

Introducción al Aprendizaje Profundo

Aprendizaje de Máquina Aplicado

Yomin Jaramillo Munera
yejaramilm@eafit.edu.co

2025

Juan David Martínez Vargas
jdmartinev@eafit.edu.co

Agenda

- Motivación
- Redes neuronales artificiales
- Redes neuronales profundas
- Ejemplos prácticos



Motivación

Motivación

Supongamos que tenemos un set de datos con 100 características y queremos aplicar regresión logística, pero tenemos suficientes datos para permitir que el modelo tenga alta varianza, por lo que decidimos incluir características polinómicas.

	grado ≤ 1	grado ≤ 2	grado ≤ 3	grado ≤ 4	grado ≤ 5
# de caracterísitcas	101	10.101	1.010.101	101.010.101	10.101.010.101

Motivación

¿Cómo podríamos generar nuevas características de manera más inteligente y eficiente?

¿Qué tal si utilizamos Machine Learning para generar las características?

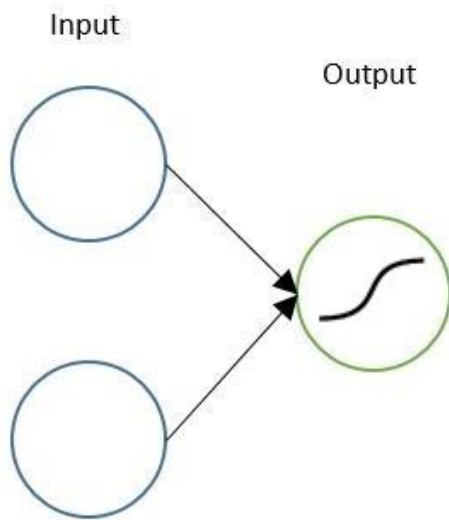




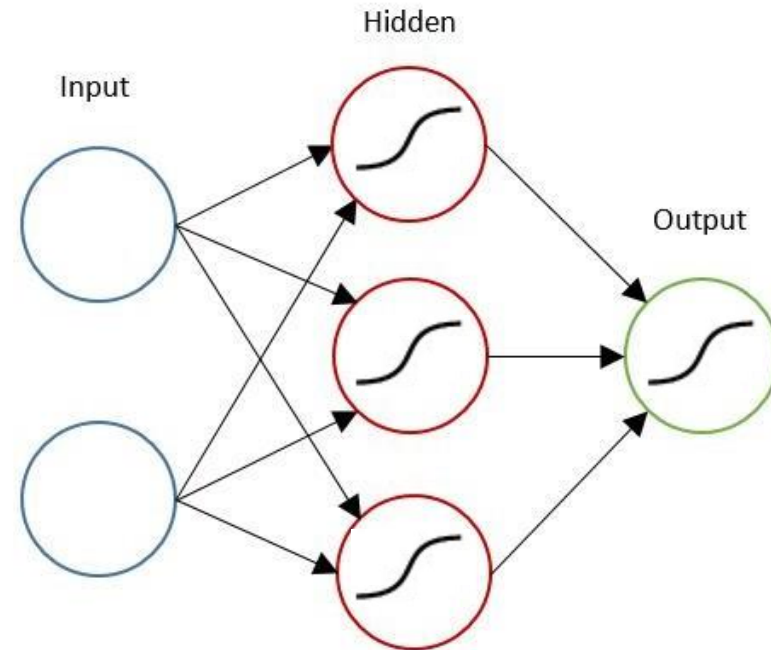
Redes Neuronales Artificiales

Redes Neuronales Artificiales (ANN)

Logistic Regression

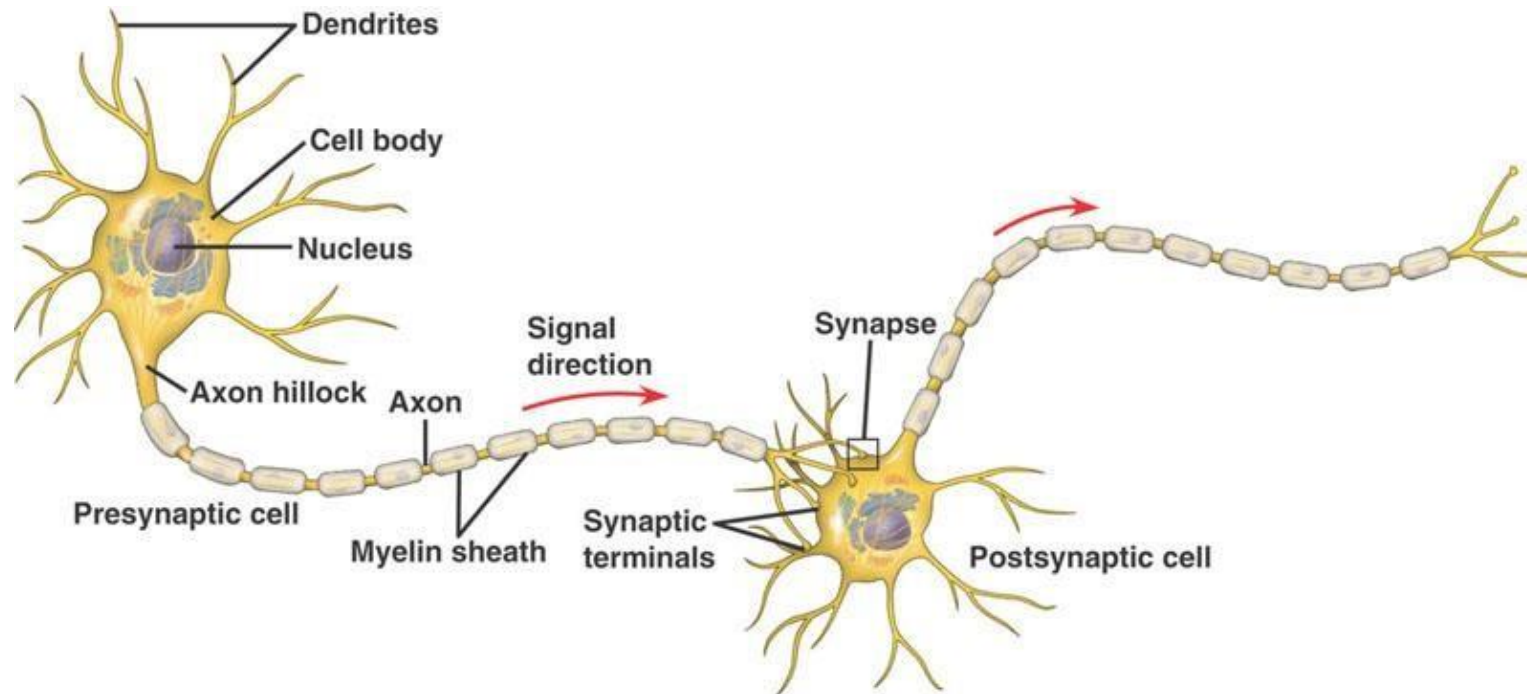


Artificial Neural Network



ANN: ¿Por Qué Reciben ese Nombre?

La conexión entre nodos de regresión logística está inspirada en las conexiones neuronales presentes en el cerebro.

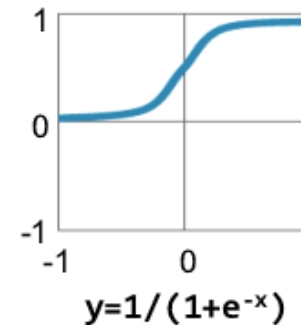


ANN: Funciones de Activación

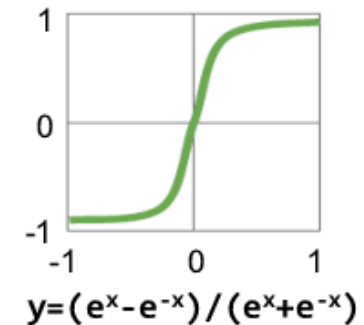
La función logística no es la única función de activación utilizada en las neuronas de una ANN.

Traditional
Non-Linear
Activation
Functions

Sigmoid

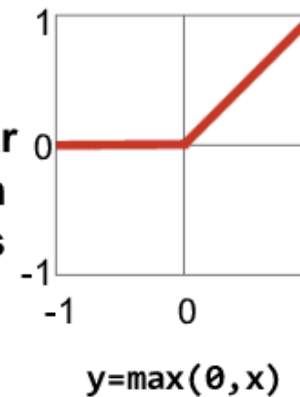


Hyperbolic Tangent

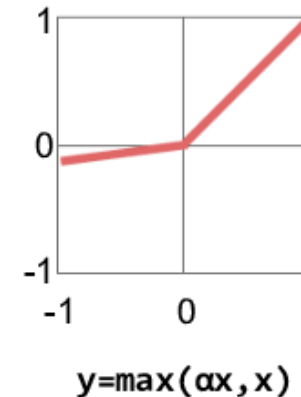


Modern
Non-Linear
Activation
Functions

Rectified Linear Unit
(ReLU)

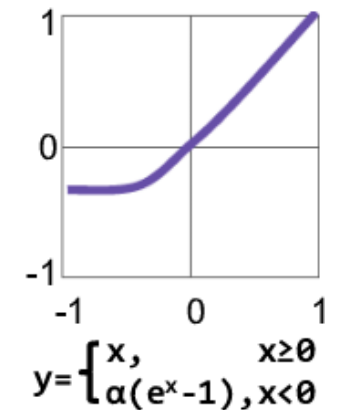


Leaky ReLU



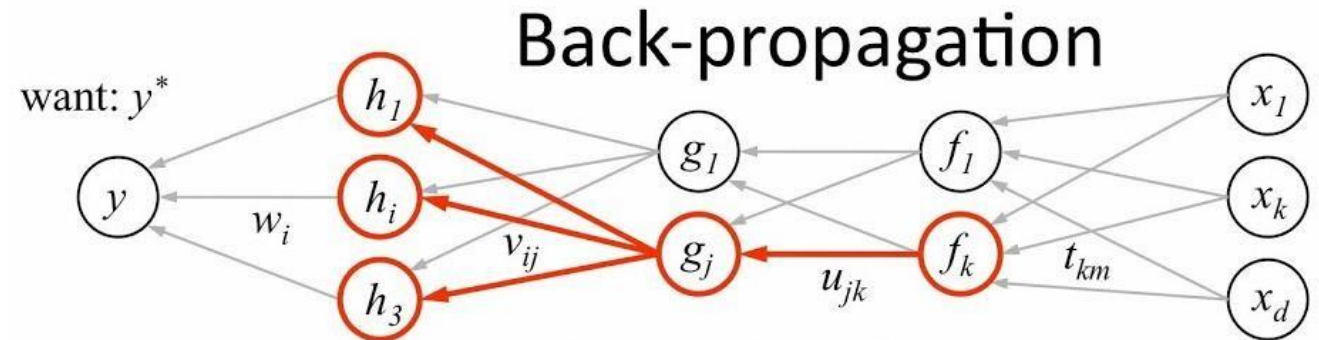
$\alpha = \text{small const. (e.g. 0.1)}$

Exponential LU



ANN: Entrenamiento

Todos los parámetros de una ANN se entrenan al tiempo mediante descenso por el gradiente. El gradiente se calcula por medio de un algoritmo llamado “backpropagation”.



1. receive new observation $\mathbf{x} = [x_1 \dots x_d]$ and target y^*
2. **feed forward:** for each unit g_j in each layer $1 \dots L$
compute g_j based on units f_k from previous layer: $g_j = \sigma \left(u_{j0} + \sum_k u_{jk} f_k \right)$
3. get prediction y and error $(y - y^*)$
4. **back-propagate error:** for each unit g_j in each layer $L \dots 1$

(a) compute error on g_j

$$\underbrace{\frac{\partial E}{\partial g_j}}_{\text{should } g_j \text{ be higher or lower?}} = \sum_i \underbrace{\sigma'(h_i)}_{\text{how } h_i \text{ will change as } g_j \text{ changes}} \underbrace{v_{ij}}_{\text{was } h_i \text{ too high or too low?}} \underbrace{\frac{\partial E}{\partial h_i}}_{\text{was } h_i \text{ too high or too low?}}$$

(b) for each u_{jk} that affects g_j

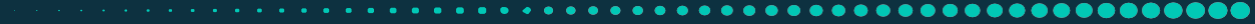
(i) compute error on u_{jk}

$$\frac{\partial E}{\partial u_{jk}} = \underbrace{\frac{\partial E}{\partial g_j}}_{\text{do we want } g_j \text{ to be higher/lower?}} \underbrace{\sigma'(g_j) f_k}_{\text{how } g_j \text{ will change if } u_{jk} \text{ is higher/lower?}}$$

(ii) update the weight

$$u_{jk} \leftarrow u_{jk} - \eta \frac{\partial E}{\partial u_{jk}}$$

Copyright © 2014 Victor Lavrenko

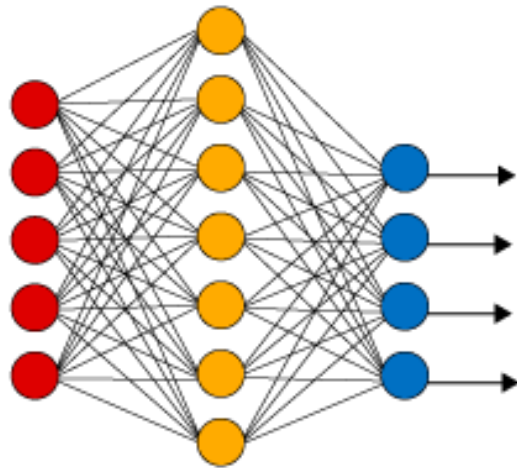


Redes Neuronales Profundas

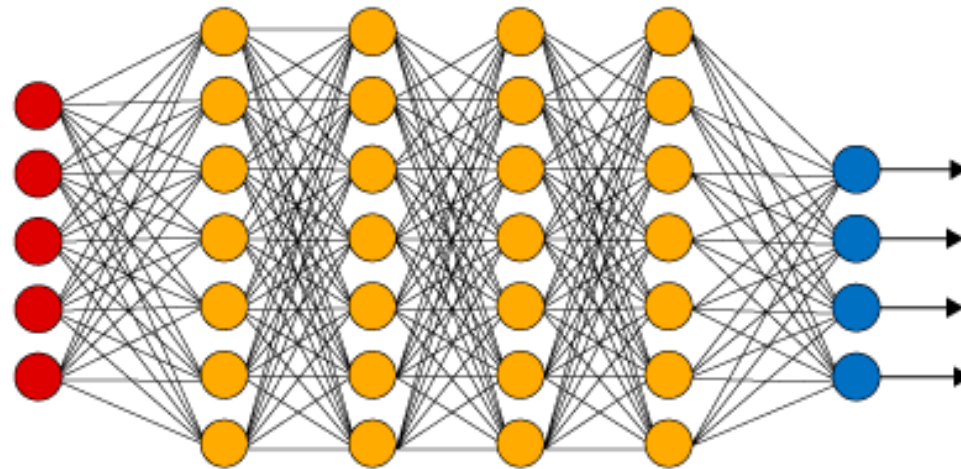
Redes Neuronales Profundas (DNN)

Podemos generar características aún más especializadas y eficientes si aumentamos el número de capas ocultas (suponiendo que tenemos suficiente necesidad de varianza para que esto sea útil).

Simple Neural Network



Deep Learning Neural Network



● Input Layer

● Hidden Layer

● Output Layer

Aprendizaje Profundo

Las dos librerías más utilizadas a la hora de construir y entrenar redes neuronales son TensorFlow (de Google) y PyTorch (de Facebook).



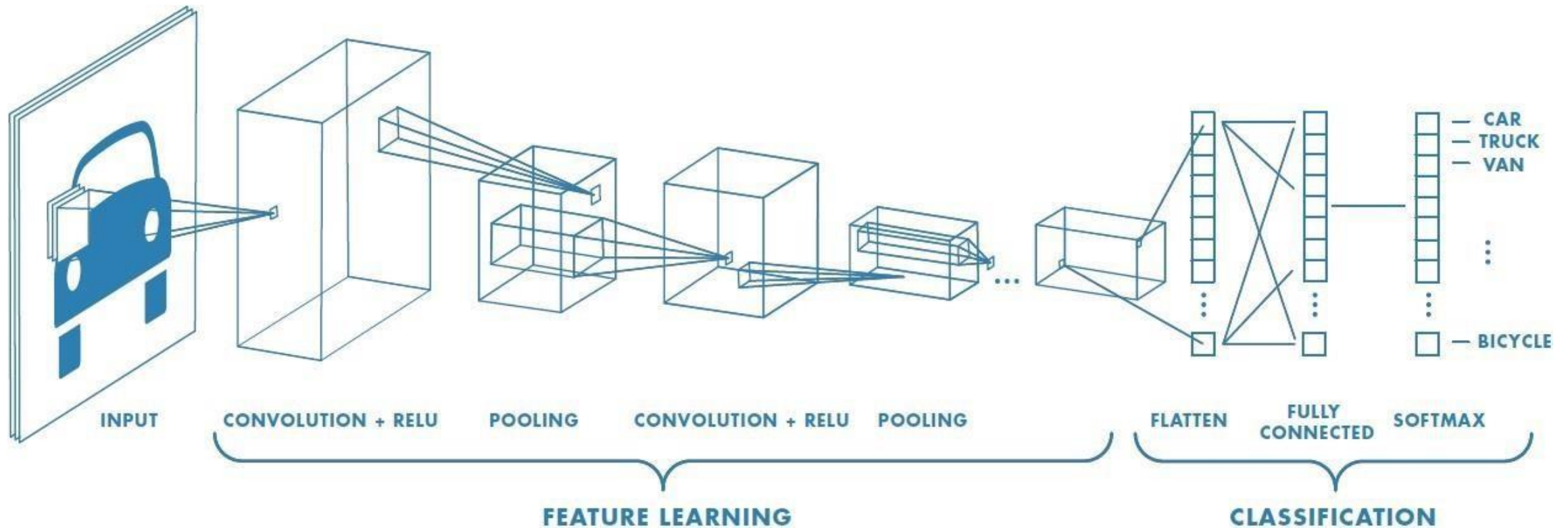
TensorFlow



PyTorch

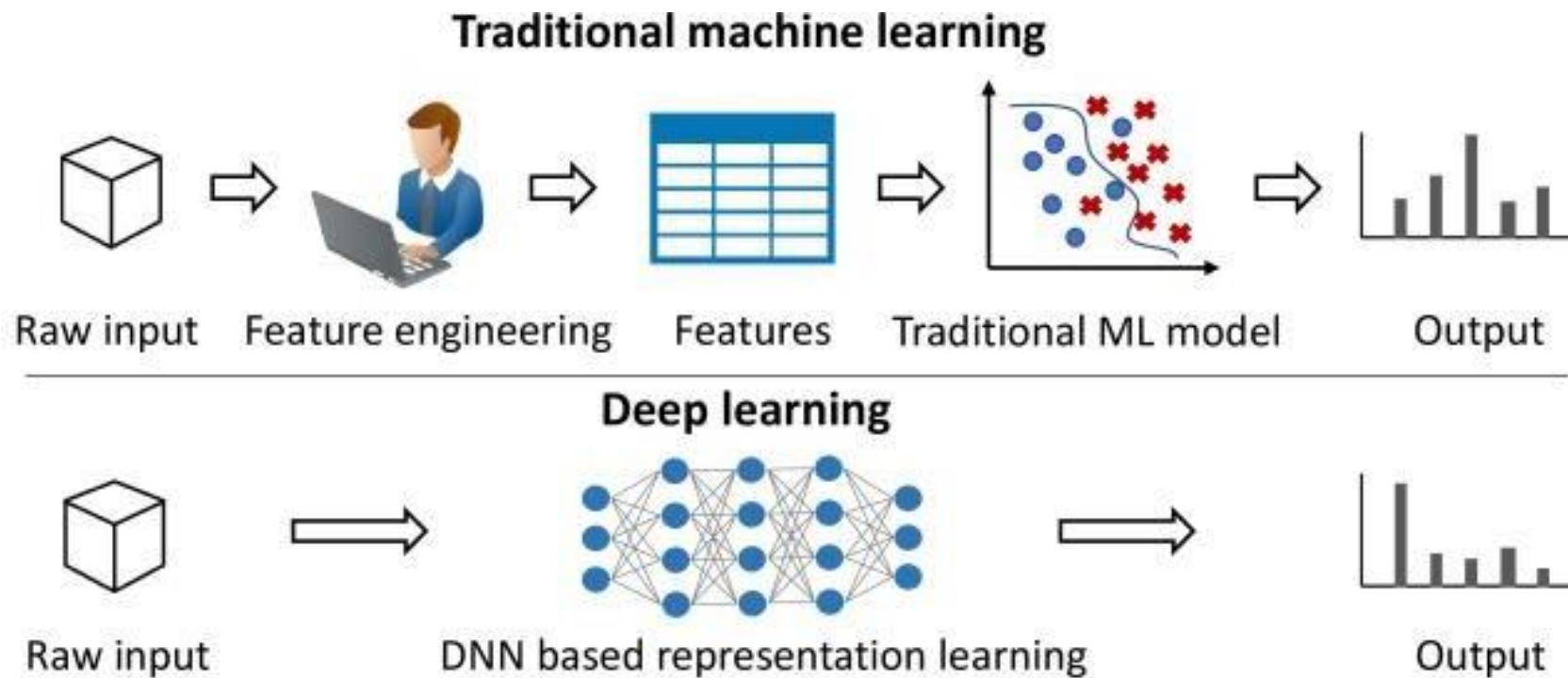
Aprendizaje Profundo: Usos

Visión por computadora:



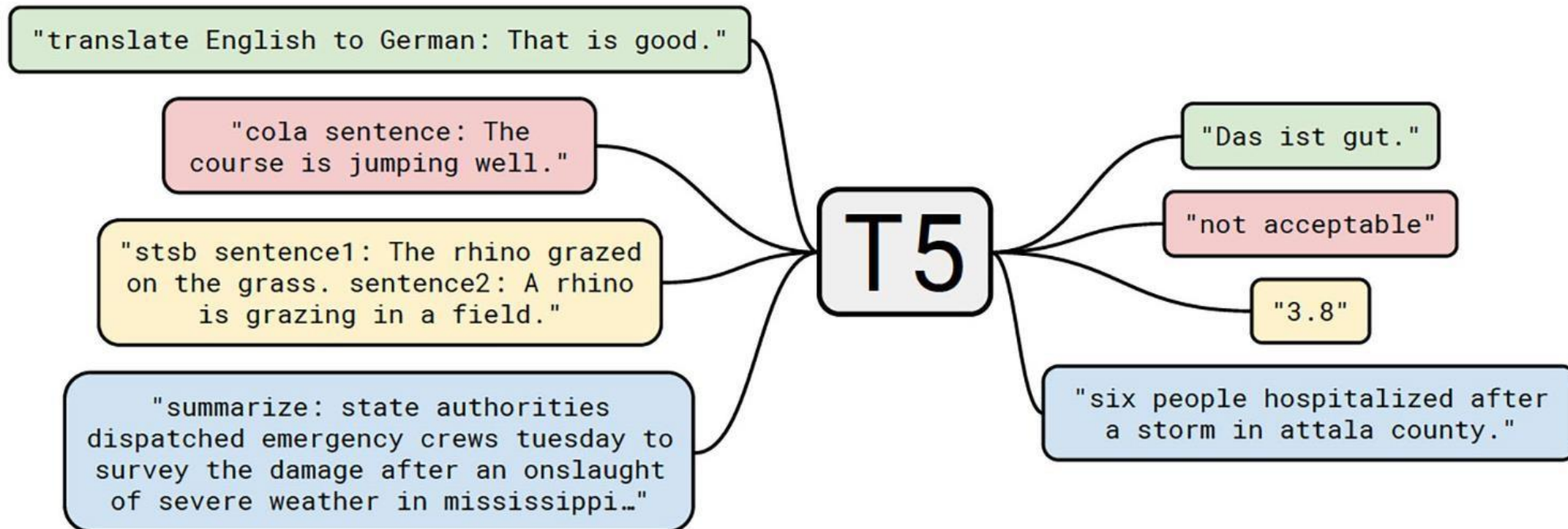
Aprendizaje Profundo

Cuando utilizamos redes neuronales profundas estamos en un subdominio del machine learning denominado aprendizaje profundo (deep learning).



Aprendizaje Profundo: Usos

Procesamiento de lenguaje natural:





Ejemplos de aprendizaje profundo



¡Muchas Gracias!