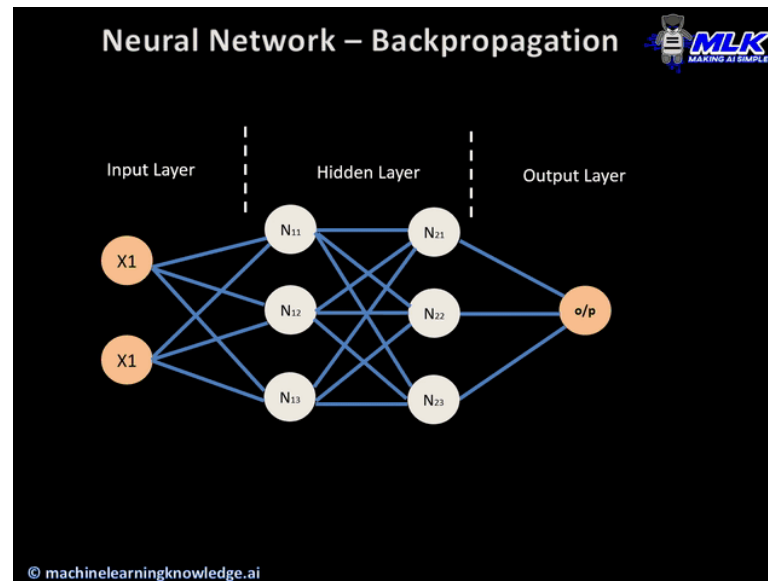


Imagen de <http://cs231n.github.io/neural-networks-3/>

ENTRENAMIENTO: BACKPROPAGATION

Técnica que permite el cálculo del gradiente de una función de costo que evalúa la RNA, a través de la propagación (hacia atrás) de los errores de los pesos de la red. En combinación con el descenso del gradiente permite que la red “aprenda” los pesos que optimizan la RNA.



ENTRENAMIENTO: BACKPROPAGATION

El algoritmo básico combinado con DG para m entradas

BACKPROPAGATION (m) :

Imagen de <http://neuralnetworksanddeeplearning.com/chap2.html>

1. Input a set of training examples

2. For each training example x : Set the corresponding input activation $a^{x,1}$, and perform the following steps:

- **Feedforward:** For each $l = 2, 3, \dots, L$ compute

$$z^{x,l} = w^l a^{x,l-1} + b^l \text{ and } a^{x,l} = \sigma(z^{x,l}).$$

- **Output error $\delta^{x,L}$:** Compute the vector

$$\delta^{x,L} = \nabla_a C_x \odot \sigma'(z^{x,L}).$$

- **Backpropagate the error:** For each

$l = L - 1, L - 2, \dots, 2$ compute

$$\delta^{x,l} = ((w^{l+1})^T \delta^{x,l+1}) \odot \sigma'(z^{x,l}).$$

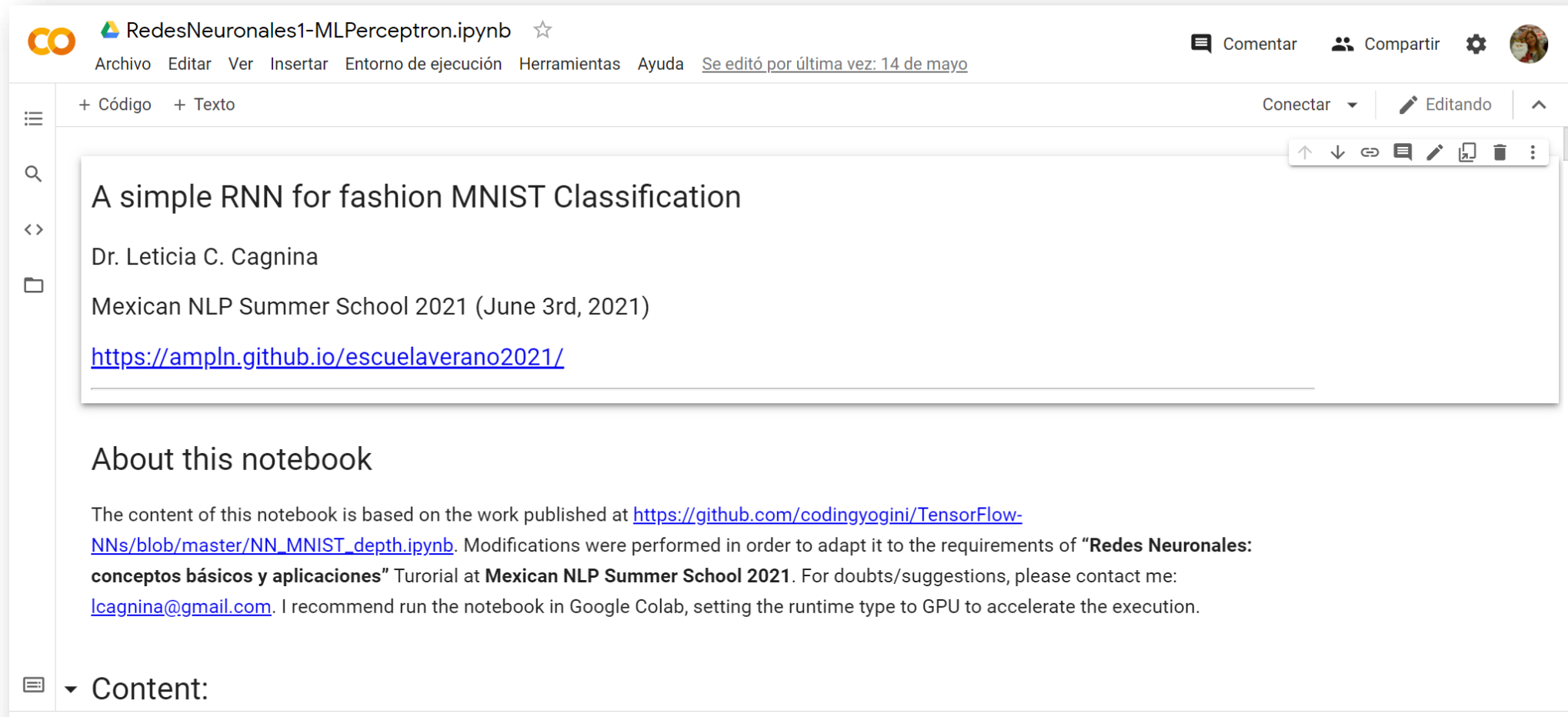
3. Gradient descent: For each $l = L, L - 1, \dots, 2$ update the weights according to the rule $w^l \rightarrow w^l - \frac{\eta}{m} \sum_x \delta^{x,l} (a^{x,l-1})^T$, and the biases according to the rule $b^l \rightarrow b^l - \frac{\eta}{m} \sum_x \delta^{x,l}$.

ENTORNO DE TRABAJO

- ❖ Lenguaje de programación: Python
- ❖ Posee innumerables bibliotecas para la carga y visualización de datos, cálculo de estadísticas, procesamiento de lenguaje natural, etc.
- ❖ Los cuadernos o notebooks de Jupyter permiten interactuar con el código
- ❖ Google Colab: buen número de paquetes instalados y permite la ejecución de las notebooks de Jupyter en forma remota (en la nube), incluso pudiendo elegir cpu, gpu o tpu.
- ❖ Material del curso: <https://github.com/lcagnina/Mexican-NLP-Summer-School-2021>



MLP PARA CLASIFICAR



The screenshot shows a Google Colab notebook interface. At the top, the title bar reads "RedesNeuronales1-MLPerceptron.ipynb" with a star icon. Below the title bar is a menu bar with options: Archivo, Editar, Ver, Insertar, Entorno de ejecución, Herramientas, Ayuda, and a link "Se editó por última vez: 14 de mayo". On the right side of the title bar are icons for "Comentar", "Compartir", a settings gear, and a user profile picture. Below the title bar is a toolbar with "+ Código" and "+ Texto" buttons. On the right side of the toolbar are "Conectar", "Editando", and a refresh icon. The main content area of the notebook contains the following text:

A simple RNN for fashion MNIST Classification

Dr. Leticia C. Cagnina

Mexican NLP Summer School 2021 (June 3rd, 2021)

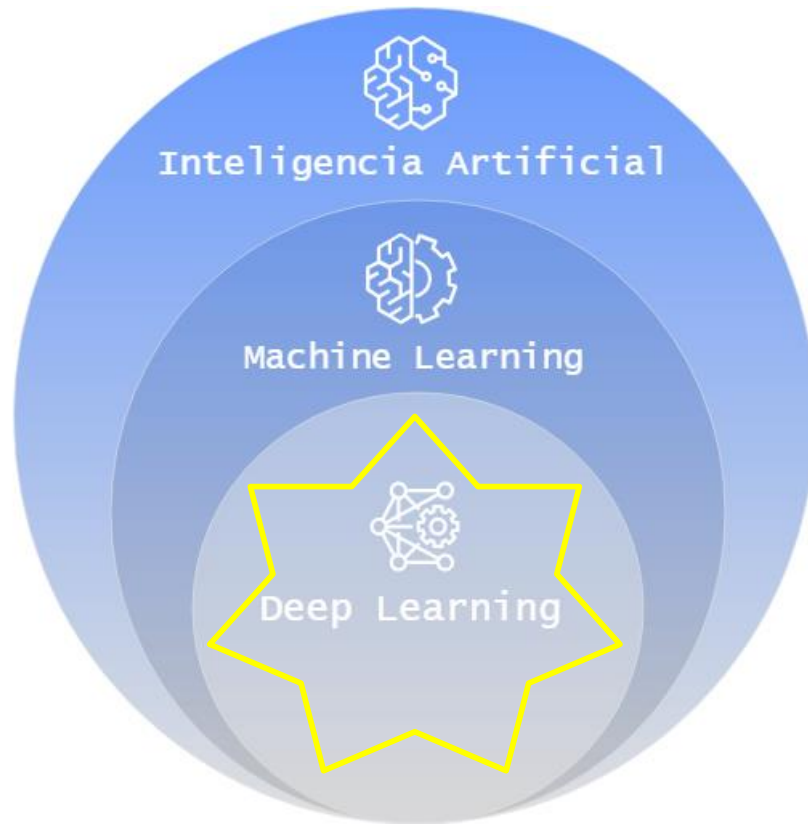
<https://ampln.github.io/escuelaverano2021/>

About this notebook

The content of this notebook is based on the work published at https://github.com/codingyogini/TensorFlow-NNs/blob/master/NN_MNIST_depth.ipynb. Modifications were performed in order to adapt it to the requirements of "Redes Neuronales: conceptos básicos y aplicaciones" Tutorial at Mexican NLP Summer School 2021. For doubts/suggestions, please contact me: lcagnina@gmail.com. I recommend run the notebook in Google Colab, setting the runtime type to GPU to accelerate the execution.

Content:

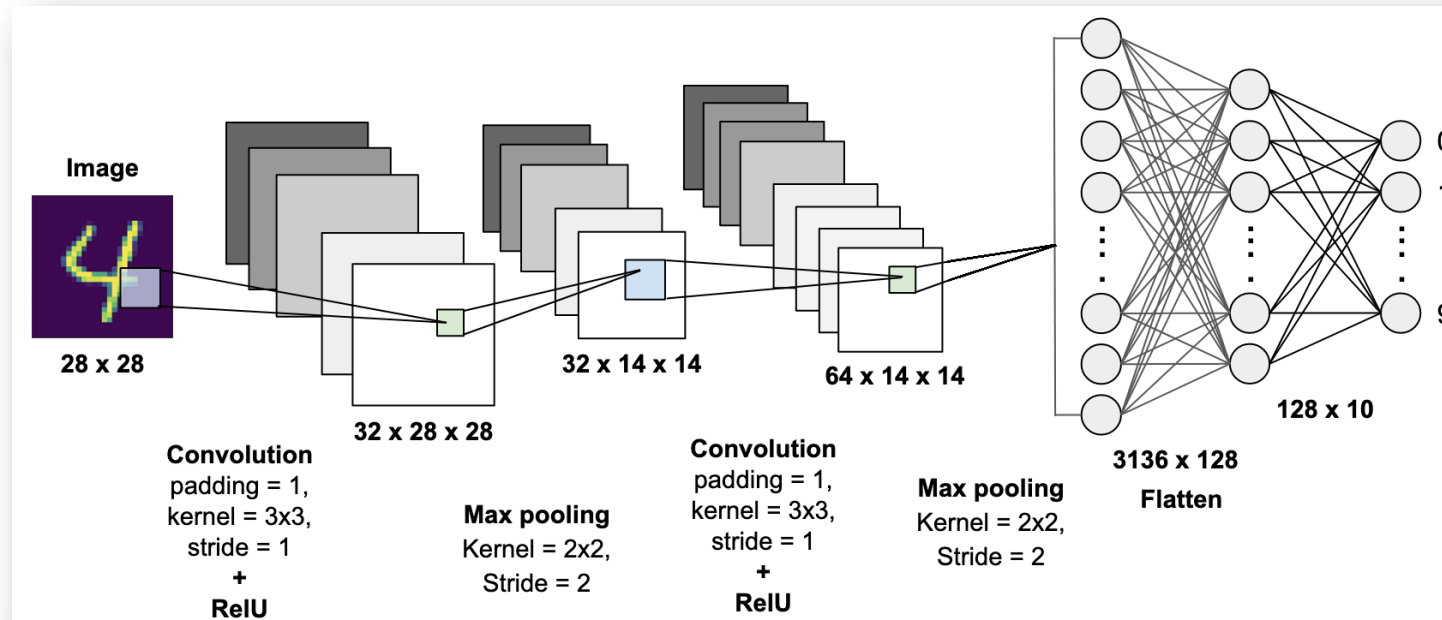
APRENDIZAJE PROFUNDO



“El aprendizaje profundo es la forma de aprender representaciones de datos en múltiples etapas. Es una idea simple, pero que resulta en mecanismos tan simples y suficientemente escalables que parece magia.” (Chollet, 2017)

APRENDIZAJE PROFUNDO

- ❖ Representaciones con complejidad incremental van obteniéndose capa a capa
- ❖ Las representaciones intermedias son aprendidas en conjunto



APRENDIZAJE PROFUNDO

Historia del *deep learning*

- ❖ En 2010 la comunidad científica trabajaba arduamente en redes neuronales: Geoffrey Hinton en la Universidad de Toronto, Yoshua Bengio en la Universidad de Montreal, Yann LeCun en Nueva York y el grupo de Universidad de Ciencias Aplicadas-Suiza (IDSIA).
- ❖ En 2011 Dan Ciresan (IDSIA) ganaba un premio en la competencia académica de clasificación de imágenes con una red neuronal profunda entrenada en GPU.
- ❖ En 2012 el grupo supervisado por Hinton alcanza una accuracy de $\sim 84\%$ con la AlexNet en el desafío ImageNet, clasificación de imágenes a gran escala (1.2 millones para entrenar y 150000 para testear, imágenes a color con alta resolución, 1000 clases diferentes).
- ❖ Desde 2012 las redes neuronales profundas convolucionales (*convnets*) es el método clásico en tareas que involucran visión por computadoras.

APRENDIZAJE PROFUNDO

Factores que posibilitaron *deep learning*

❖ Hardware:

- [1990-2010] las CPUs multiplicaron su factor de procesamiento en 5000.
- [2000] NVIDIA y AMD invirtieron billones de U\$ en desarrollar chips rápidos y masivamente paralelos (GPU) para potenciar la calidad de los gráficos y hacer más realistas los videos juegos.
- [2007] NVIDIA lanzó CUDA: interface de programación para las GPU. Esto permitió reemplazar clusters de CPUs por pocas GPUs para el procesamiento de aplicaciones altamente paralelizables.
- [2011] Dan Ciresan y Alex Krizhevsky implementan redes neuronales con CUDA.
- [2016] Google lanzó la unidad de procesamiento tensor (TPU): un circuito integrado diseñado para ejecutar redes neuronales profundas con una velocidad de al menos 10 veces superior a las más eficientes GPUs.

APRENDIZAJE PROFUNDO

Factores que posibilitaron *deep learning*

❖ Datos:

- Progreso en las fuentes de almacenamiento en los últimos 20 años.
- Internet: Fuente de información para recopilar y distribuir grandes volúmenes de datos: imágenes generadas por usuarios en Flickr, videos de YouTube, textos de Wikipedia...
- ImageNet: el primer dataset de imágenes etiquetadas.

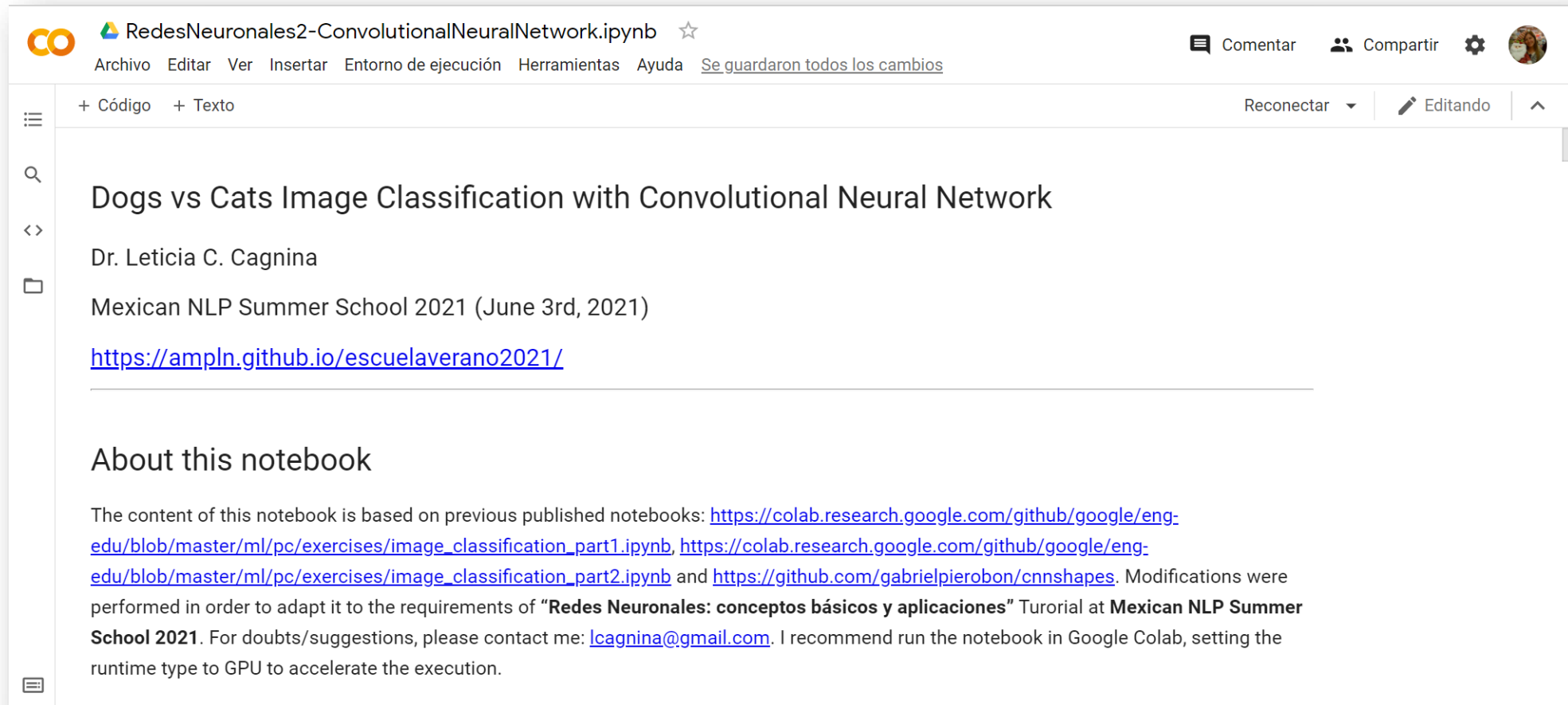
APRENDIZAJE PROFUNDO

Factores que posibilitaron *deep learning*

❖ Algoritmos:

- [2009-2010] se mejoró la propagación del gradiente entre muchas capas.
- Nuevas funciones de activación.
- Mejor inicialización de los pesos.
- Mejores esquemas de optimización (Adam y RMSProp).

APRENDIZAJE PROFUNDO



The screenshot shows a Google Colab interface for a notebook titled "RedesNeuronales2-ConvolutionalNeuralNetwork.ipynb". The top bar includes the Colab logo, the notebook title, and a star icon. Below this is a menu bar with options: Archivo, Editar, Ver, Insertar, Entorno de ejecución, Herramientas, Ayuda, and a link "Se guardaron todos los cambios". On the right side of the top bar are icons for "Comentar", "Compartir", settings, and a user profile picture.

The notebook content area has a left sidebar with icons for file explorer, search, and code execution. The main content area displays the title "Dogs vs Cats Image Classification with Convolutional Neural Network", the author "Dr. Leticia C. Cagnina", and the context "Mexican NLP Summer School 2021 (June 3rd, 2021)". A link to the GitHub repository is provided: <https://ampln.github.io/escuelaverano2021/>.

Below the link is a section titled "About this notebook". The text in this section states: "The content of this notebook is based on previous published notebooks: https://colab.research.google.com/github/google/eng-educ/blob/master/ml/pc/exercises/image_classification_part1.ipynb, https://colab.research.google.com/github/google/eng-educ/blob/master/ml/pc/exercises/image_classification_part2.ipynb and <https://github.com/gabrielpierobon/cnnshapes>. Modifications were performed in order to adapt it to the requirements of "Redes Neuronales: conceptos básicos y aplicaciones" Tutorial at Mexican NLP Summer School 2021. For doubts/suggestions, please contact me: lcagnina@gmail.com. I recommend run the notebook in Google Colab, setting the runtime type to GPU to accelerate the execution."

¿PARA QUÉ SIRVEN LAS RNA?

(Nelson-illingworth, 1991)

- ❖ Evaluador de Riego de Hipoteca: entrenado con más de 10000 solicitudes crediticias (aceptadas y rechazadas) el modelo identificaba patrones en solicitudes que podrían ser riesgosas. Fue utilizado por AVCO Financial Services (California) argumentado un 27% de éxito en la evaluación de solicitudes.
- ❖ Detector de bombas: utilizado en el aeropuerto internacional JFK de Nueva York, el detector examinaba hasta 10 bultos por minutos en busca de explosivos. La red neuronal fue entrenada con información sobre los materiales característicos en explosivos y a través de rayos gama podía identificar si había algo sospechoso en el bulto que examinaba. En casos indefinidos, el bulto era marcado para revision manual. La red podía aprender de componentes nuevos. Costó más de 1 millón de dólares.
- ❖ Revisión de ventiladores en motores de autos: en Siemens construían ventiladores de motores de autos Ford y para detectar problemas en esos, personas dedicadas al control de esas partes escuchaban el sonido que hacían. Con una RNA se aumentó en un 90% el éxito en detección de partes defectuosas.
- ❖ Reconocimiento de discurso: Intel desarrolló una aplicación que identificaba palabras (unas cientos) o frases para que pueda usarse en la industria productiva y así facilitar la grabación de inspecciones de productos con el habla (que se procesaba y grababa en el sistema).

¿PARA QUÉ SIRVEN LAS RNA?

En la actualidad...

- ❖ Industria automotriz: control de inyectores de combustible, Sistema automático de frenos, detección de problemas en el sistema de arranque, etc.
- ❖ Defensa: trayectoria de armas de guerra, seguimiento de objetivos, reconocimiento de objetos, reconocimiento facial, nuevos tipos de sensores, sonar, radar, procesamiento de imagen, compresión de datos, extracción de información y supresión de ruido, identificación de señales/imágenes, detección de fraude, etc.
- ❖ Electrónica: predicción de secuencia de códigos, diseño de chips de circuitos integrados, control de procesos, análisis de fallas de chips, visión artificial, síntesis de voz, etc.
- ❖ Entretenimiento: animaciones, efectos especiales, video juegos, sistemas recomendadores, etc.
- ❖ Manufactura: control de procesos de fabricación, diseño y análisis de productos, diagnóstico de procesos y máquinas, sistemas de inspección de calidad visual, análisis de calidad de soldaduras, predicción de calidad de papel, análisis de calidad de chips de computadora, análisis de operaciones de molienda, análisis de calidad de cervezas, planificación y gestión de proyectos, etc.

¿PARA QUÉ SIRVEN LAS RNA?

En la actualidad...

- ❖ Medicina: análisis de células cancerígenas, diseño de prótesis, optimización de tiempos de trasplantes, mejoramiento de calidad de atención hospitalaria, sistemas de diagnóstico de enfermedades complejas, análisis de imágenes, etc.
- ❖ Robótica: control de trayectoria, sistemas de visión, vehículos autónomos, etc.
- ❖ Telecomunicaciones: compresión de datos e imágenes, servicios de información automáticos, traducción del habla en tiempo real, sistemas de procesamiento de pagos, etc.
- ❖ Discurso: reconocimiento de discurso, compresión de lenguaje hablado, clasificación de palabras, conversión de texto a discurso hablado y viceversa, etc.
- ❖ Generación de texto....

BIBLIOGRAFÍA CONSULTADA

- ❖ Neural Networks -a comprehensive Foundation-. Simon Haykin. IEEE PRESS. 1994.
- ❖ Neural networks and learning machines. Simon Haykin. 3rd edition. PEARSON. 2009.
- ❖ Deep Learning with Python. François Chollet. MANNING. 2018.
- ❖ Neural Networks and Deep Learning. Michael A. Nielsen. DETERMINATION PRESS. 2015. Disponible en línea (accedido Mayo 2021): <http://neuralnetworksanddeeplearning.com/>

PARA FINALIZAR...

iii Muchas Gracias!!!

¿dudas, comentarios?

lcagnina@gmail.com

<https://github.com/lcagnina/Mexican-NLP-Summer-School-2021>

<https://sites.google.com/view/leticia-cagnina/home>