Redes Neuronales y Deep learning

M.C. JUAN EQUIHUA

Agenda

	Calentamieto
1:20	Multi-Layer Perceptron y FNN
	Break (15min)
	NLP basics
	Demo 1 – Sentiment Analysis
	Computer vision y deep learning
1:00	Demo 2 – Clasificando imagenes
	Break (15min)
	RNN
	Arquictureas avanzadas en NN
1:20	Demo 3 – AE & LSTM
	Resúmen y Recursos adicionales

Juan Equihua

Lic. Actuaría

Sr Data Analyst

MSc Data Science

PhD Data Science & AI

2014

2015

2016

2018

2019

2020

...

Actuarial Analyst

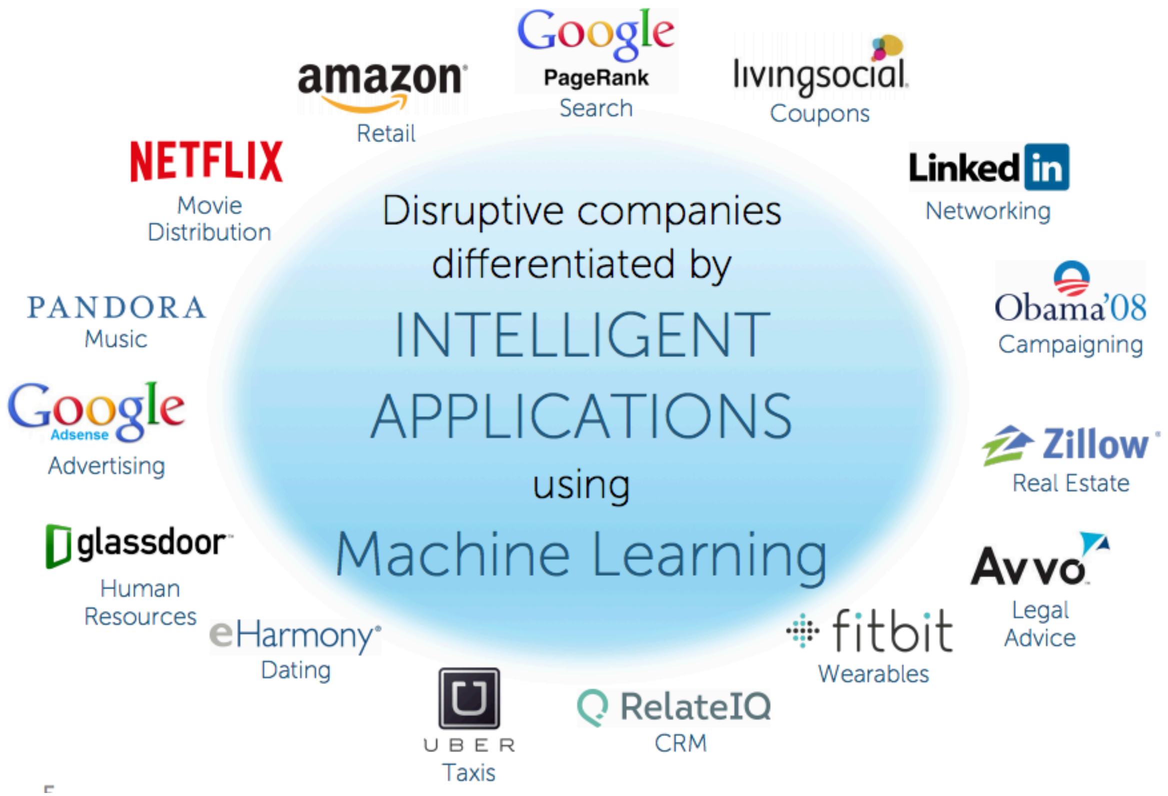
Especialista desarrollo de modelos

Research Data Scientist

profusion

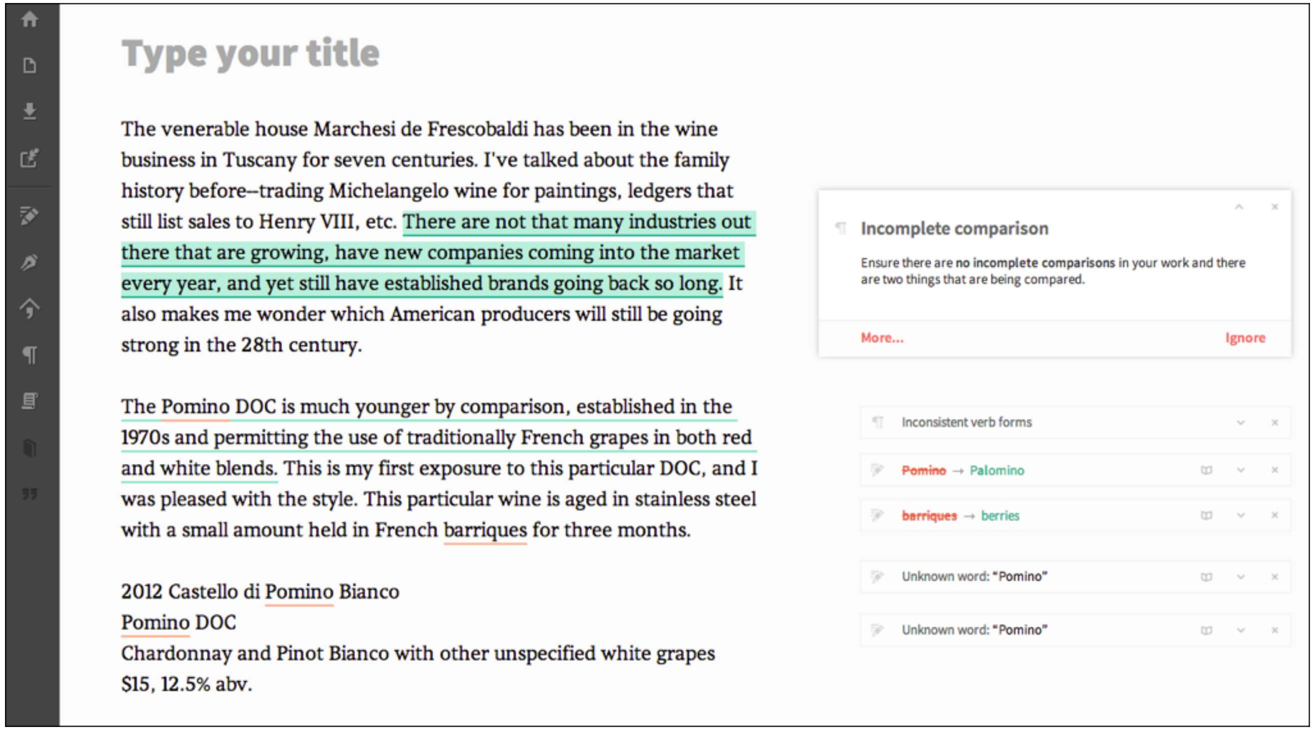
University of Essex

Machine Learning está en todos lados

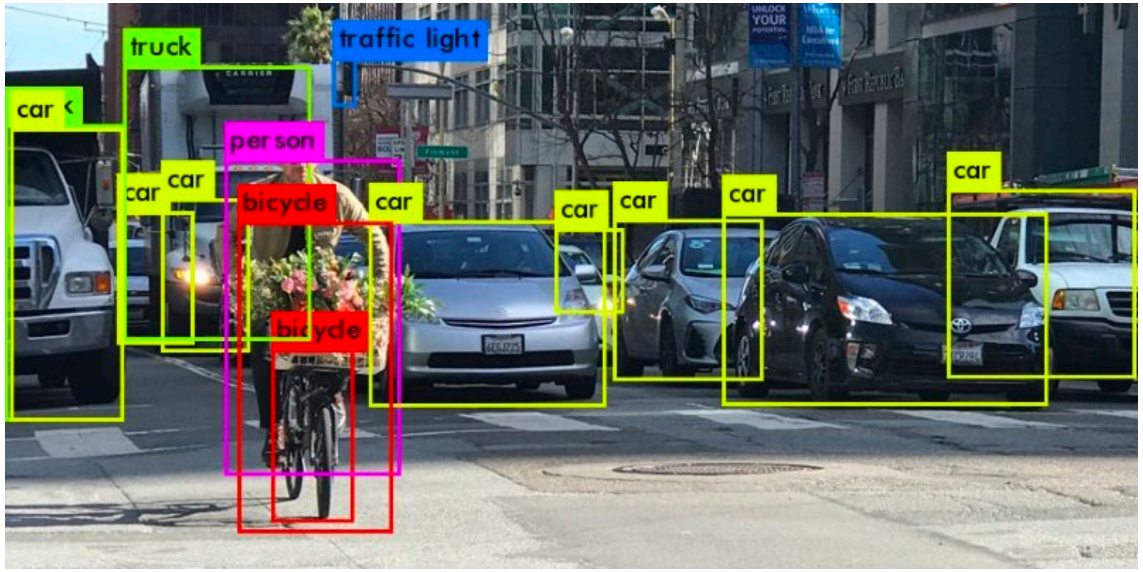


Redes Neuronales en la vida real

Grammarly at correcting sentences

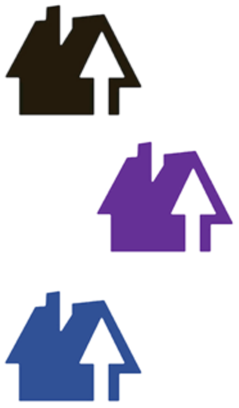


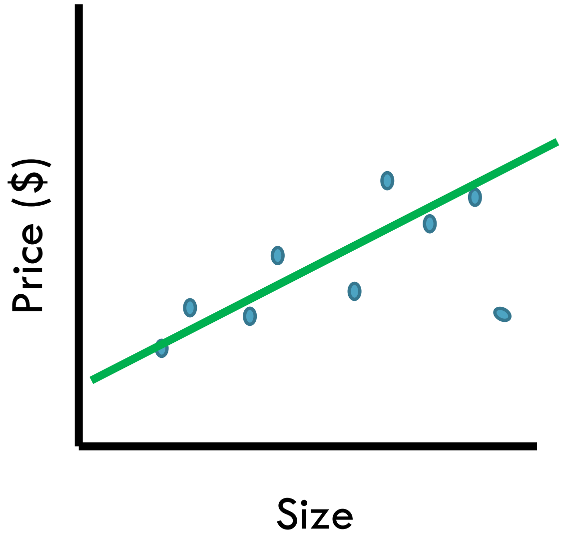
Object recognition for self-driving cars



Ejemplo 1. Prediciendo precios de casas

Datos | **Entrenamiento** | **Predicción** | **Objetivo**





Predecir para nuevas observaciones

Precio?

Encontrar los pesos *w* para la regression linear. Como? minimizando el una función de Perdida.

$$d(x) = \mathbf{w_0} + \mathbf{w_1} x_1 + \mathbf{w_2} x_2$$

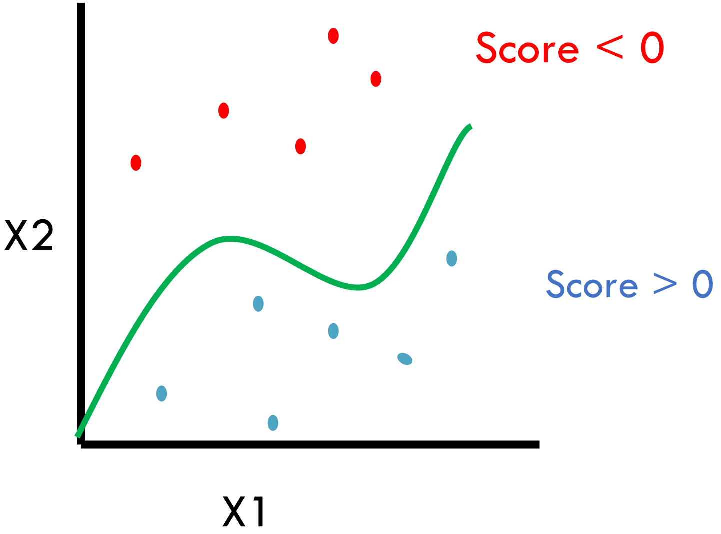
$$L(w) = \frac{1}{\ell} \|Xw - y\|^2$$

Metricas comunes:

$$\mathrm{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2}$$

Ejemplo 2. Prediciendo sentimientos en reviews

Datos	Entrenamiento	Predicción	Objetivo



Encontrar los pesos w para el límite de decisión

$$a(x) = \text{sign}(w^T x)$$

Via maximizando el "Classification Accuracy"

$$\frac{1}{\ell}\sum_{i=1}^{\ell}[a(x_i) = y_i]$$

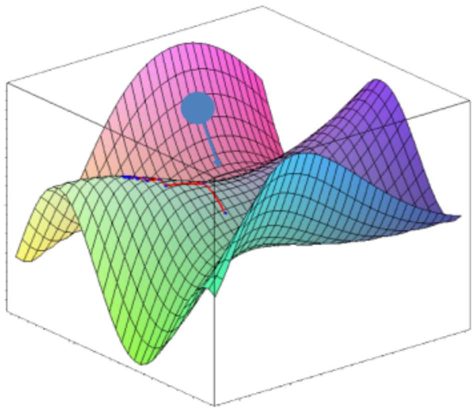
El único problema es que ésta función es no diferenciable, y dificil de maximizar.

Como encontrar *w*?

Stochastic Gradient Decent

SGD es un algorítmo para minimizar cualquier function de pérdida diferenciable.

Optimization problem L(w) = min(w)



w^0 — initialization

$$\nabla L(w^0) = \left(\frac{\partial L(w^0)}{\partial w_1}, \dots, \frac{\partial L(w^0)}{\partial w_n}\right) \text{— gradient vector}$$

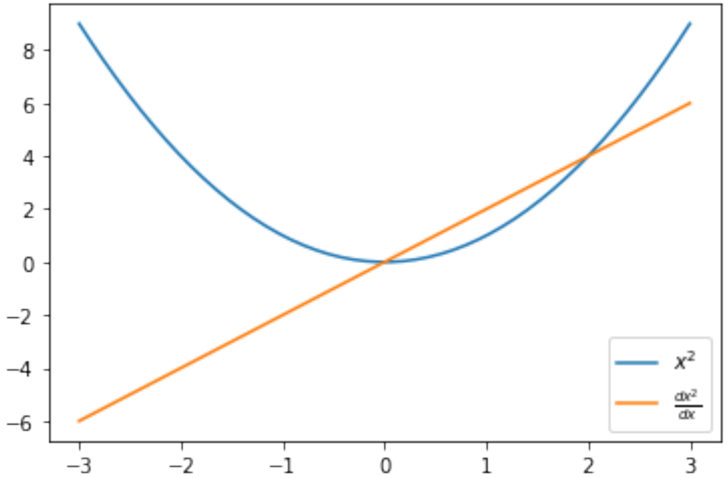
$$w^1 = w^0 - \eta_1 \nabla L(w^0) \text{— gradient step}$$

while True:

$$w^t = w^{t-1} - \eta_t \nabla L(w^{t-1})$$

if $\|w^t - w^{t-1}\| < \epsilon$ then break

1-Dimension SGD



$$\nabla L(w^0) = \left(\frac{\partial L(w^0)}{\partial w_1}, \dots, \frac{\partial L(w^0)}{\partial w_n}\right) \text{ — gradient vector}$$

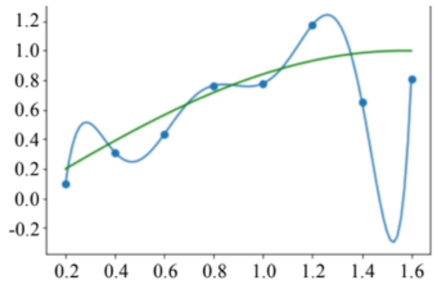
$$w^1 = w^0 - \eta_1 \nabla L(w^0) \text{ — gradient step}$$

L1 y L2 Losses

L1 Penalty

$$L_{reg}(w) = L(w) + \lambda \|w\|_1 \rightarrow \min_w$$

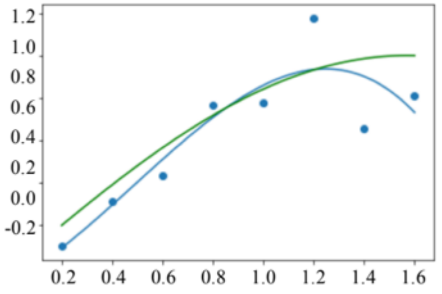
- $\|w\|_1 = \sum_{j=1}^{d} |w_j|$
- Drives some weights **exactly** to zero
- Learns sparse models
- Cannot be optimized with simple gradient methods



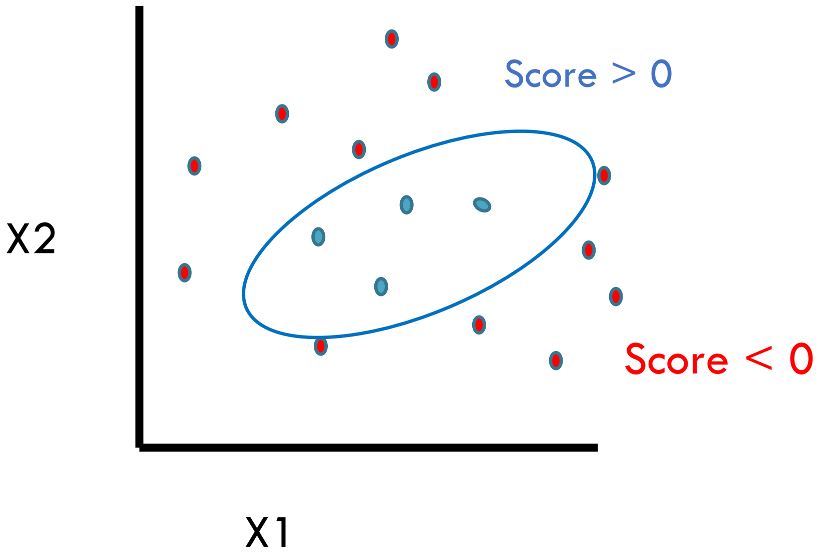
L2 Penalty

$$L_{reg}(w) = L(w) + \lambda \|w\|^2 \rightarrow \min_w$$

- $\|w\|^2 = \sum_{j=1}^{d} w_j^2$
- Drives all weights **closer** to zero
- Can be optimized with gradient methods



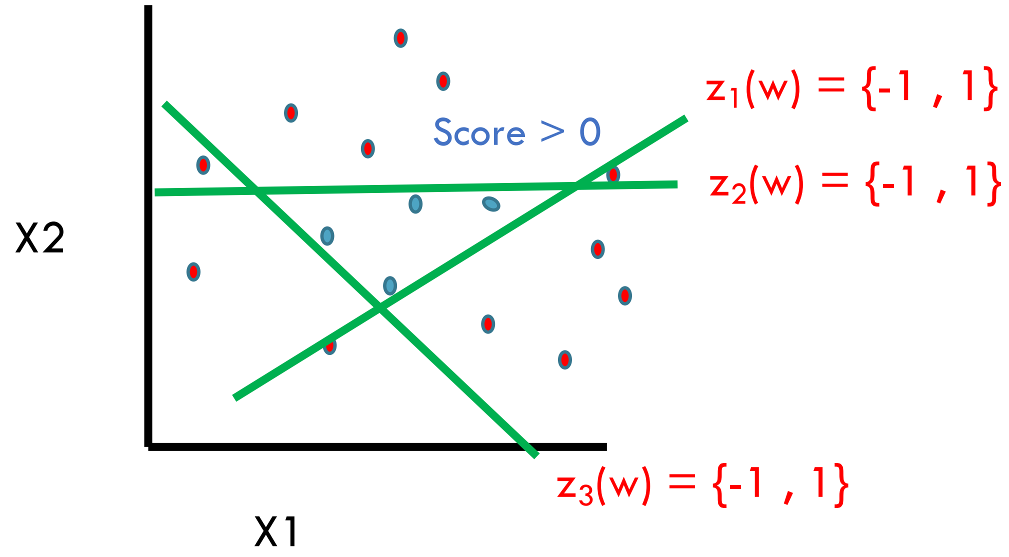
Como resolver un problema no linear?



No existe un único limite de decision (decision boundary) que separe las dos diferentes clases

Regressiones aggregadas

Podemos resolver éste problema mediante differentes 3 regresiones lineales y después una regresion logística.

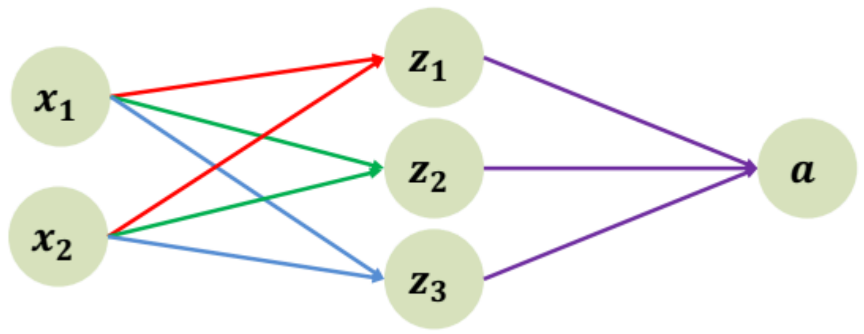


$$Y = b_0 + B_1 Z_1 + b_2 Z_2 + b_3 Z_3$$

This model depends of X and Z

La primer red neuronal: MLP

- Sabemos la forma de nuestro clasificador
  - $z_i = \sigma(\boldsymbol{w}_{0,i} + \boldsymbol{w}_{1,i}x_1 + \boldsymbol{w}_{2,i}x_2)$
  - $a(x) = \sigma(\boldsymbol{w}_0 + \boldsymbol{w}_1 z_1(x) + \boldsymbol{w}_2 z_2(x) + \boldsymbol{w}_3 z_3(x))$
- Ahora podemos reescribir nuestro clasificador como gráfica computacional



Nodes: computed variables $(\boldsymbol{x_1}, \boldsymbol{x_2}, \boldsymbol{z_1}, \boldsymbol{z_2}, \boldsymbol{z_3}, \boldsymbol{a})$
Edges: dependencies (we need $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$ to compute $\boldsymbol{z_1}$)

Esta gráfica tiene el nombre de Multi-Layer Perceptron y es la red neuronal más simple:

- Input Layer (X_i)
- Hidden layer (Z_i)
- Output Layer (a)

Cada nodo es una regression (o neurona) con su propia function de activación

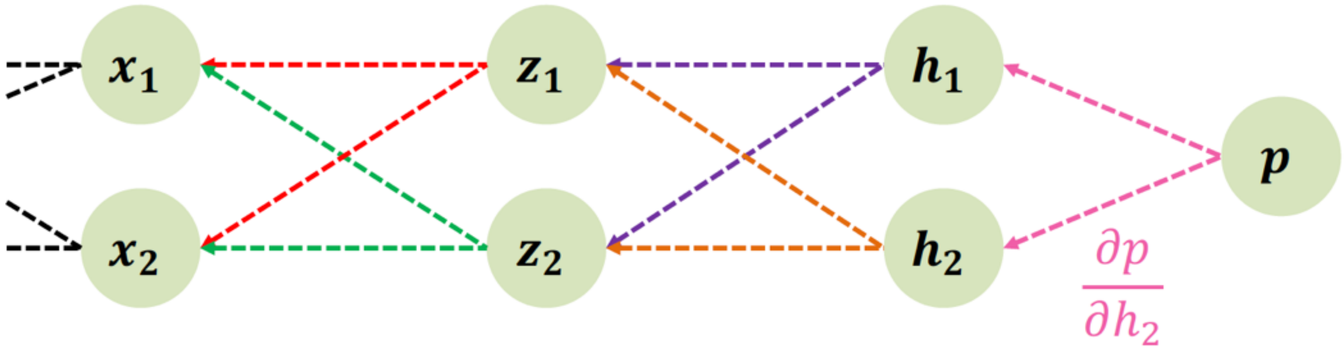
Back - Propagation

Como Encontrar w en un MLP?

$$h_2 = \sigma(\boldsymbol{w_0} + \boldsymbol{w_1} z_1 + \boldsymbol{w_2} z_2)$$

Gradient Descent:

$$\frac{\partial L}{\partial w_1} = \frac{\partial L}{\partial p}\frac{\partial p}{\partial w_1} = \frac{\partial L}{\partial p}\frac{\partial p}{\partial h_2}\frac{\partial h_2}{\partial w_1}$$



3: $\frac{\partial p}{\partial h_1}$ $\frac{\partial p}{\partial h_2}$

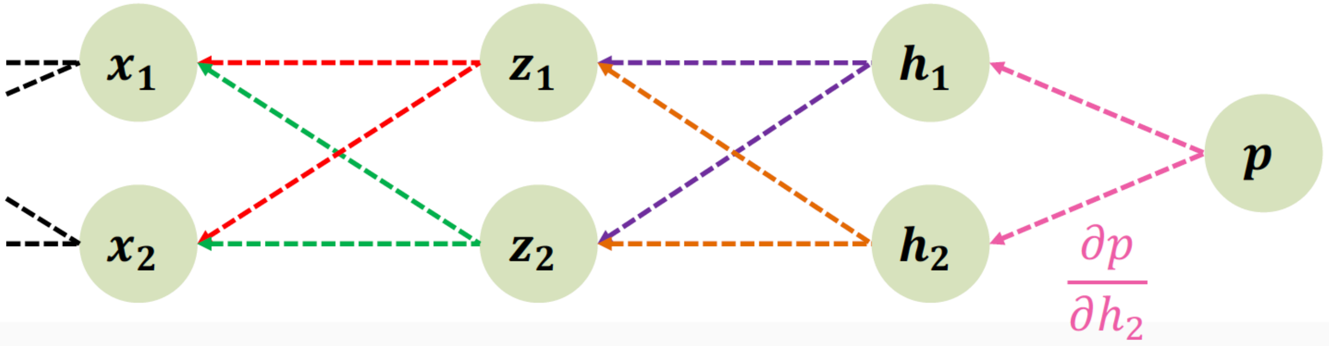
We will need these for GD

2: $\frac{\partial p}{\partial z_1} = \frac{\partial p}{\partial h_1}\frac{\partial h_1}{\partial z_1} + \frac{\partial p}{\partial h_2}\frac{\partial h_2}{\partial z_1}$ $\frac{\partial p}{\partial z_2} = \frac{\partial p}{\partial h_1}\frac{\partial h_1}{\partial z_2} + \frac{\partial p}{\partial h_2}\frac{\partial h_2}{\partial z_2}$

1:

$$\frac{\partial p}{\partial x_1} = \frac{\partial p}{\partial h_1}\frac{\partial h_1}{\partial z_1}\frac{\partial z_1}{\partial x_1} + \frac{\partial p}{\partial h_2}\frac{\partial h_2}{\partial z_1}\frac{\partial z_1}{\partial x_1} + \frac{\partial p}{\partial h_1}\frac{\partial h_1}{\partial z_2}\frac{\partial z_2}{\partial x_1} + \frac{\partial p}{\partial h_2}\frac{\partial h_2}{\partial z_2}\frac{\partial z_2}{\partial x_1}$$

$$\frac{\partial p}{\partial x_2} = \frac{\partial p}{\partial h_1}\frac{\partial h_1}{\partial z_1}\frac{\partial z_1}{\partial x_2} + \frac{\partial p}{\partial h_2}\frac{\partial h_2}{\partial z_1}\frac{\partial z_1}{\partial x_2} + \frac{\partial p}{\partial h_1}\frac{\partial h_1}{\partial z_2}\frac{\partial z_2}{\partial x_2} + \frac{\partial p}{\partial h_2}\frac{\partial h_2}{\partial z_2}\frac{\partial z_2}{\partial x_2}$$



3: $\frac{\partial p}{\partial h_1}$ $\frac{\partial p}{\partial h_2}$

We will need these for GD

2: $$\frac{\partial p}{\partial z_1} = \frac{\partial p}{\partial h_1}\frac{\partial h_1}{\partial z_1} + \frac{\partial p}{\partial h_2}\frac{\partial h_2}{\partial z_1} \qquad \frac{\partial p}{\partial z_2} = \frac{\partial p}{\partial h_1}\frac{\partial h_1}{\partial z_2} + \frac{\partial p}{\partial h_2}\frac{\partial h_2}{\partial z_2}$$

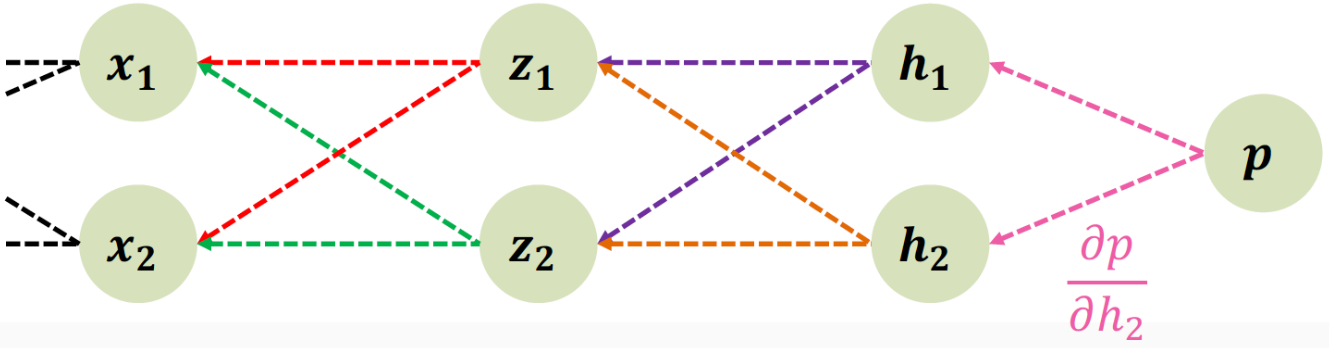
Podemos re-utilizar derivadas anteriores

1: $$\frac{\partial p}{\partial x_1} = \boxed{\frac{\partial p}{\partial h_1}\frac{\partial h_1}{\partial z_1}\frac{\partial z_1}{\partial x_1} + \frac{\partial p}{\partial h_2}\frac{\partial h_2}{\partial z_1}}\frac{\partial z_1}{\partial x_1} + \frac{\partial p}{\partial h_1}\frac{\partial h_1}{\partial z_2}\frac{\partial z_2}{\partial x_1} + \frac{\partial p}{\partial h_2}\frac{\partial h_2}{\partial z_2}\frac{\partial z_2}{\partial x_1}$$

$$\frac{\partial p}{\partial x_2} = \frac{\partial p}{\partial h_1}\frac{\partial h_1}{\partial z_1}\frac{\partial z_1}{\partial x_2} + \frac{\partial p}{\partial h_2}\frac{\partial h_2}{\partial z_1}\frac{\partial z_1}{\partial x_2} + \frac{\partial p}{\partial h_1}\frac{\partial h_1}{\partial z_2}\frac{\partial z_2}{\partial x_2} + \frac{\partial p}{\partial h_2}\frac{\partial h_2}{\partial z_2}\frac{\partial z_2}{\partial x_2}$$

$$\frac{\partial p}{\partial x_1} = \boxed{\left(\frac{\partial p}{\partial z_1}\right)\frac{\partial z_1}{\partial x_1}} + \frac{\partial p}{\partial h_1}\frac{\partial h_1}{\partial z_2}\frac{\partial z_2}{\partial x_1} + \frac{\partial p}{\partial h_2}\frac{\partial h_2}{\partial z_2}\frac{\partial z_2}{\partial x_1}$$

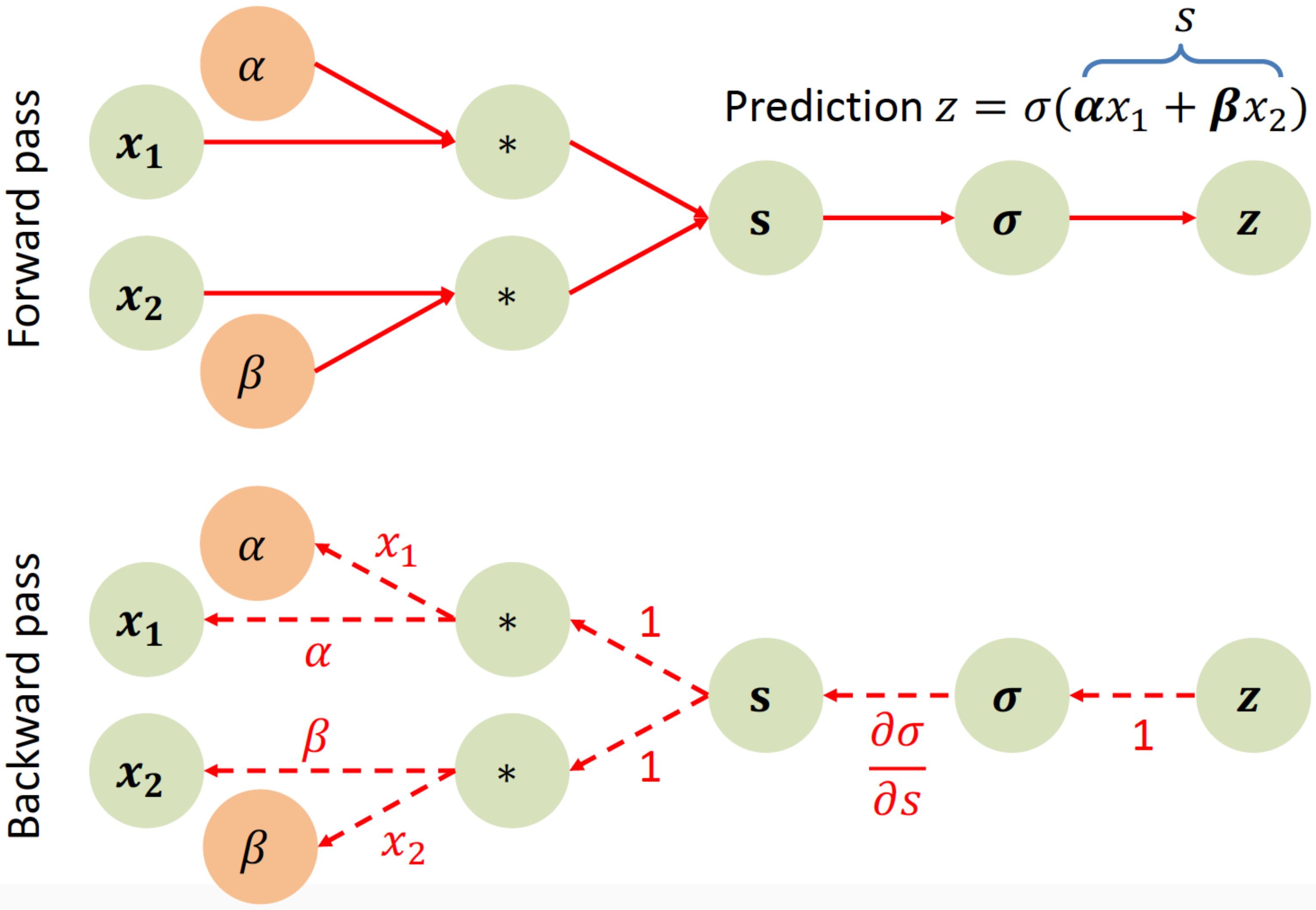
$$\frac{\partial p}{\partial x_2} = \boxed{\left(\frac{\partial p}{\partial z_1}\right)\frac{\partial z_1}{\partial x_2}} + \frac{\partial p}{\partial h_1}\frac{\partial h_1}{\partial z_2}\frac{\partial z_2}{\partial x_2} + \frac{\partial p}{\partial h_2}\frac{\partial h_2}{\partial z_2}\frac{\partial z_2}{\partial x_2}$$



Back-propagation es un modo de diferenciacion automático

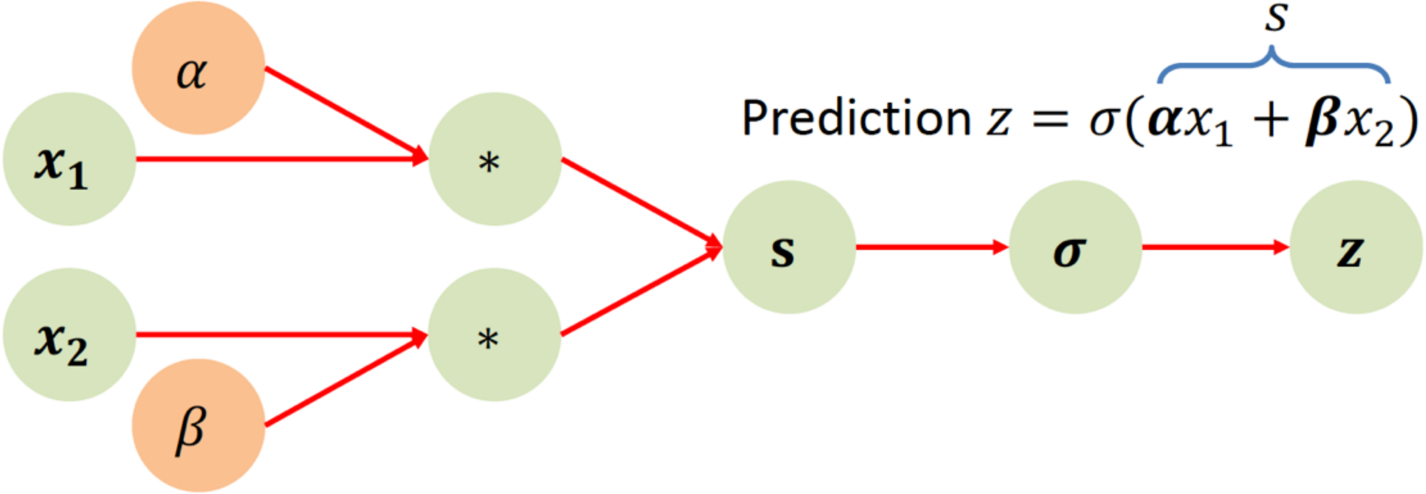
Es utilizado en todos los frameworks de AI hoy en dia.

Es muy rápido

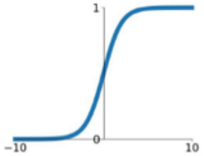


Funciones de Activación

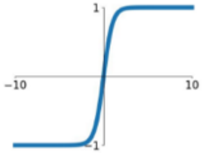
Prediction $z = \sigma(\overbrace{\boldsymbol{\alpha} x_1 + \boldsymbol{\beta} x_2}^{s})$



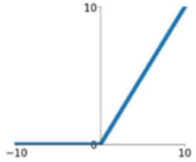
Sigmoid
$\sigma(x) = \frac{1}{1+e^{-x}}$



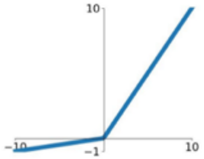
tanh
$\tanh(x)$



ReLU
$\max(0, x)$

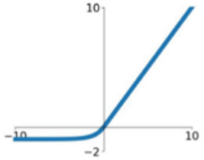


Leaky ReLU
$\max(0.1x, x)$

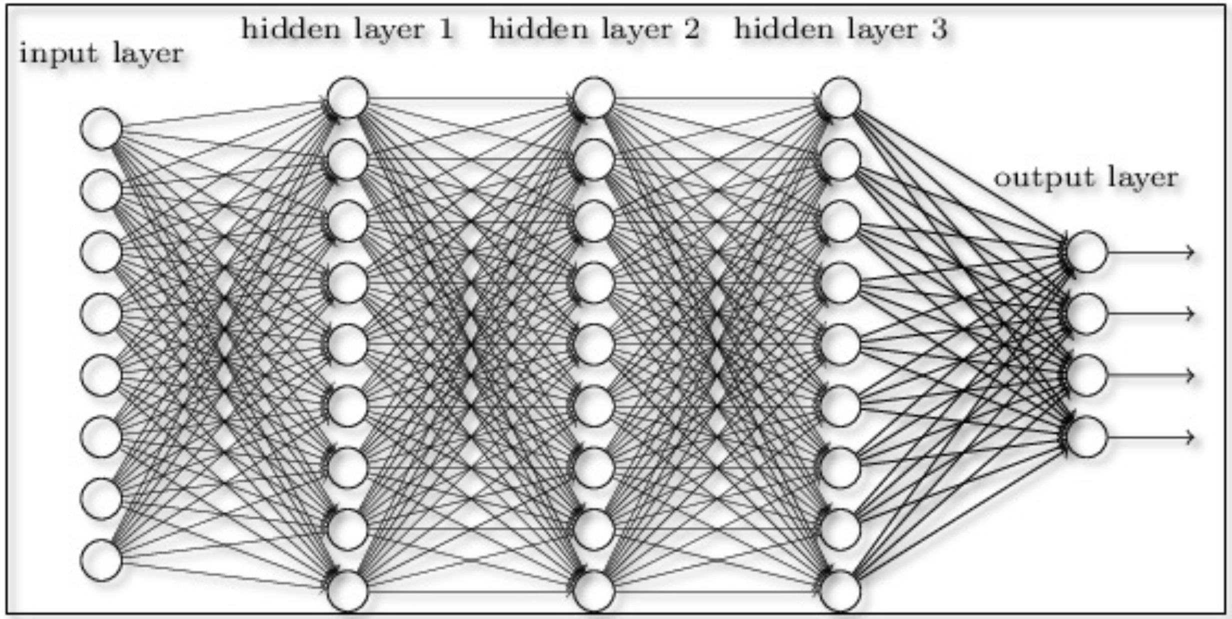


Maxout
$\max(w_1^T x + b_1, w_2^T x + b_2)$

ELU
$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$



Deep Neural Networks

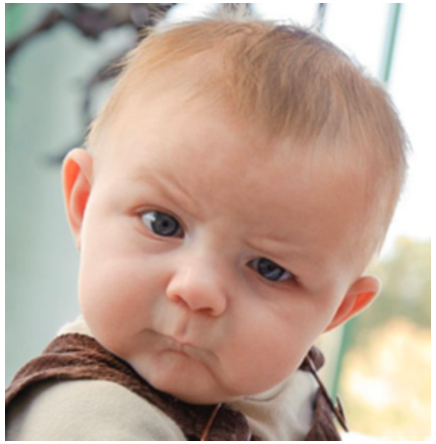



```
tf.keras.backend.clear_session()

model = Sequential()

model.add(Dense(hidden_nodes1, input_dim=input_nodes,
activation='relu'))
model.add(Dense(10, activation='relu’))
model.add(Dense(10, activation='relu'))
model.add(Dense(10, activation='relu'))
model.add(Dense(1, activation='sigmoid'))

model.compile(loss='binary_crossentropy',
              optimizer='adam',
              metrics=['accuracy'])
```

Por qué las NN son tan poderosas?

1. A MLP can approximate any function.

   Cybenko, G. (1989) - Approximation by Superpositions of a Sigmoidal Function



Lo que se resume en que no importa que tan complejo es el decision boundary, un MLP siempre puede aproximarlo

(Cuando tienes datos y suficiente poder computacional)

Questions

UDACITY

Programs Careers For Enterprise | Blog **MY CLASSROOM**

Home > Catalog >

FREE COURSE

Intro to TensorFlow for Deep Learning

by TensorFlow

This course is a practical approach to deep learning for software developers

START FREE COURSE

RELATED NANODEGREE PROGRAM

Machine Learning Engineer

by aws

Get a Nanodegree certificate that accelerates your career!

About this Course

Learn how to build deep learning applications with TensorFlow. This

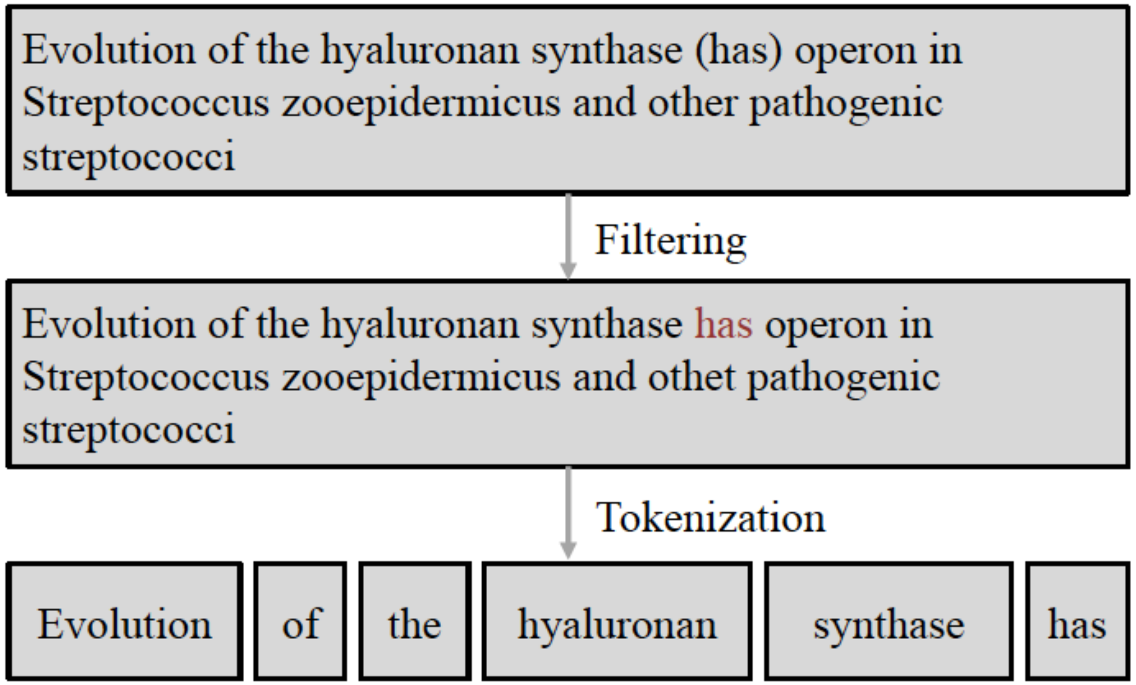
COURSE COST	TIMELINE	SKILL LEVEL
Free	Approx. 2 Months	Intermediate

Primeros pasos en NLP

- Bag of words and tokens
- Lowercase
- Stemming
- Word embeddings

Filtros y Tokens

Cómo podemos representar una cadena de texto?



Eliminar todos los caracteres inecesarios
(, . $ % & / ! “ · < >) _ ^ * Ç ¨ ` + ´

Separar las cadenas en palabras individuales

Lowercase & Stemming

Lowercase

Raw	Lowercased
Canada CanadA CANADA	canada
TOMCAT Tomcat toMcat	tomcat

Stemming

	words	stemmed words
0	connect	connect
1	connected	connect
2	connection	connect
3	connections	connect
4	connects	connect

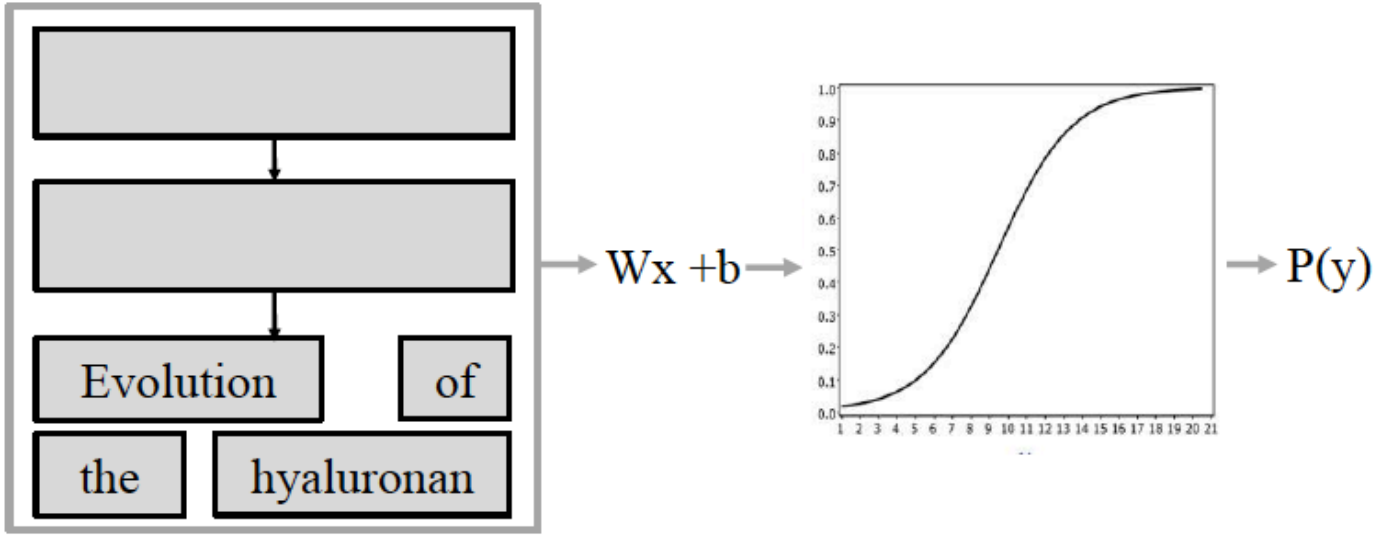
Bag of Words

Journal of Artificial Intelligence Research
JAIR is a refereed journal, covering the areas of Artificial intelligence, which is distributed free of charge over the Internet. Each volume of the journal is also published by Morgan Kaufmann

0	learning
3	journal
2	intelligence
0	text
1	Internet
...	...

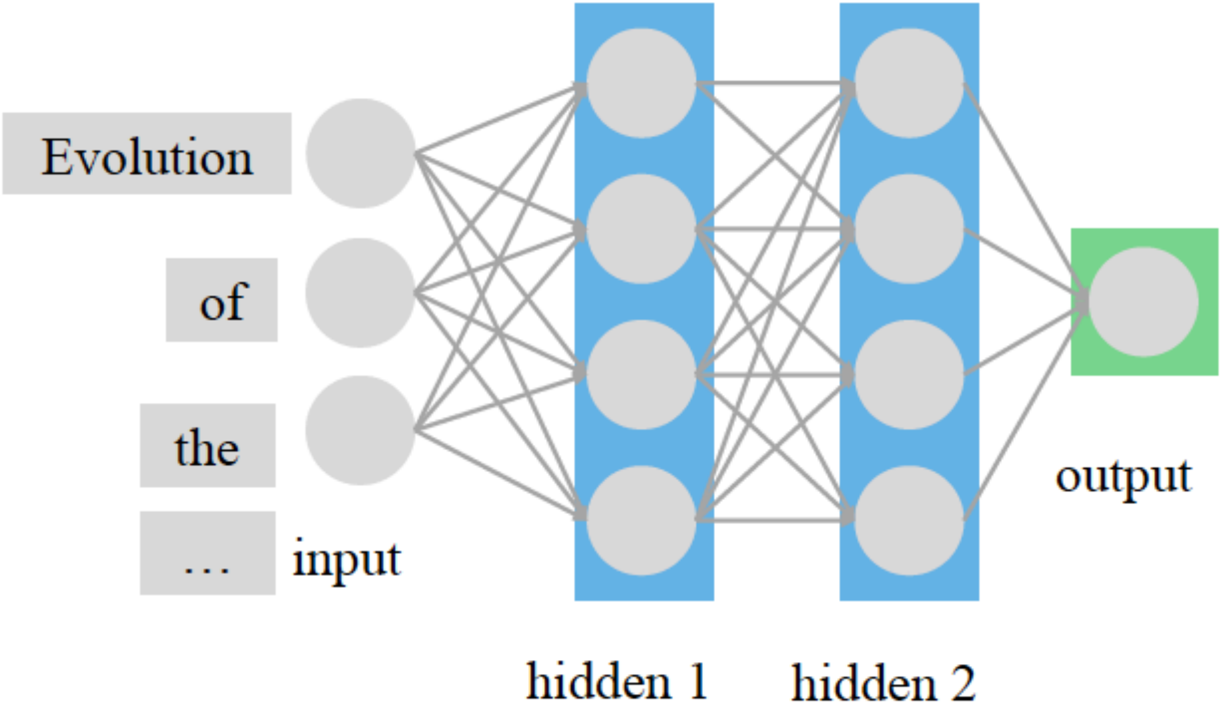
id	T1	T2	T3	T4	T5	.	.	.	.	n
1	1	1	1	1	1	1	0	0	0	0
2	1	1	1	0	1	1	1	0	0	0
3	1	1	1	0	1	0	0	1	1	0
4	1	1	1	0	1	0	0	1	0	1

Log reg + BoW



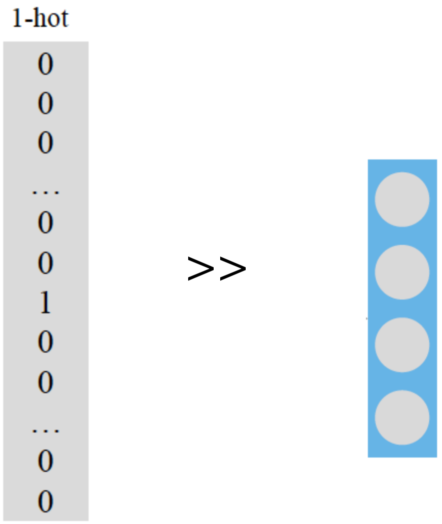
Cuantas variables tendrá este modelo?

NN + BoW

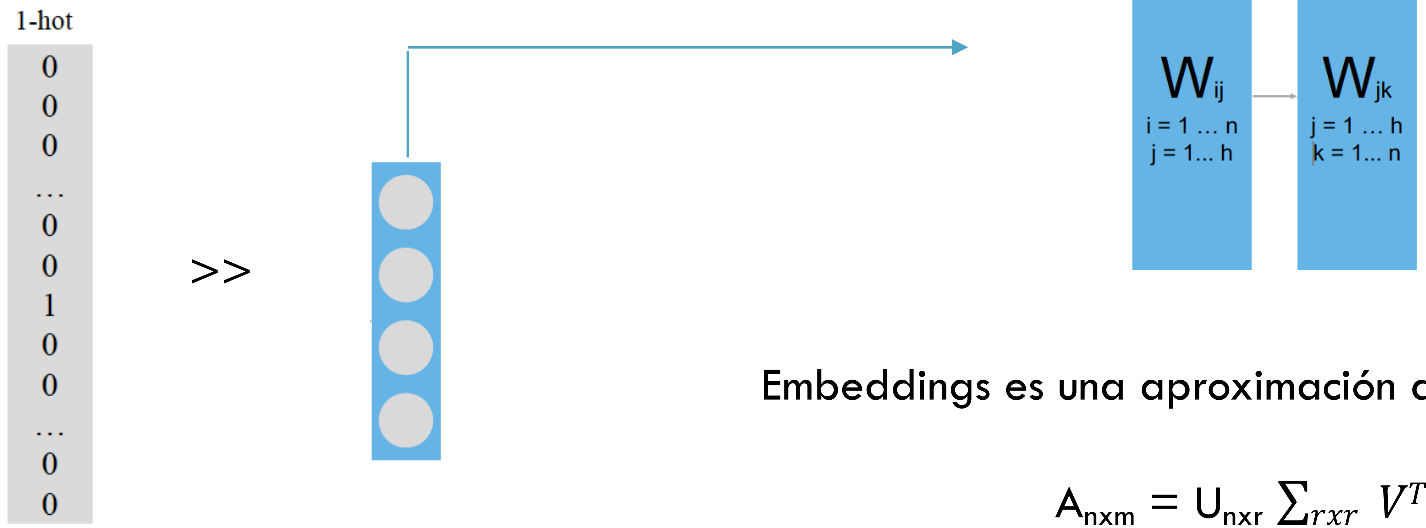


Cuantas variables tendrá este modelo?

Embeddings



Embeddings

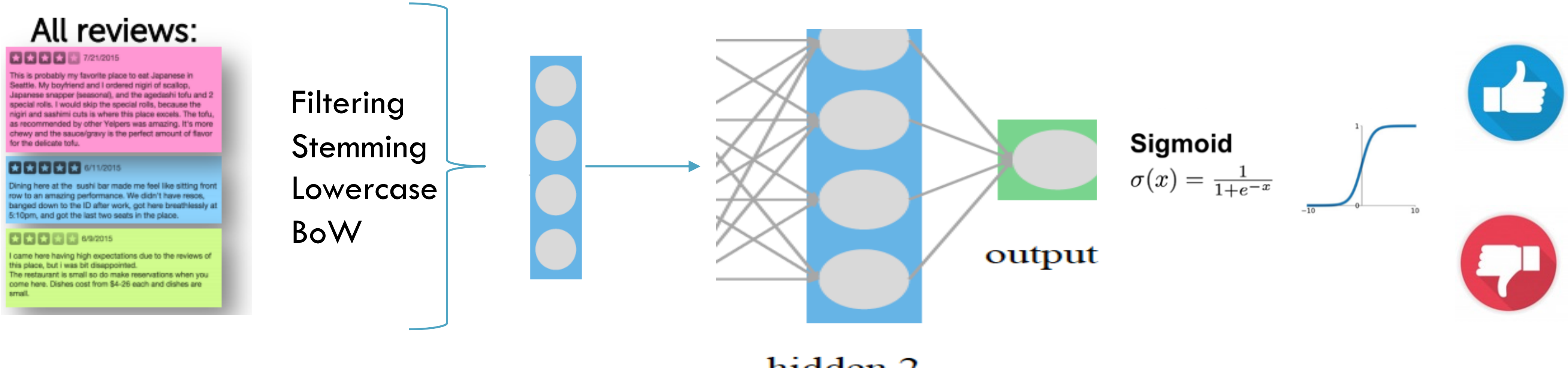


Embeddings es una aproximación al teorema de SVD

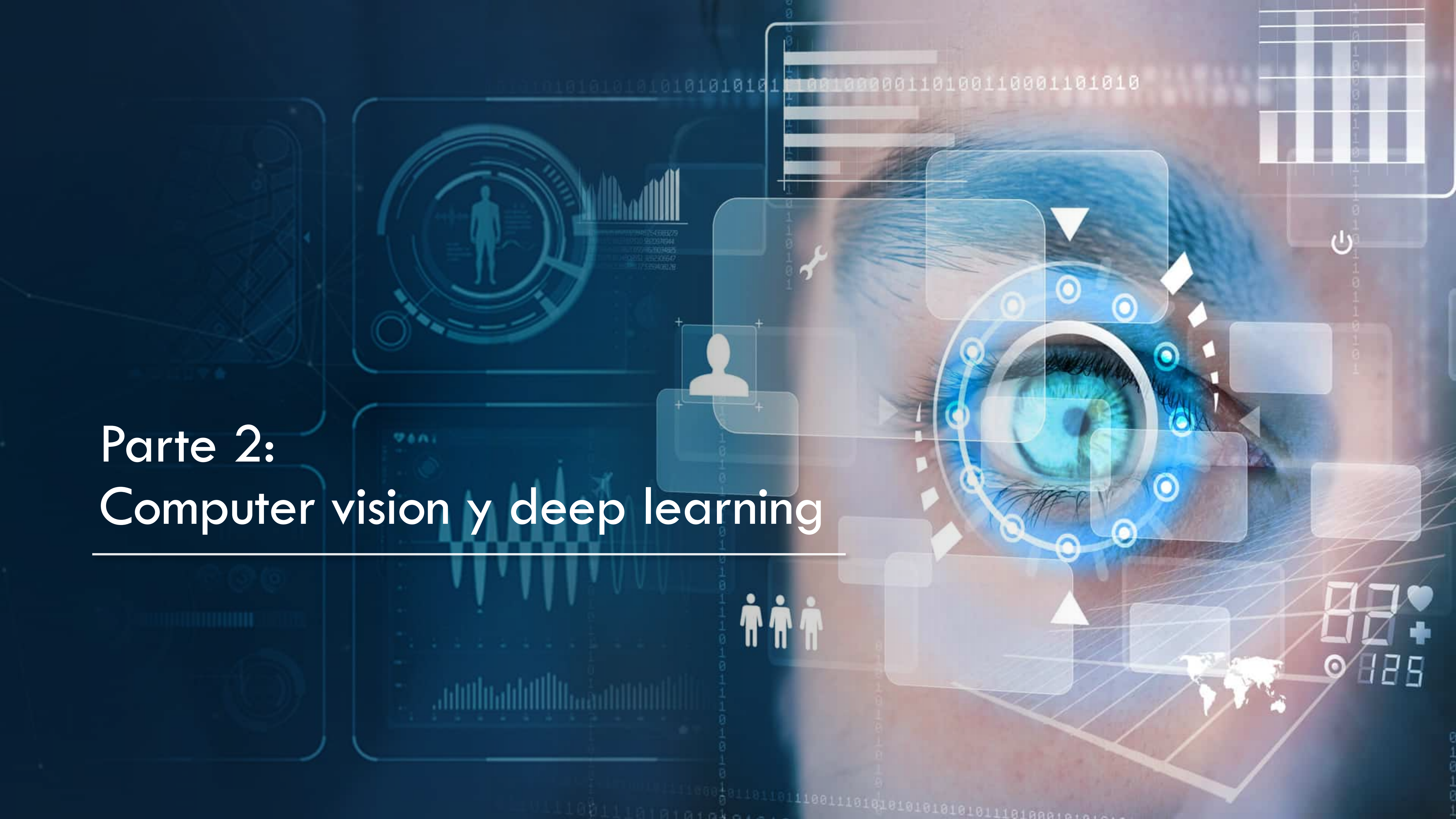
$$A_{nxm} = U_{nxr} \sum_{rxr} V^{T}{}_{rxm}$$

Donde $\sum_{rxr}$ (matríx diagonal) representa la decomposicion latente de la matrix original A_{nxm}.
Y los r factores de la matrz son denominados factores latentes, SVD asegura la existencia de $\sum_{rxr}$.

Sentiment Analysis

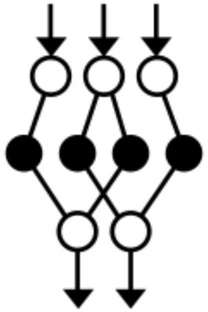


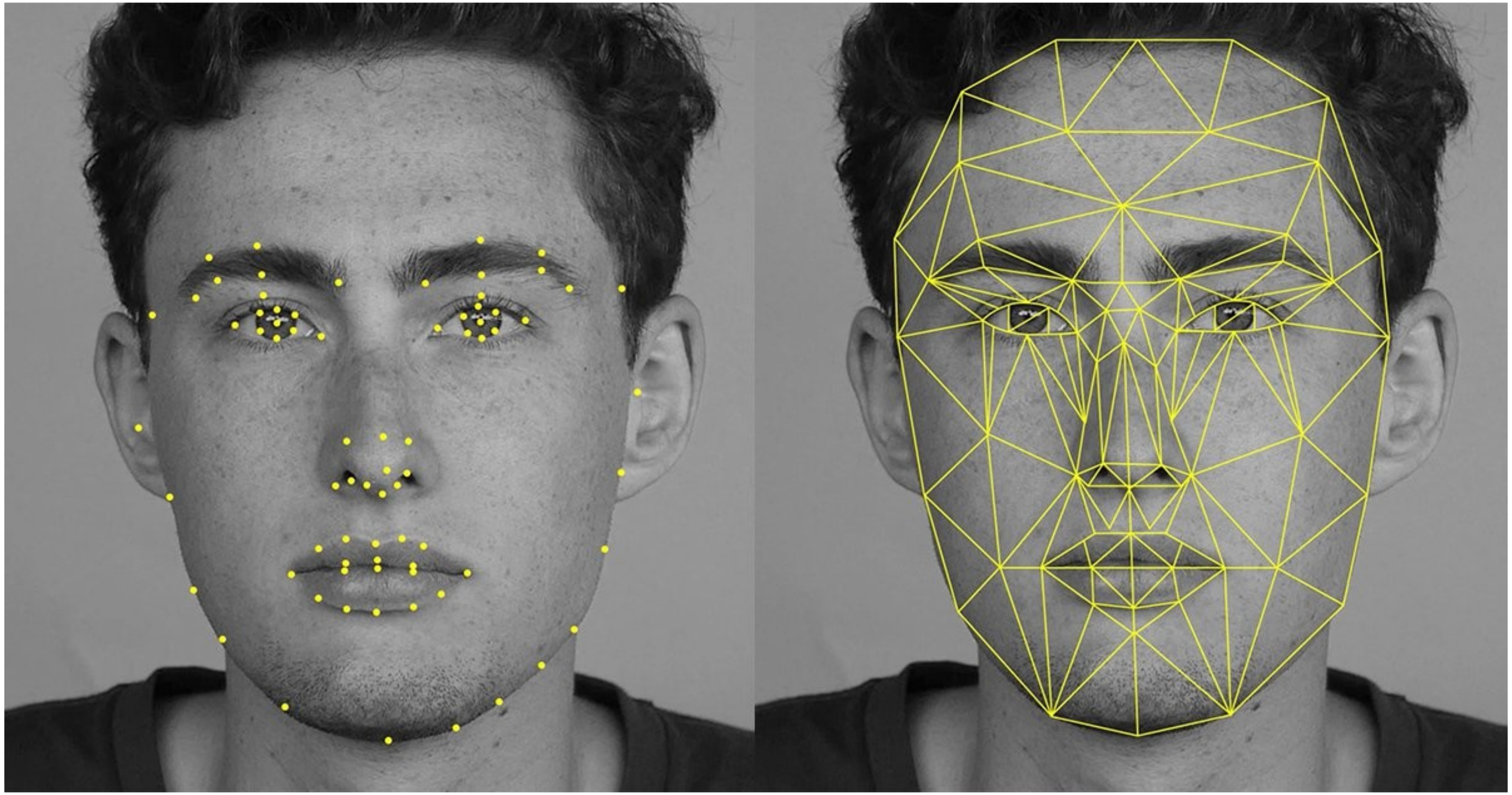
Demo 1
NN In
Tensorflow



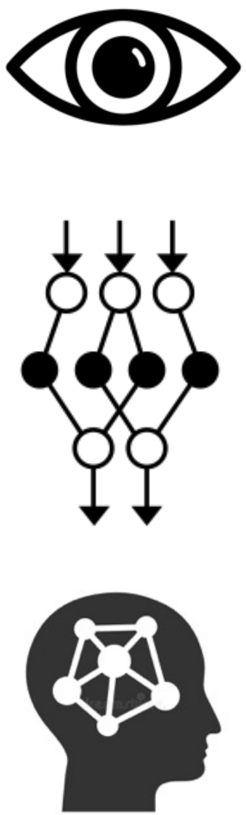
Parte 2:
Computer vision y deep learning

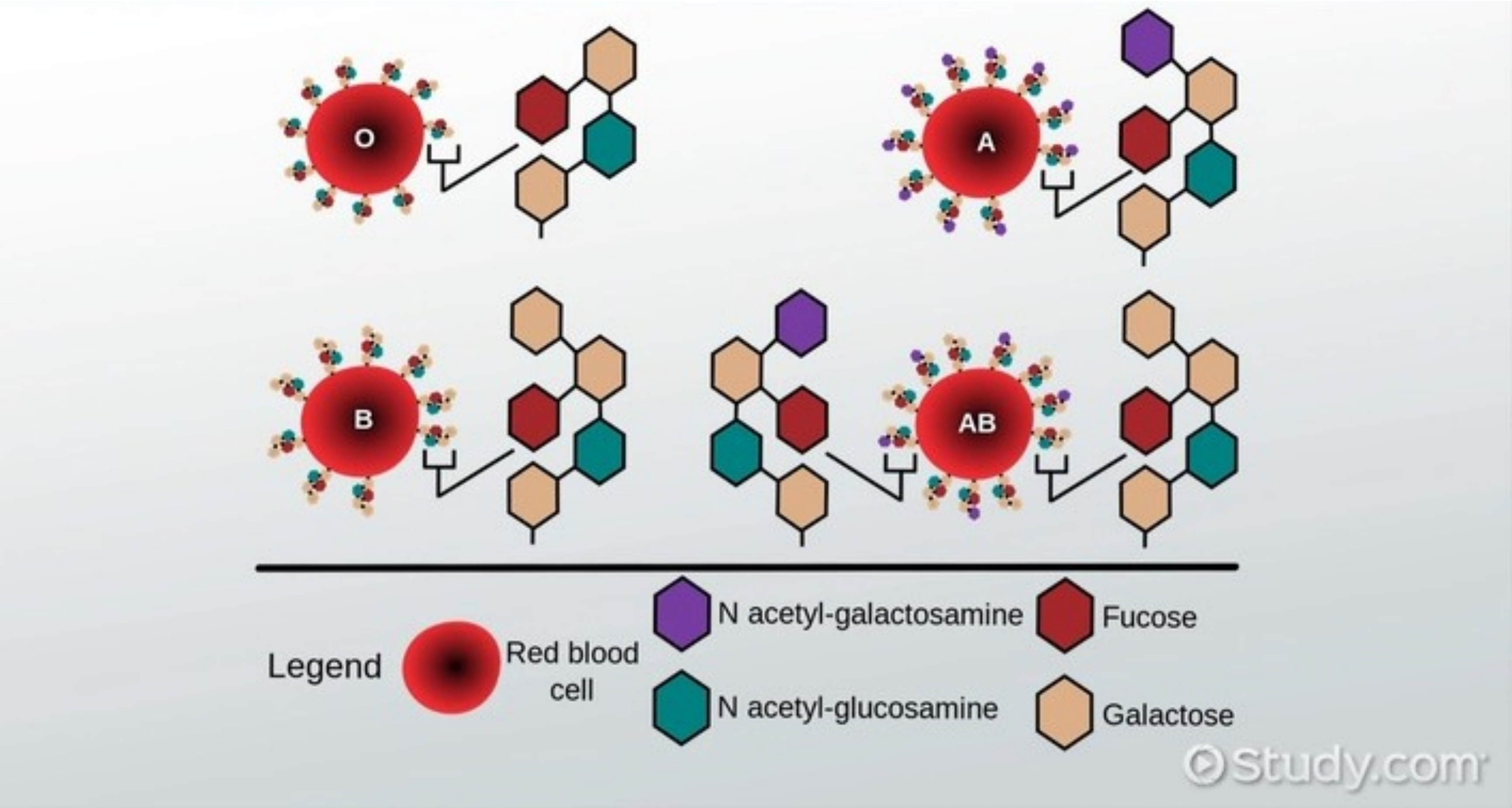
Facial detection & recognition





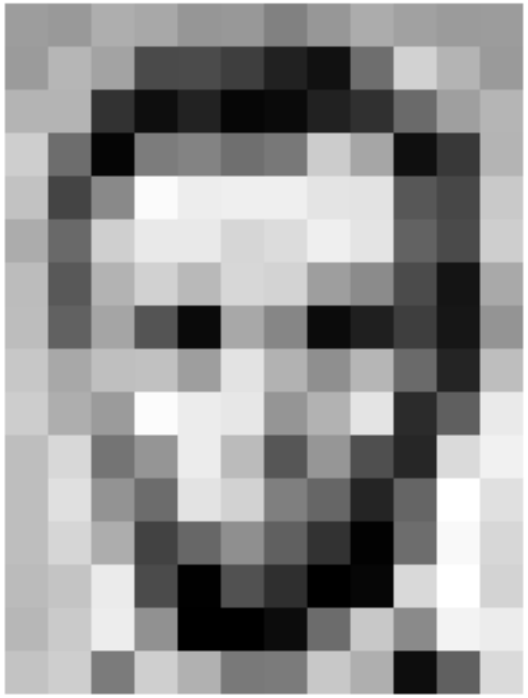
Infecction detection & recognition

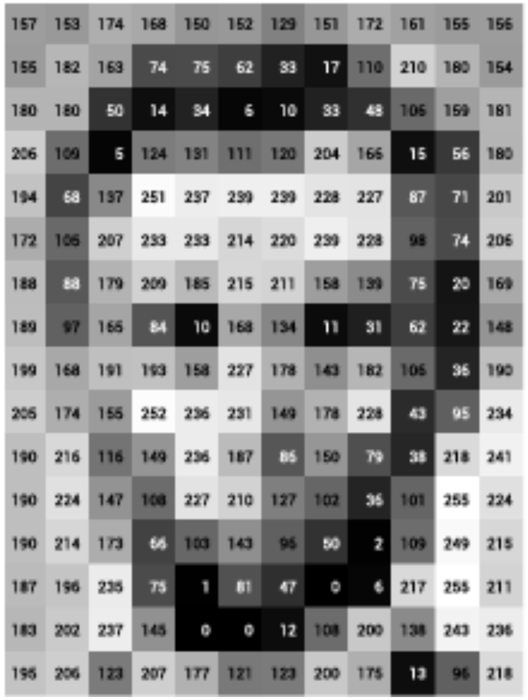




Cómo ve una computadora?

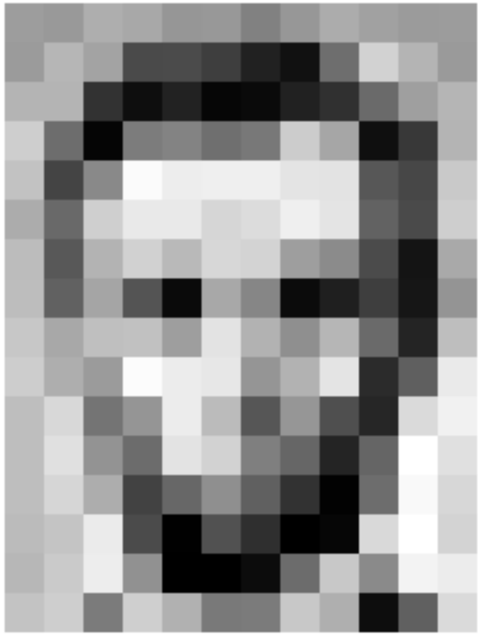
Las imagenes son matrices numericas





157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

Clasificación



Input

157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

Representación

Classificador

Lincoln	0.8
Obama	0.1
AMLO	0.05
Madona	0.05

Variables

Naríz
Ojos
Boca

Llantas
Placa
Luces

Ventanas
Puerta
Jardín

Variables

Naríz
Ojos
Boca

Llantas
Placa
Luces

Ventanas
Puerta
Jardín

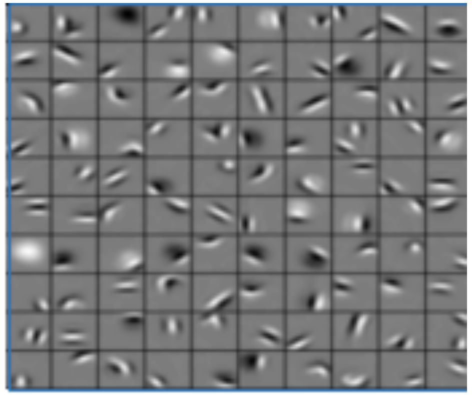
Opcion1: Usar domain Knowledge

- Crear
- Probar
- Seleccionar variables.

Cuál es el problema?

Neural Networks

Low Level



Hidden layer 1

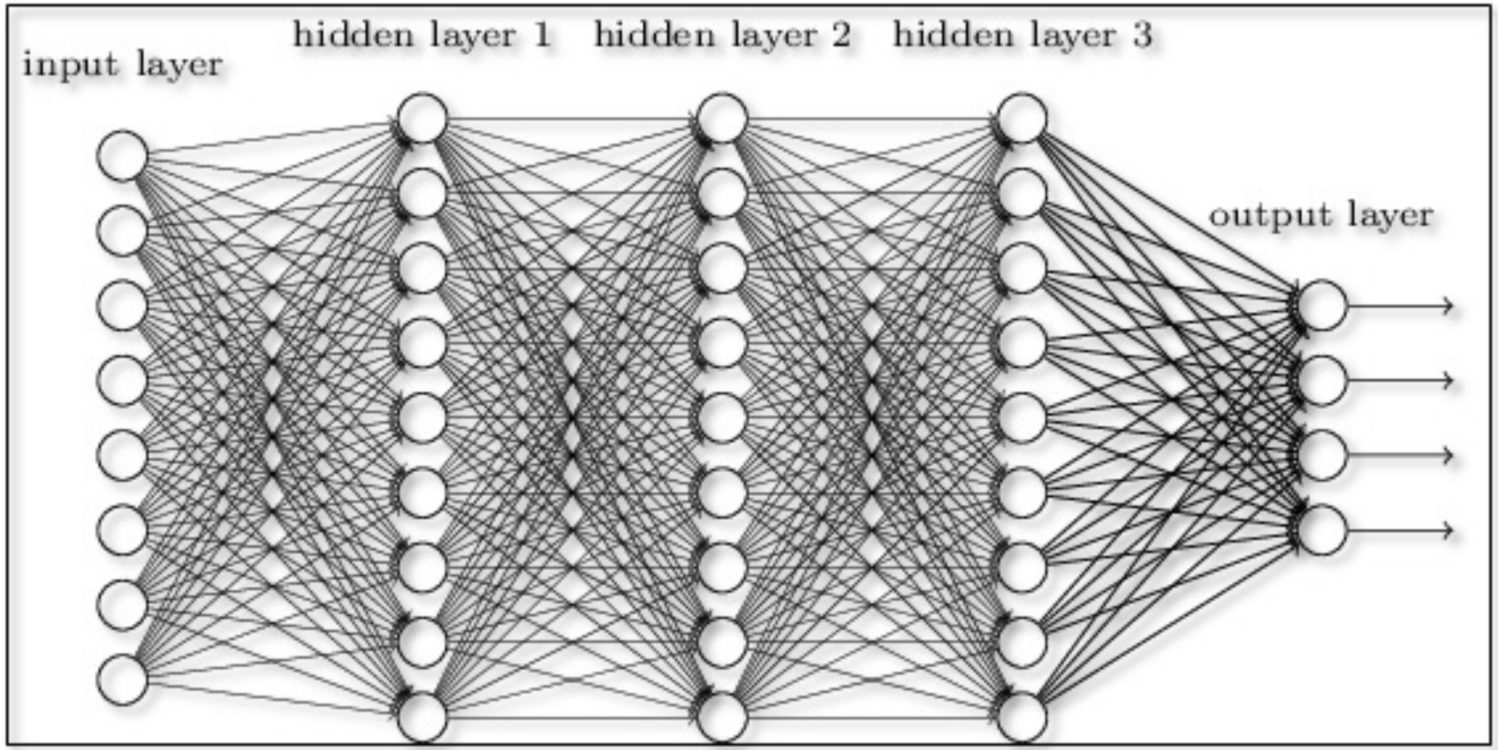
Mid Level

Hidden layer 2

High Level features

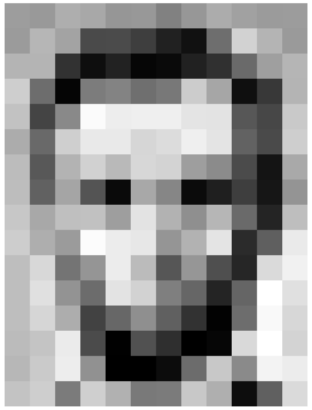
Hidden layer 3

Obteniendo Variables

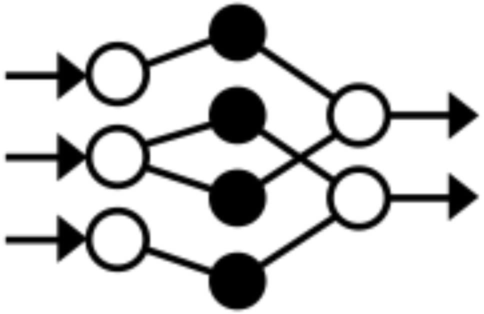


Deep Neural Network

2D Input



[255 x 255]

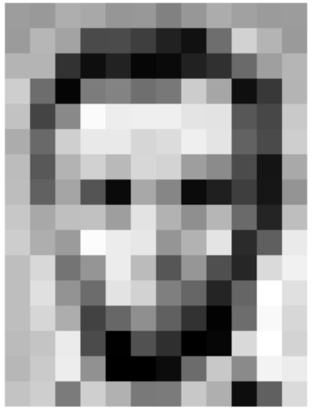


[65,025 ….   N   …..]

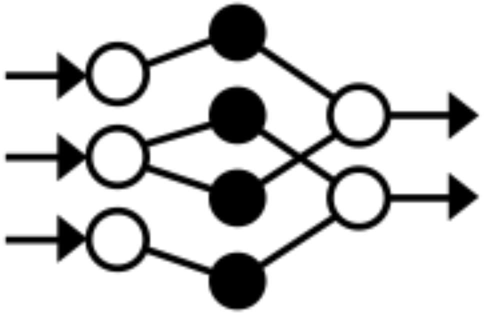
FNN

[10]

2D Input



[255 x 255]



[65,025 ….   N   …..]

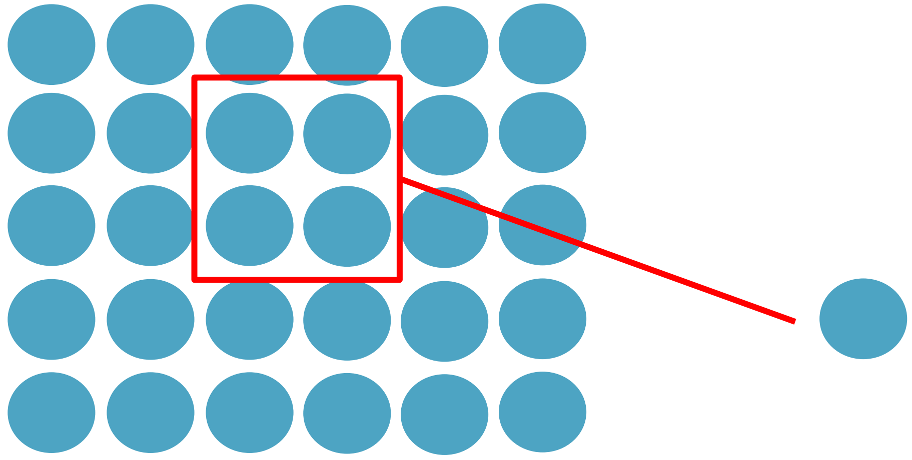
FNN

[10]

Qué otras opciones hay?

Utilizando la estructura espacial

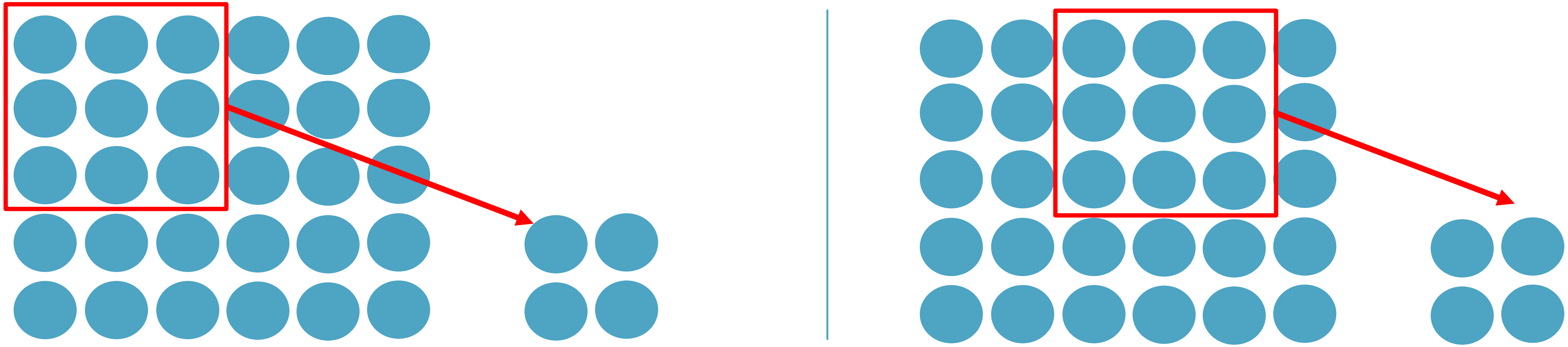
2D Input



Idea:

Connectar subconjuntos de datos que afecten un solo nodo de la red.

Utilizando la estructura espacial



Utilizamos un desplazamiento (slide) para obtener features de la imagen original

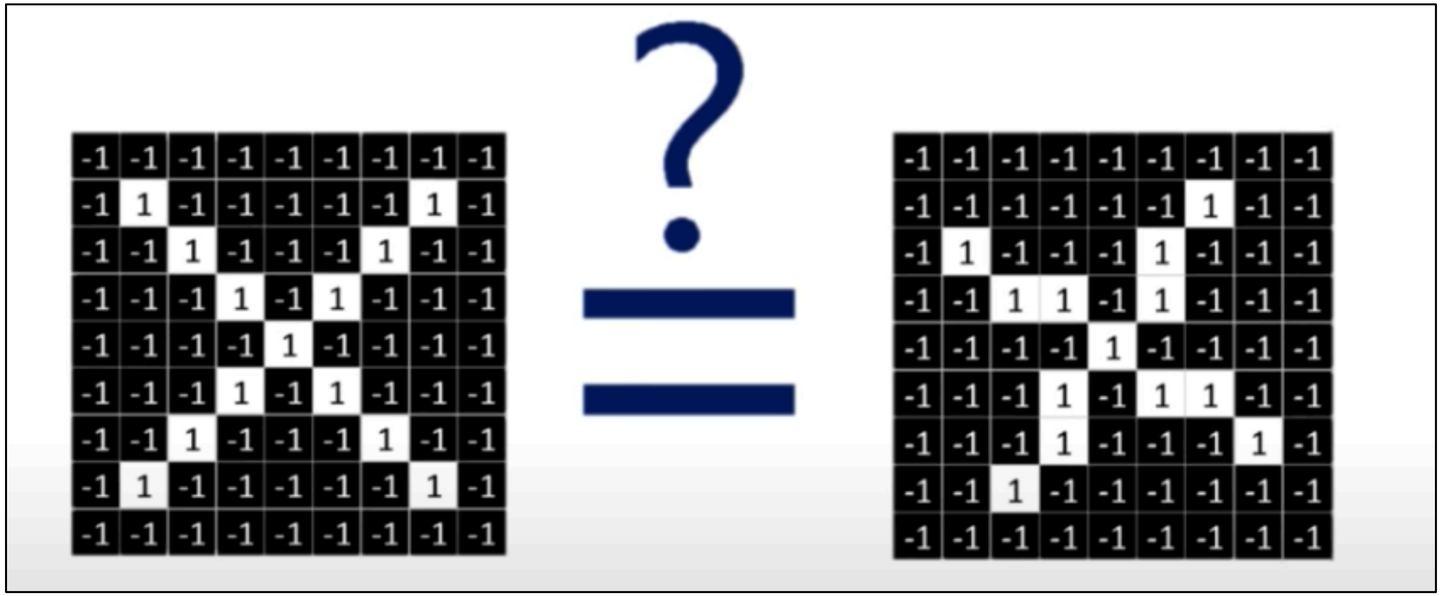
En la practica, ésta operación en Rojo se llama convolución con un filtro de 3x3.

Obteniendo Variables con Convoluciones

X vs X

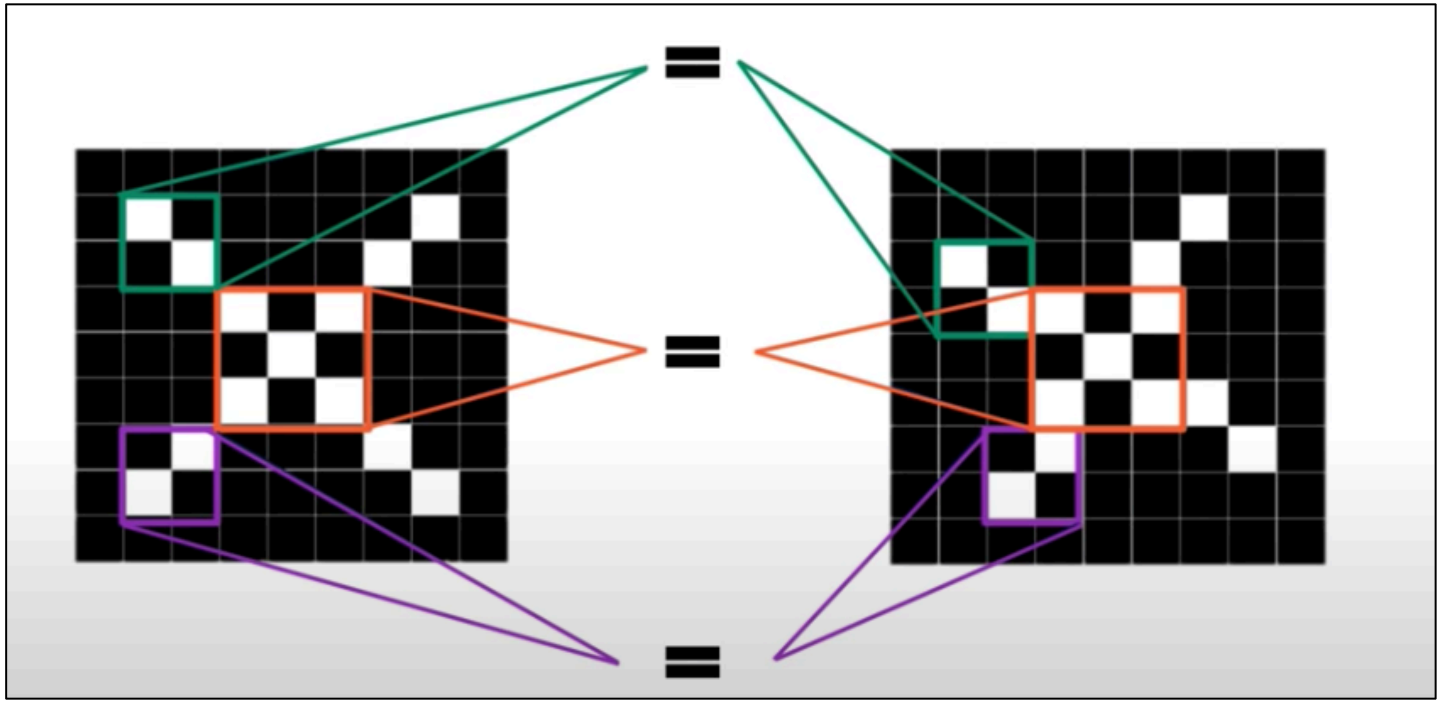
La imagen es representada como una matríz de pixeles…

Queremos clasificar una imagen como X incuso si está distorcionada o invertida,

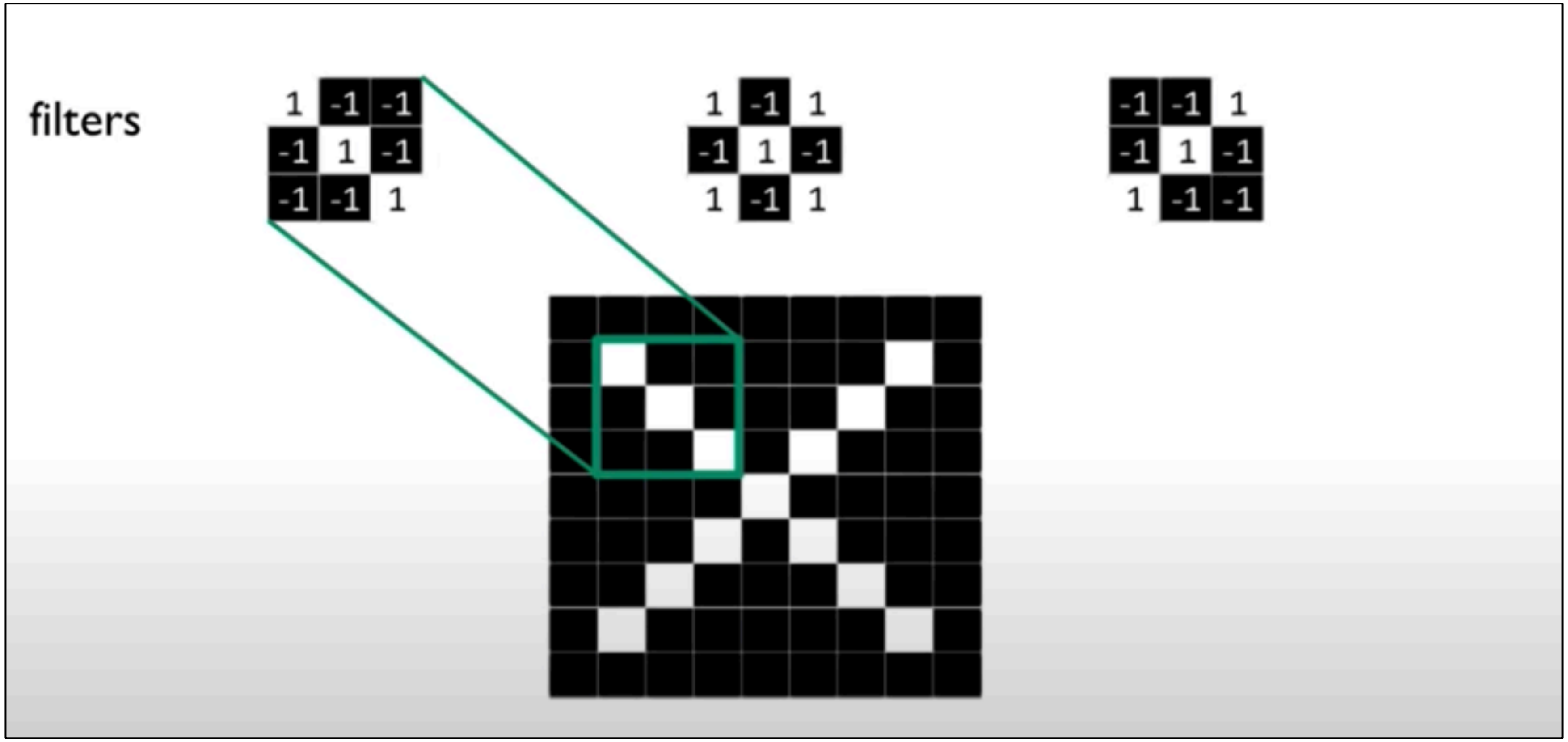


X vs X

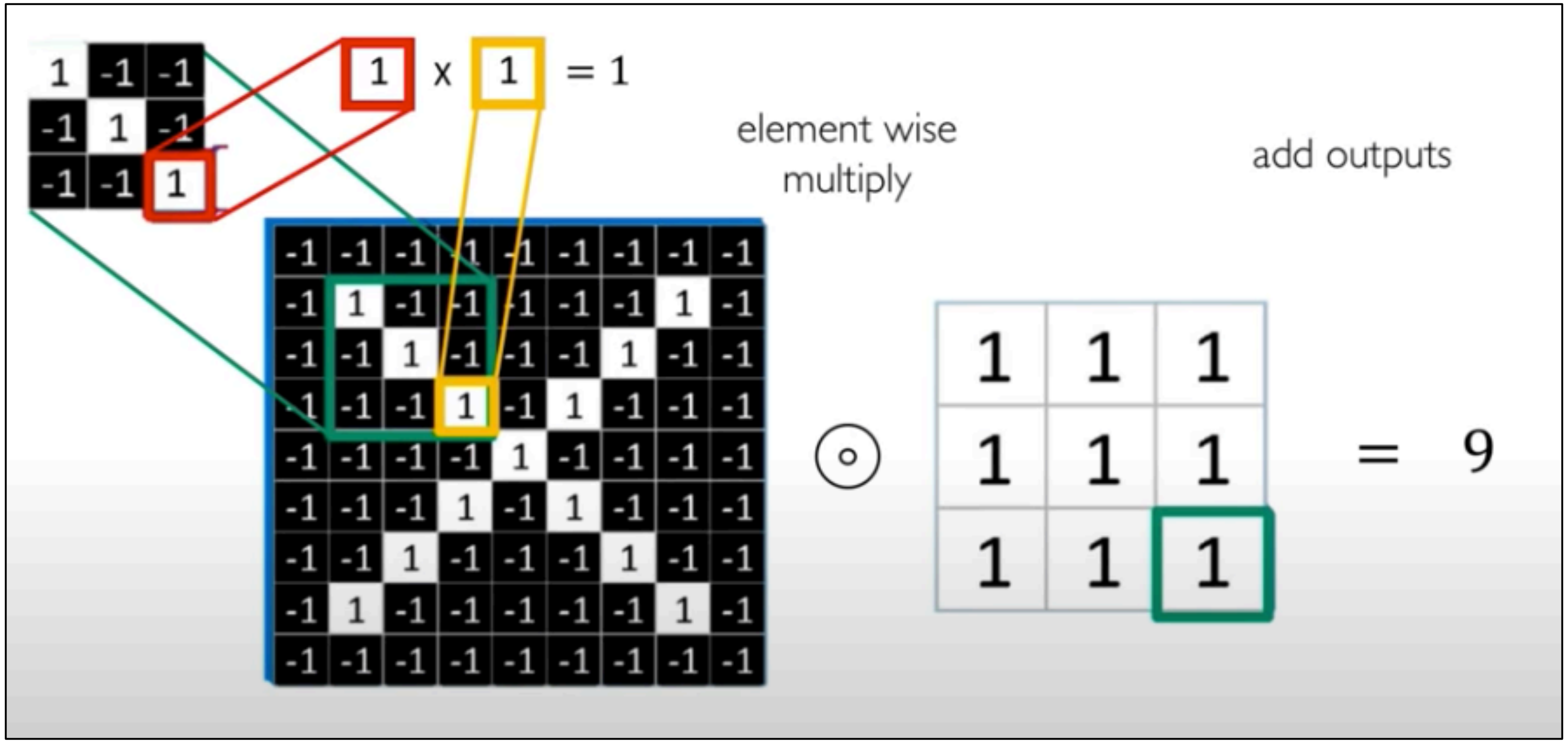
Lo que queremos es obtener variables que tengan simililtud considerando la estructura.



Utilizamos filtros para detectar patrones

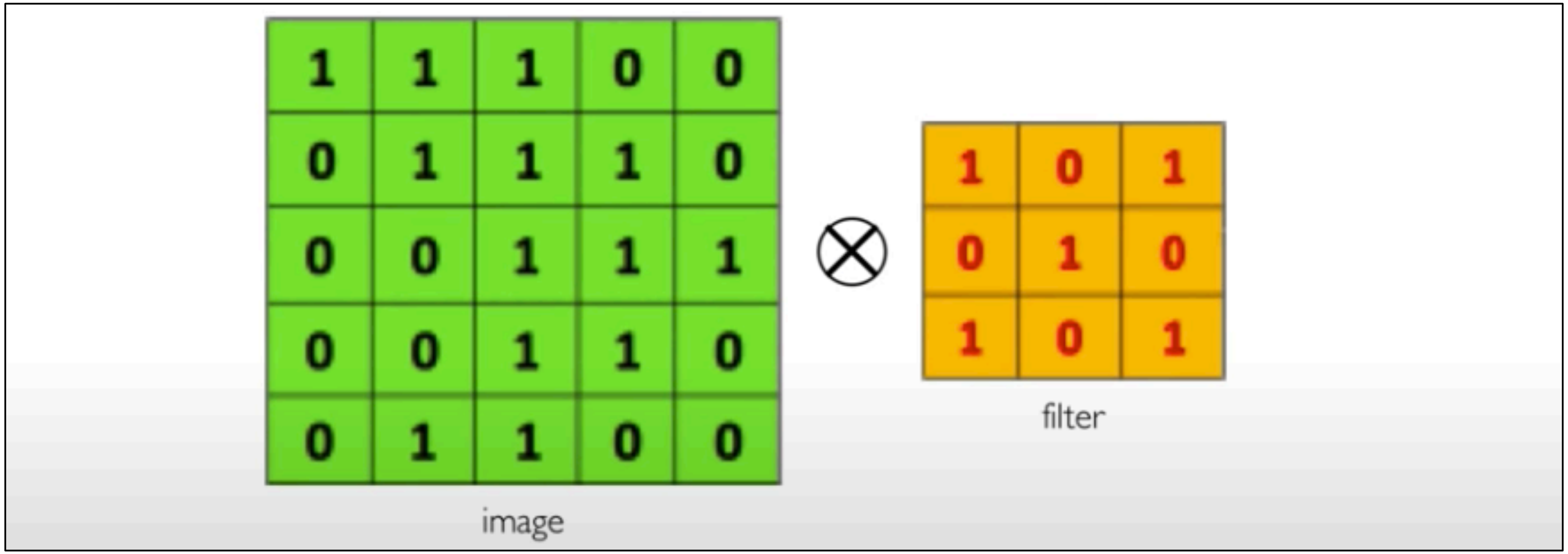


Utilizamos filtros para detectar patrones



Utilizamos filtros para detectar patrones

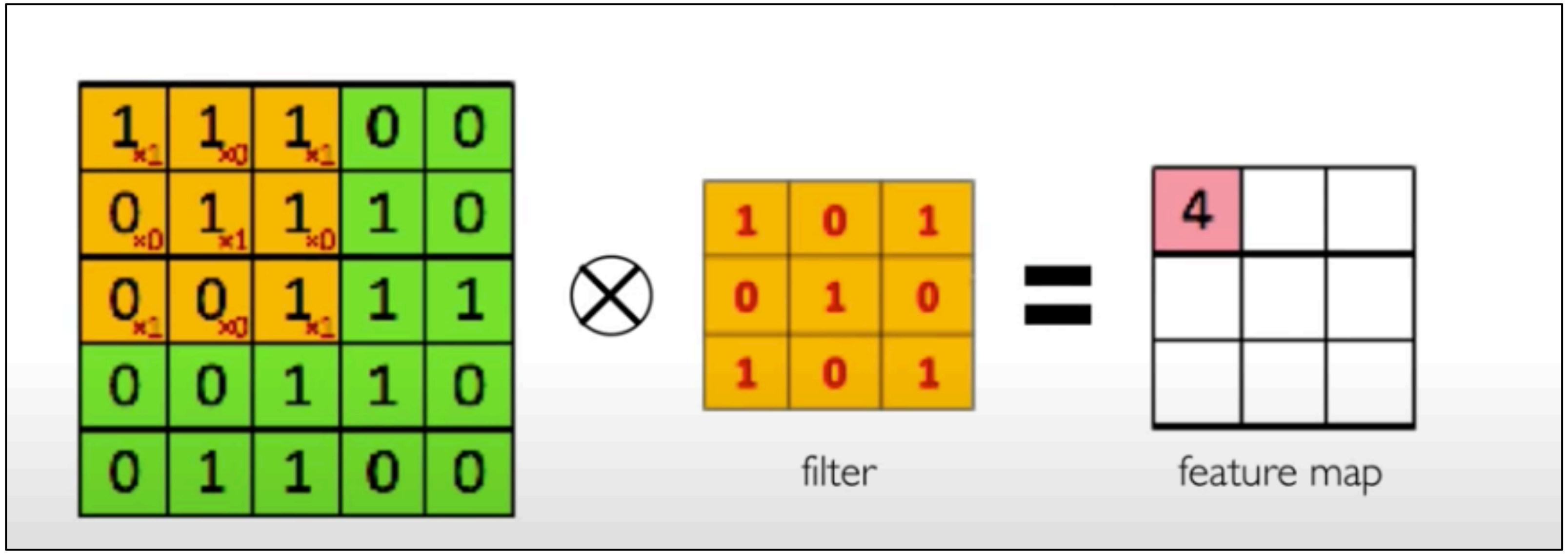
Vamos a calcular la convolución de una imagen de 5x5 con un filtro de 3x3



Vamos a deslizar el filtro de 3x3 sobre la imagen, multiplicar cada elemento y sumar.

Utilizamos filtros para detectar patrones

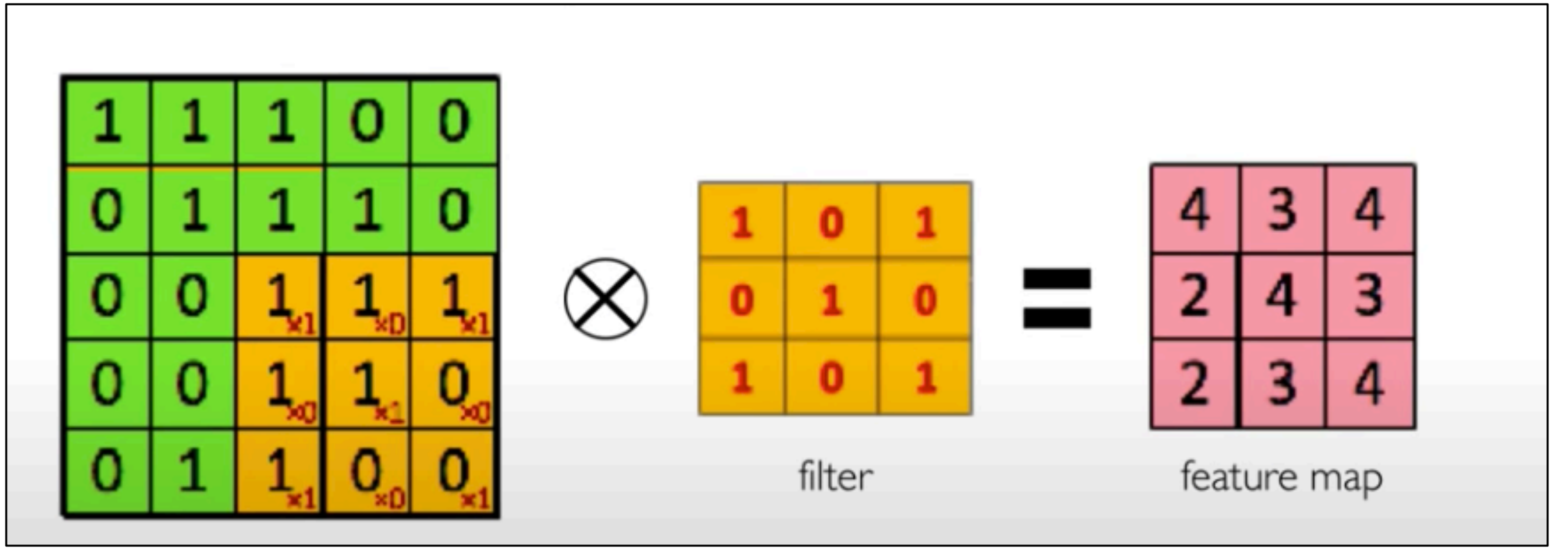
Vamos a calcular la convolución de una imagen de 5x5 con un filtro de 3x3



Vamos a deslizar el filtro de 3x3 sobre la imagen, multiplicar cada elemento y sumar.

Utilizamos filtros para detectar patrones

Vamos a calcular la convolución de una imagen de 5x5 con un filtro de 3x3



El mapeo de variables (feature map) nos indica en donde hay datos relevantes en la imagen original

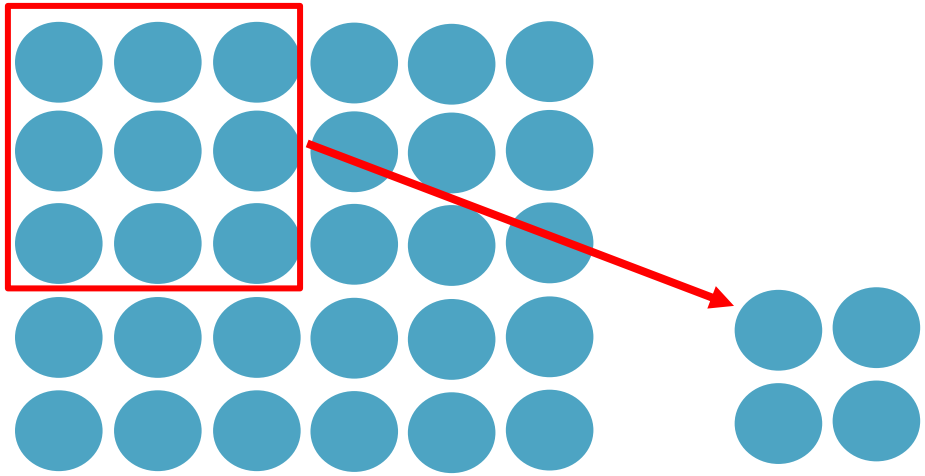
Utilizar filtros diferentes

Original — Sharpen — Edge Detect — "Strong" Edge Detect

Resumen:

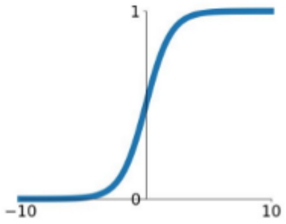
1. Aplicar filtros nos ayuda a extrar variables locales
2. Podemos utilizar diferentes iltros para obtener multiples variables.
3. Las convoluciones "comparten" información de la imagen original

Convolución



Sigmoid

$$\sigma(x) = \frac{1}{1+e^{-x}}$$

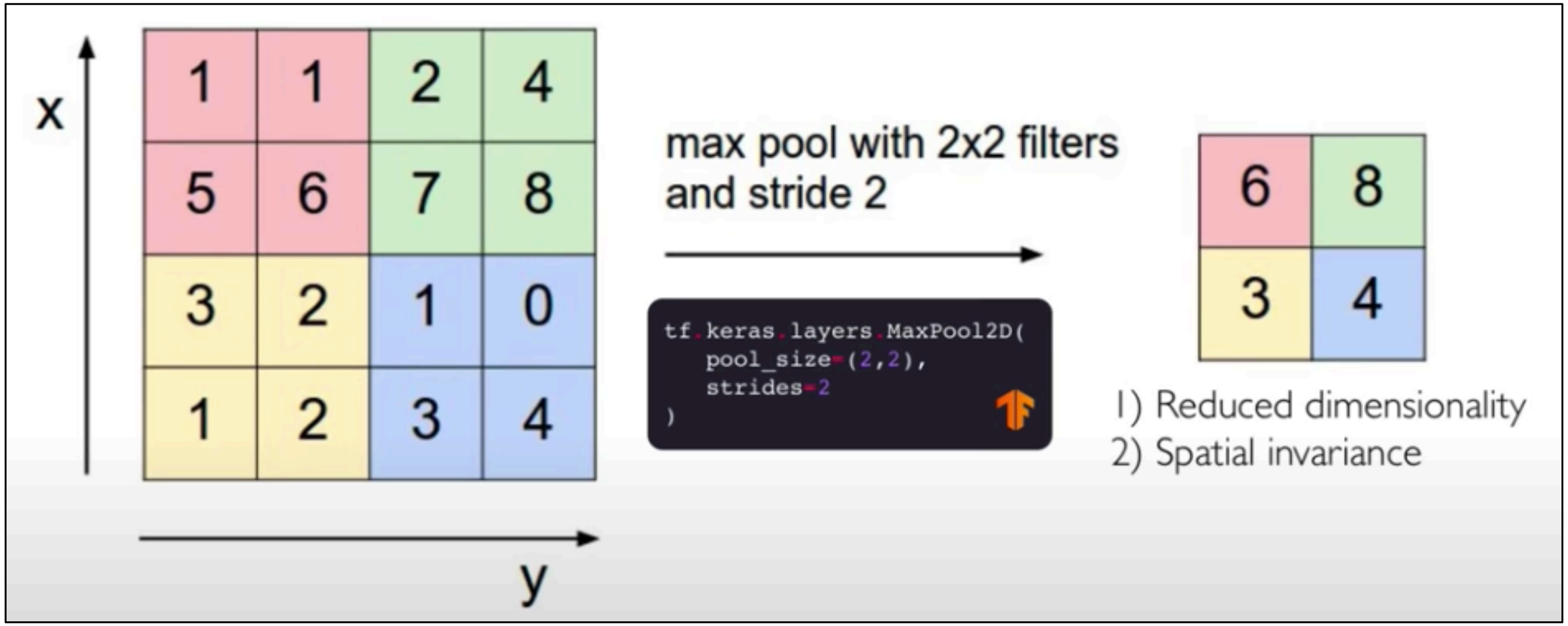


Convolución

$$\sum_i \sum_j w_{ij} + x_{i\, +\, p,\, j\, +\, q} + b$$

Para neuronas p, q en hidden layer

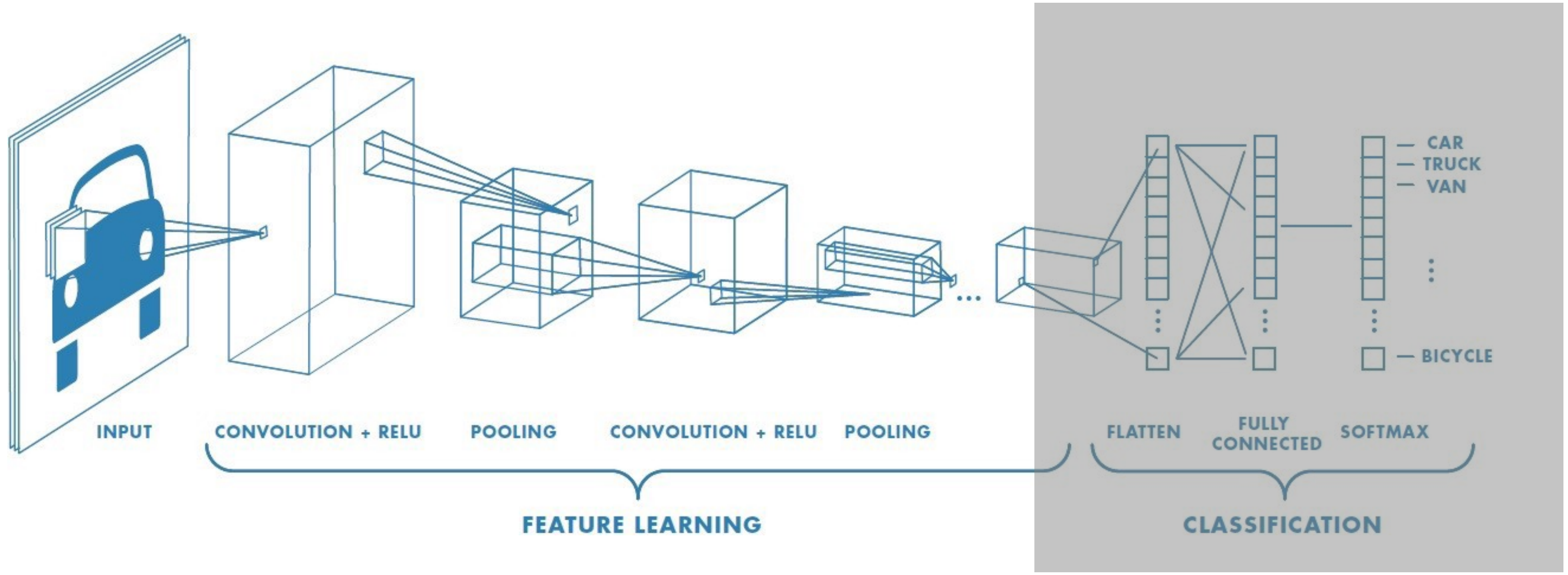
Maxpooling



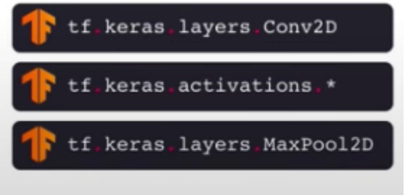
Cuál es otra forma de reducir la dimensión despues de una convolución?

Convolutional Neural Networks (CNNs)

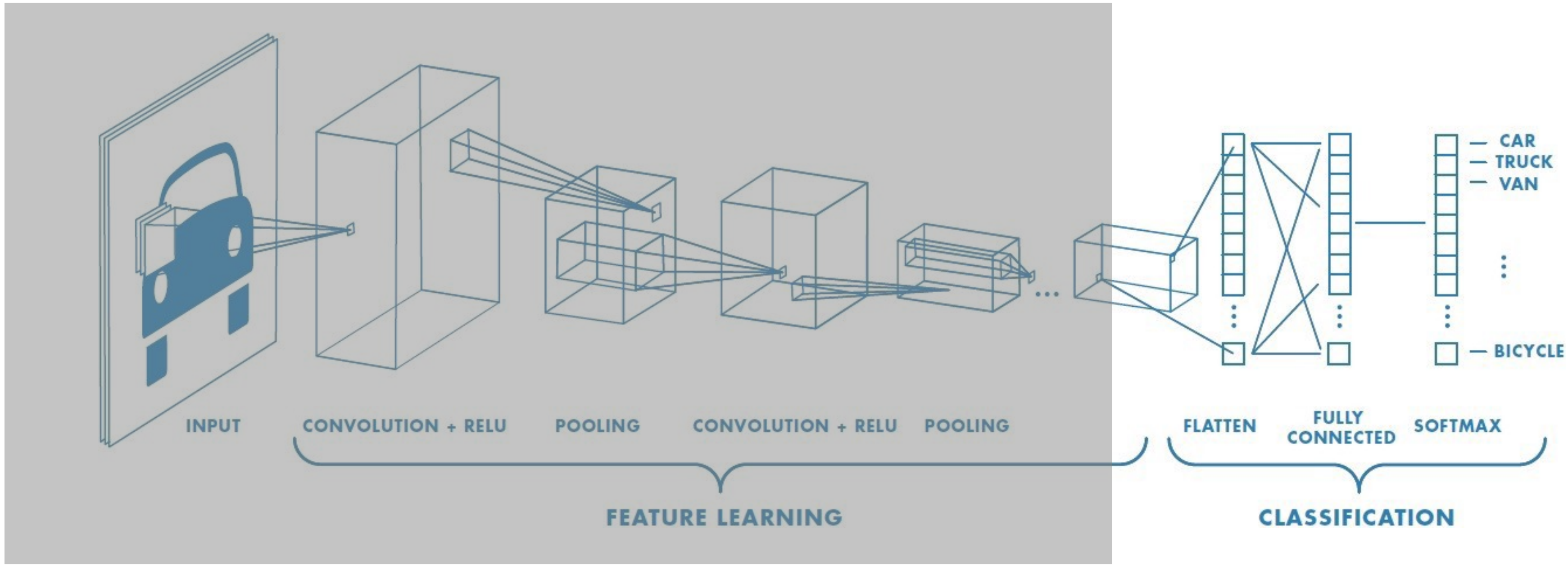
CNN



1. Convoluciones aplican filtros para general feature maps.
2. Aplicar función de activación no linear como ReLU.
3. Subsampling via maxpooling para reducir la dimención de los features maps.



CNN



4. Fully connected layers sobre las variables que obtuvimos de las convouciones.
5. Función de activación sigmoidal para obtener una probabilidad.

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

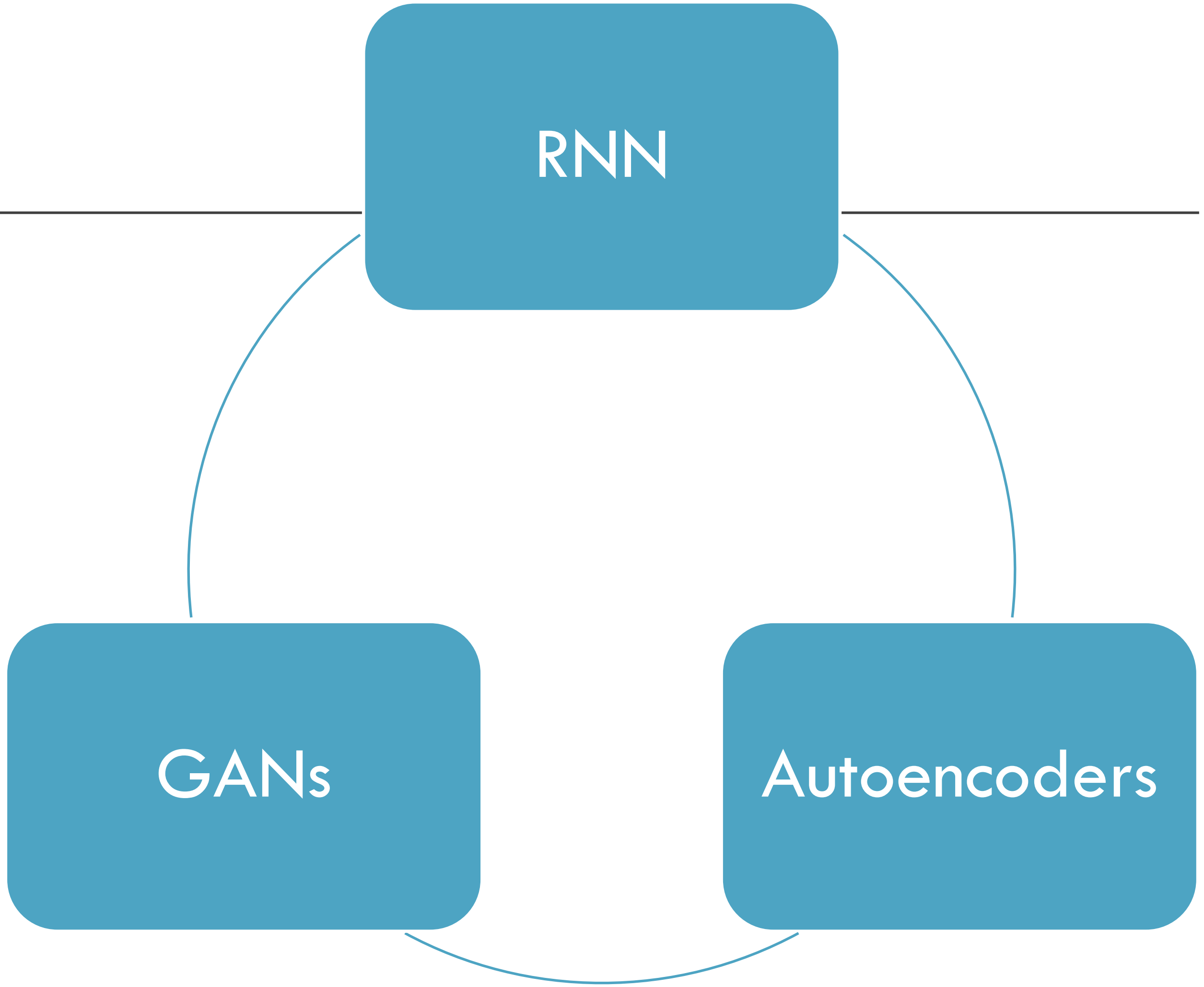
CNN In Tensorflow


Hand bag
Shopping Bags
Shopping Bag
Shopping Bags
Context classification

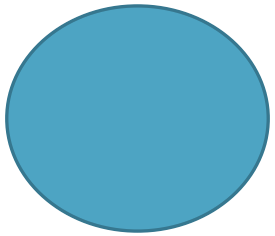
Object recognition for Self-driving cars.

Arquitecturas avanzadas

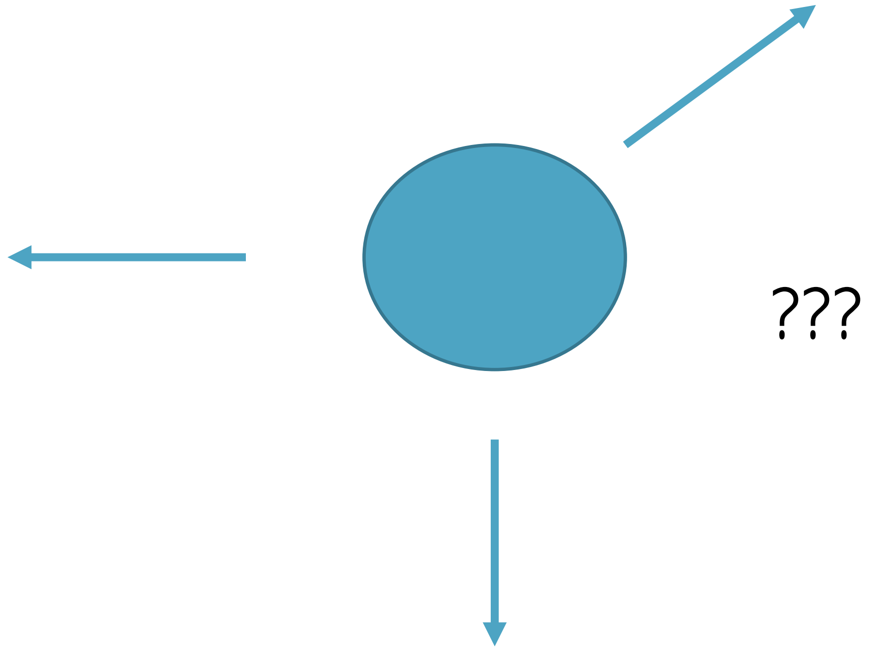
Arquitecturas avanzadas


RNN
GANs
Autoencoders

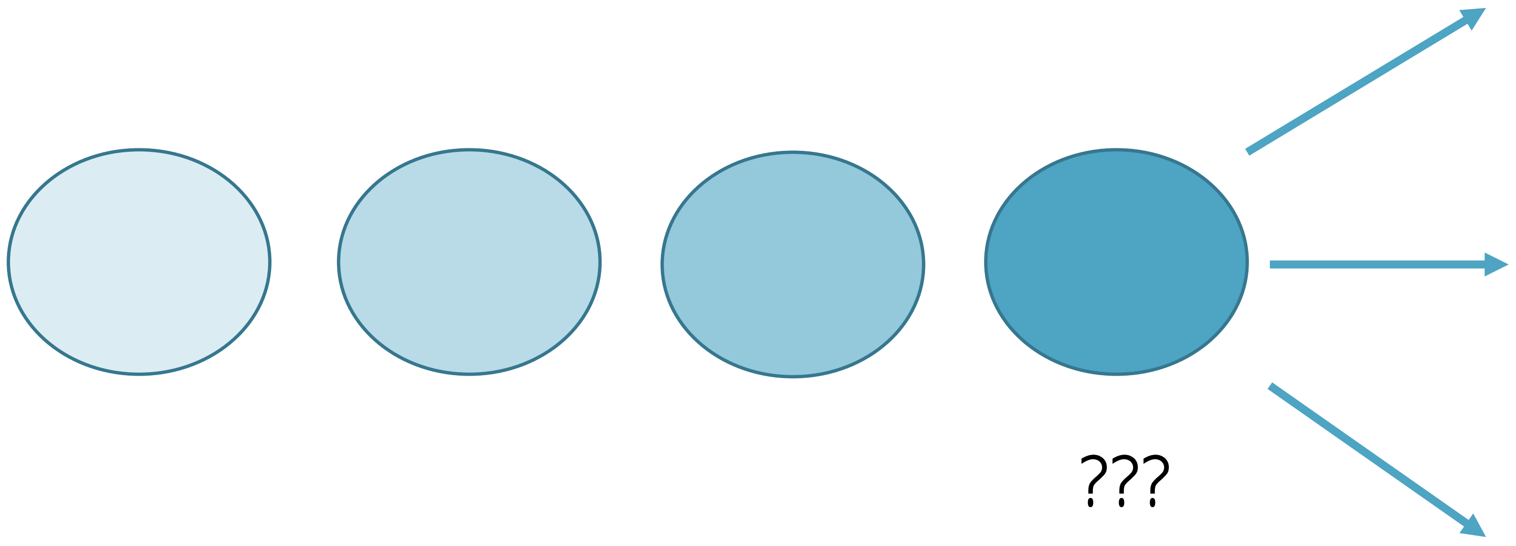
Supongamos que tenemos la imagen de una pelota, podemos predecir a donde se va a mover?



Supongamos que tenemos la imagen de una pelota, podemos predecir a donde se va a mover?



Supongamos que tenemos la imagen de una pelota, podemos predecir a donde se va a mover?



En todos lados hay sequencias de datos

Problema:

Dado un texto, predecir la siguiente palabra

Predecir la siguiente palabra

This morning I took my dog for a walk

Predecir la siguiente palabra

This morning I took my dog for a walk

Datos

Predicción

Predecir la siguiente palabra

This morning I took my dog for a walk

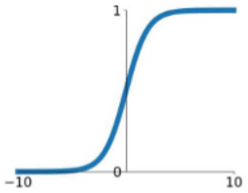
Datos

Predicción

Opción 1: Regresión Logistica

Sigmoid

$$\sigma(x) = \frac{1}{1+e^{-x}}$$



Predecir la siguiente palabra

This morning I took my dog for a walk

Datos

Predicción

Opción 1: Regresión Logistica

Opción 2 : Modelos probabilisticos como Naïve Bayes o LDA

$$P(text) = P(x_0, \dots, x_n) =$$
$$= P(x_0)P(x_1|x_0)P(x_2|x_0, x_1)\dots P(x_n|\dots)$$

Predecir la siguiente palabra

This morning I took my dog for a walk

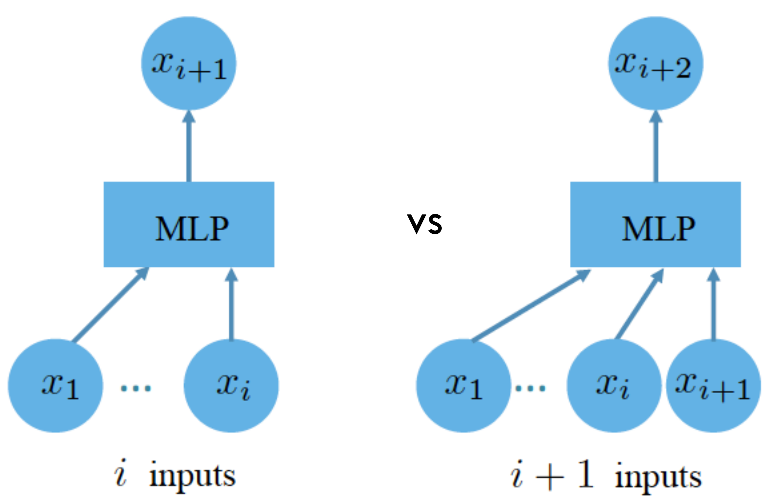
Datos

Predicción

Opción 1: Regresión Logistica

Opción 2 : Modelos probabilisticos como Naïve Bayes o LDA

Opción 3: Feedforward NN



Entonces, qué necesitamos?

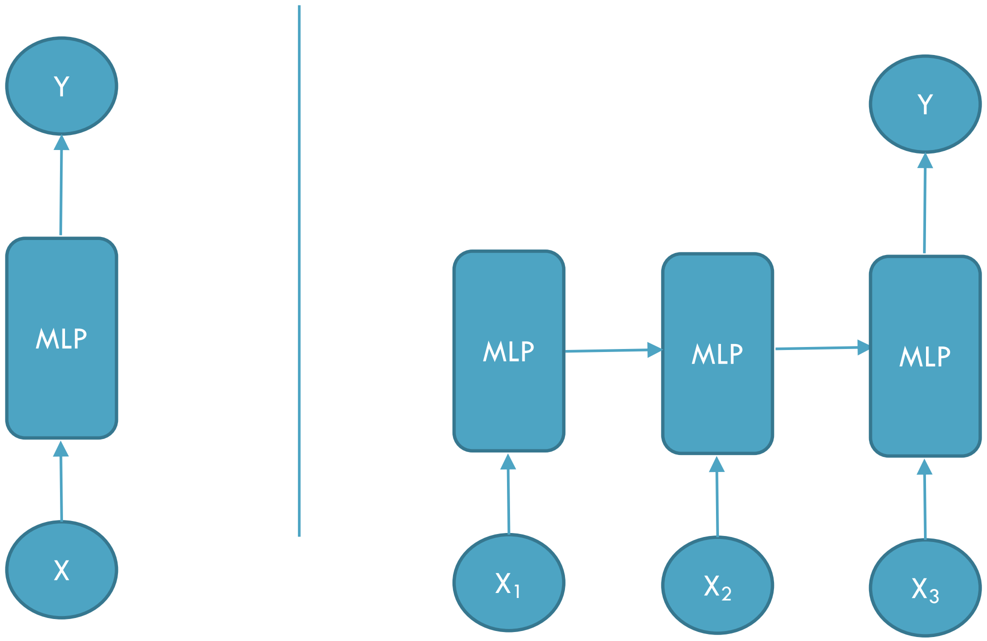
Un modelo (o layer) que pueda:

- Aceptar longitud variable como input
- Capturar dependencias en datos independientemente de su posición
- Compartir parámetros a travez de la sequencia.

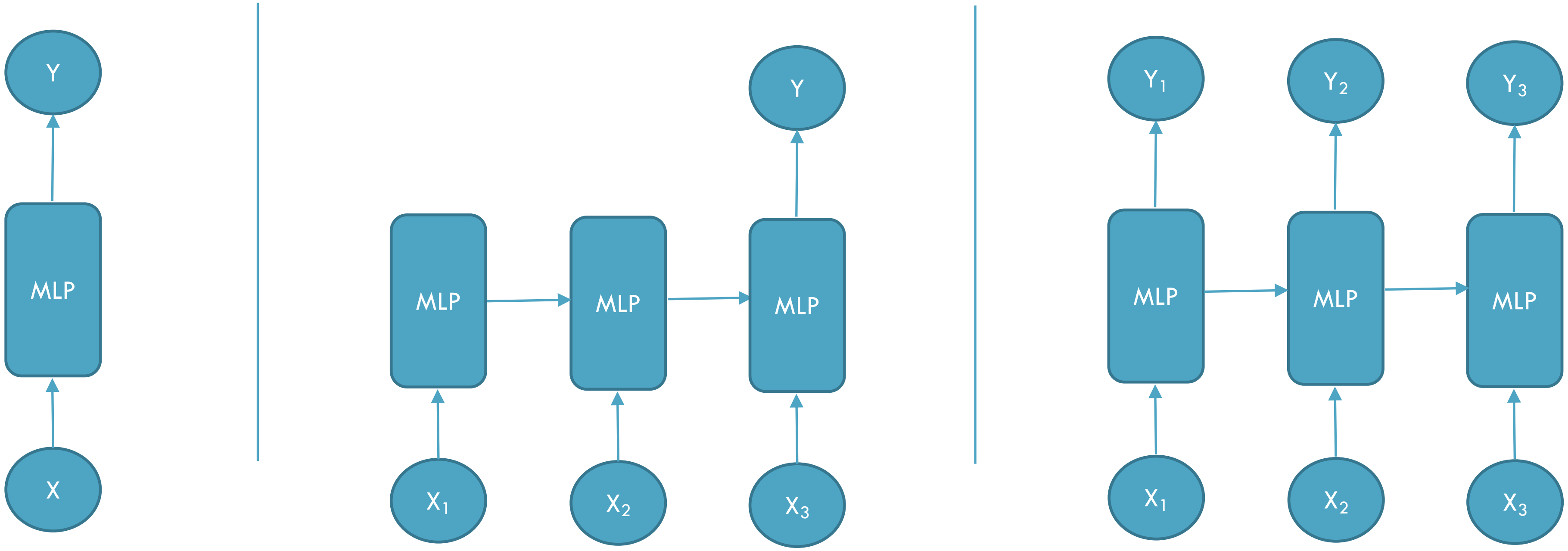
Recurrent Neural Network

Recurrent Layer

Recurrent Layer

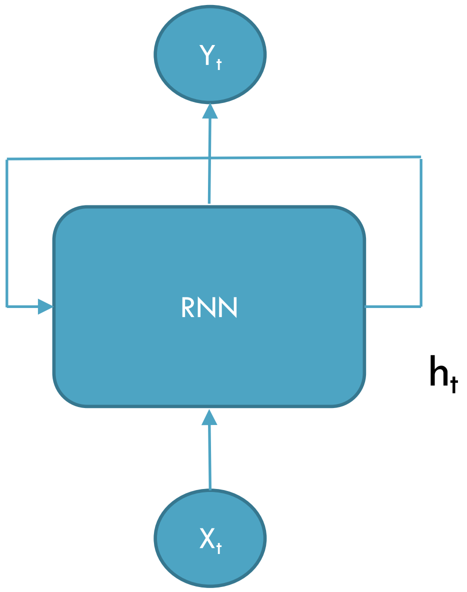


Recurrent Layer

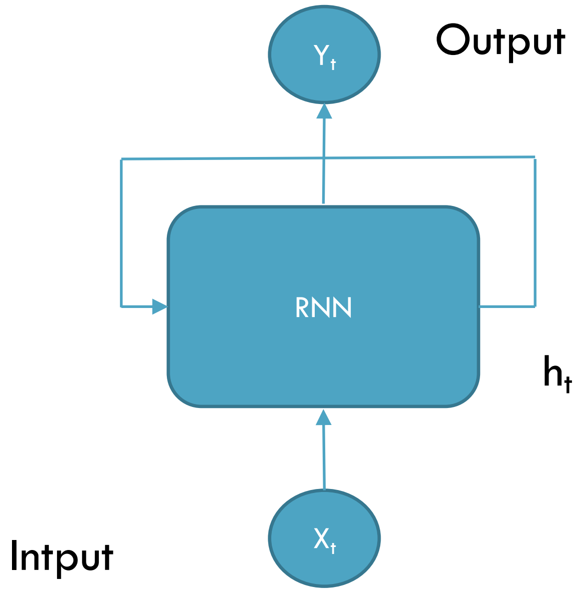


Y muchas otras arquitecturas más…

Recurrent Layer



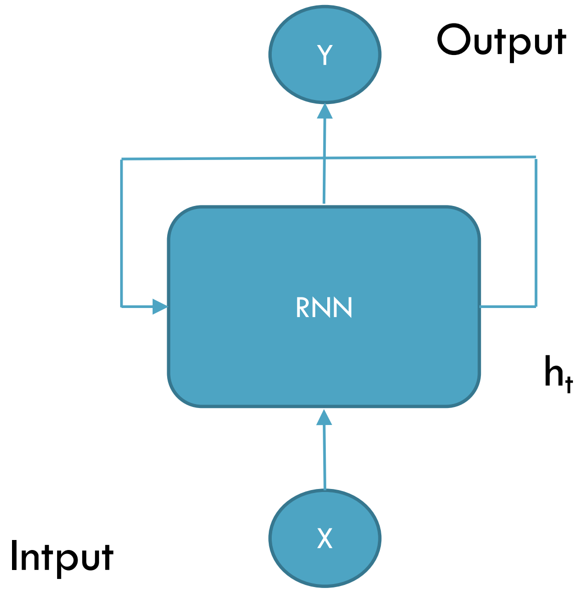
Recurrent Layer



RNN aplican una relación sequencial en cada paso para procesar la sequencia

$$h_t = f_W(h_{t-1}, x_t)$$

Recurrent Layer

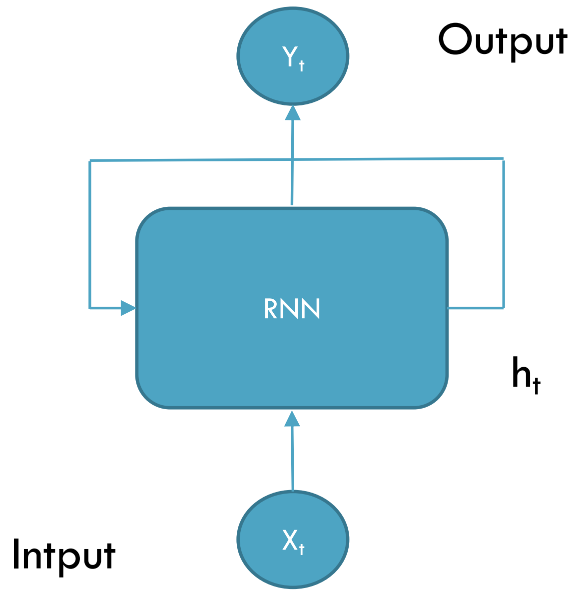


RNN aplican una relación sequencial en cada paso para procesar la sequencia

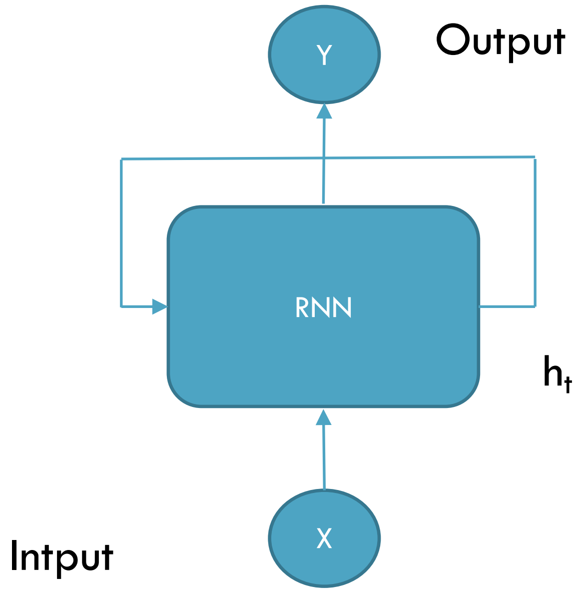
$$\underbrace{h_t}_{\text{Cell state}} = \underbrace{f_W}_{\text{Matriz de pesos W}}(\underbrace{h_{t-1}}_{\text{Cell state anterior}}, \underbrace{x_t}_{\text{Nueva observación}})$$

Recurrent NN intuición

```
my_rnn()

hidden_state = [0 , 0 , 0 , 0]

sentence = [“I” , “love” , “recurrent” , “neural” ]

For word in sentence:
        prediction, hidden_state = my_rnn ( word hidden_state )

Next_word = prediction
```



Entrenamiento

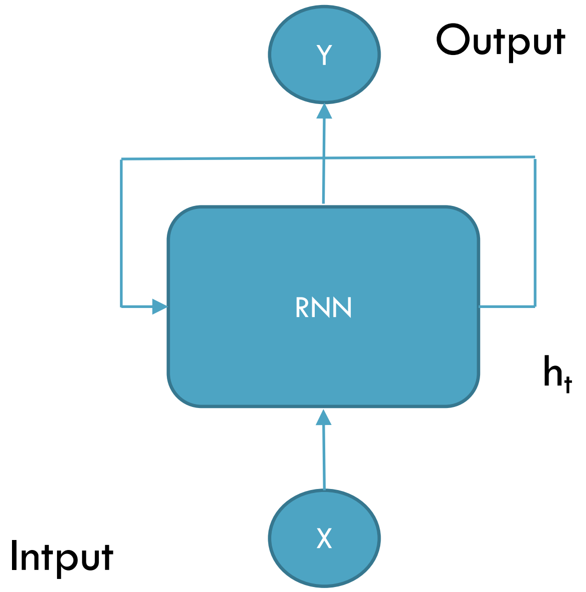


Como se entrena una RNN?

$$h_t = tanh(W^T{}_{hh}h_{t-1}, W^T{}_{xh}x_t)$$

$$Y_t = W^T{}_{hy}h_t$$

Entrenamiento

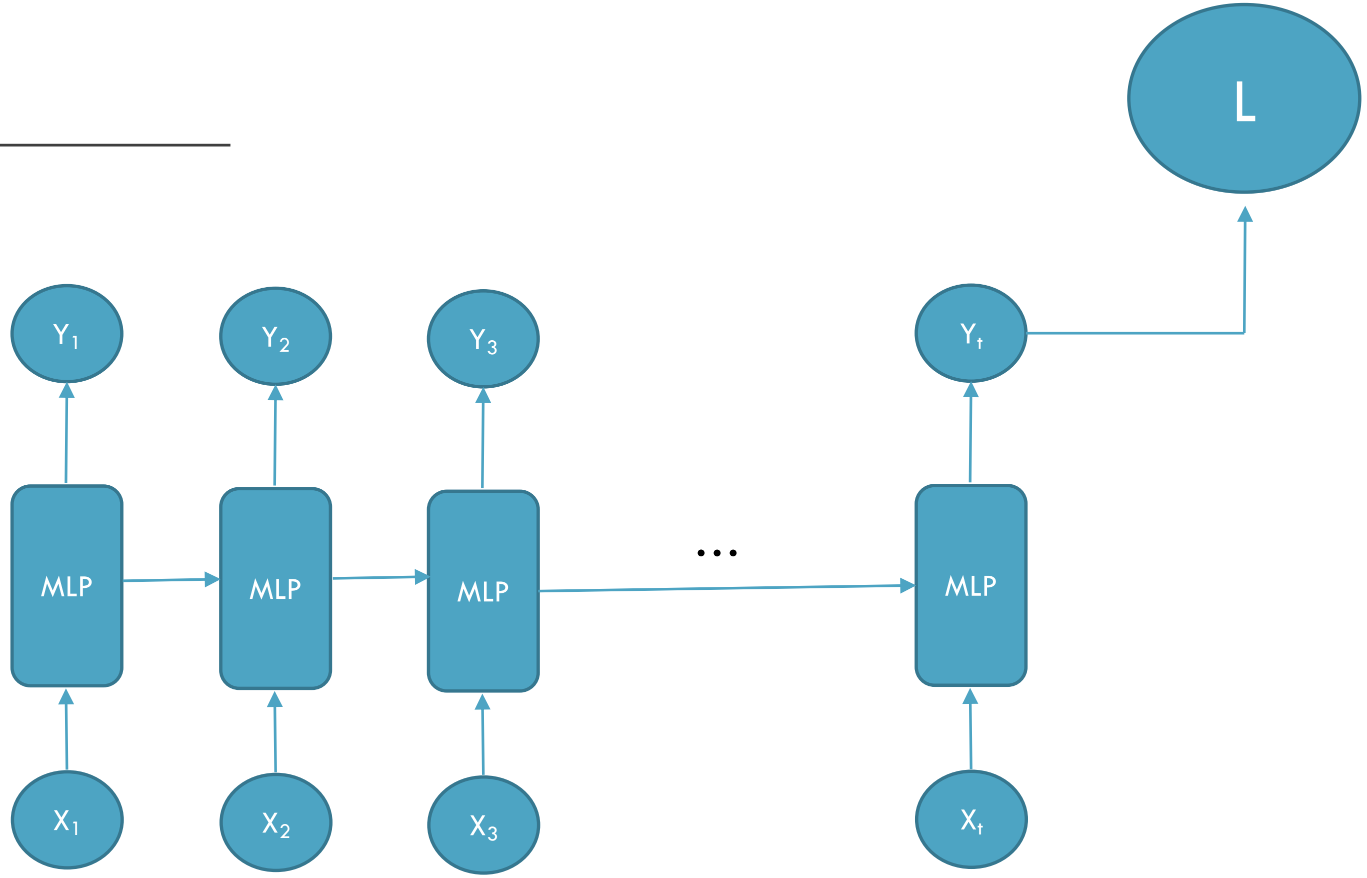


Como se entrena una RNN?

Backpropagation
Through Time

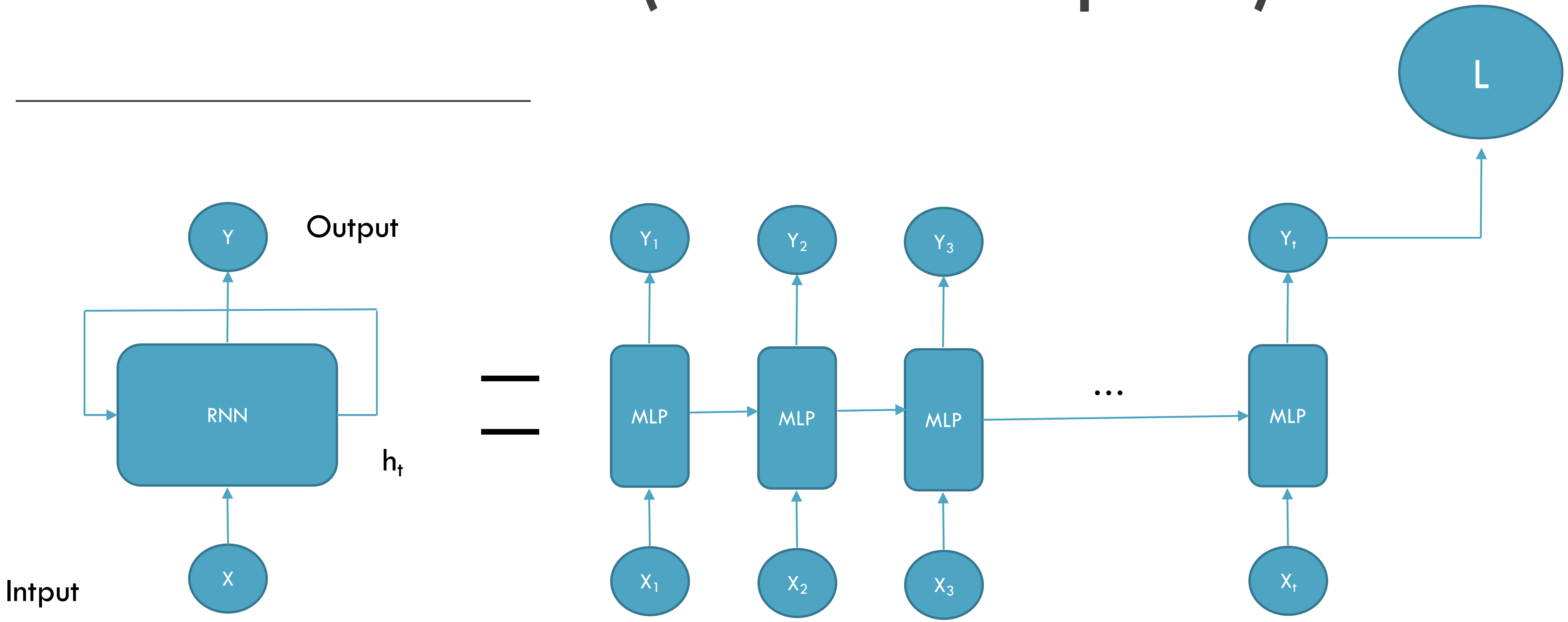
$$h_t = tanh(W^T_{hh}h_{t-1}, W^T_{xh}x_t)$$

$$Y_t = W^T_{hy}h_t$$

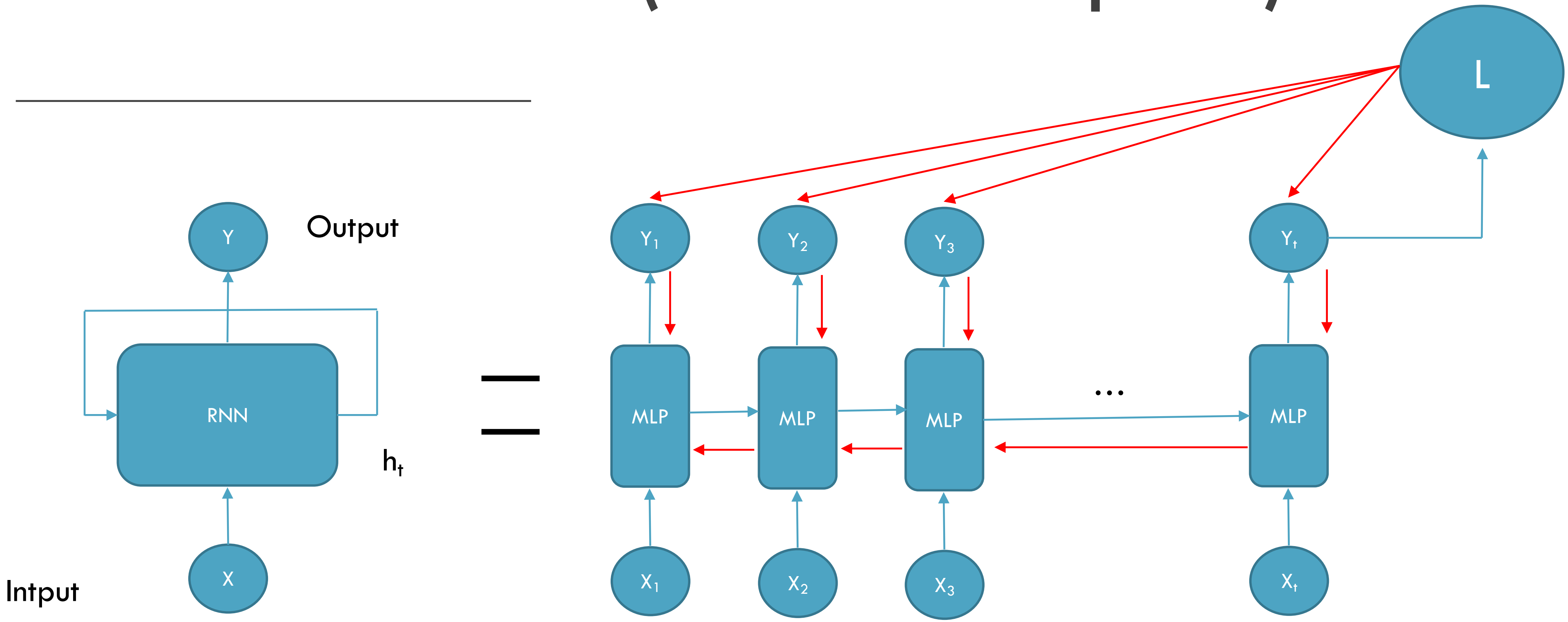


L
Y1
Y2
Y3
Yt
MLP
MLP
MLP
…
MLP
X1
X2
X3
Xt

BBTT intuición (Fordward pass)

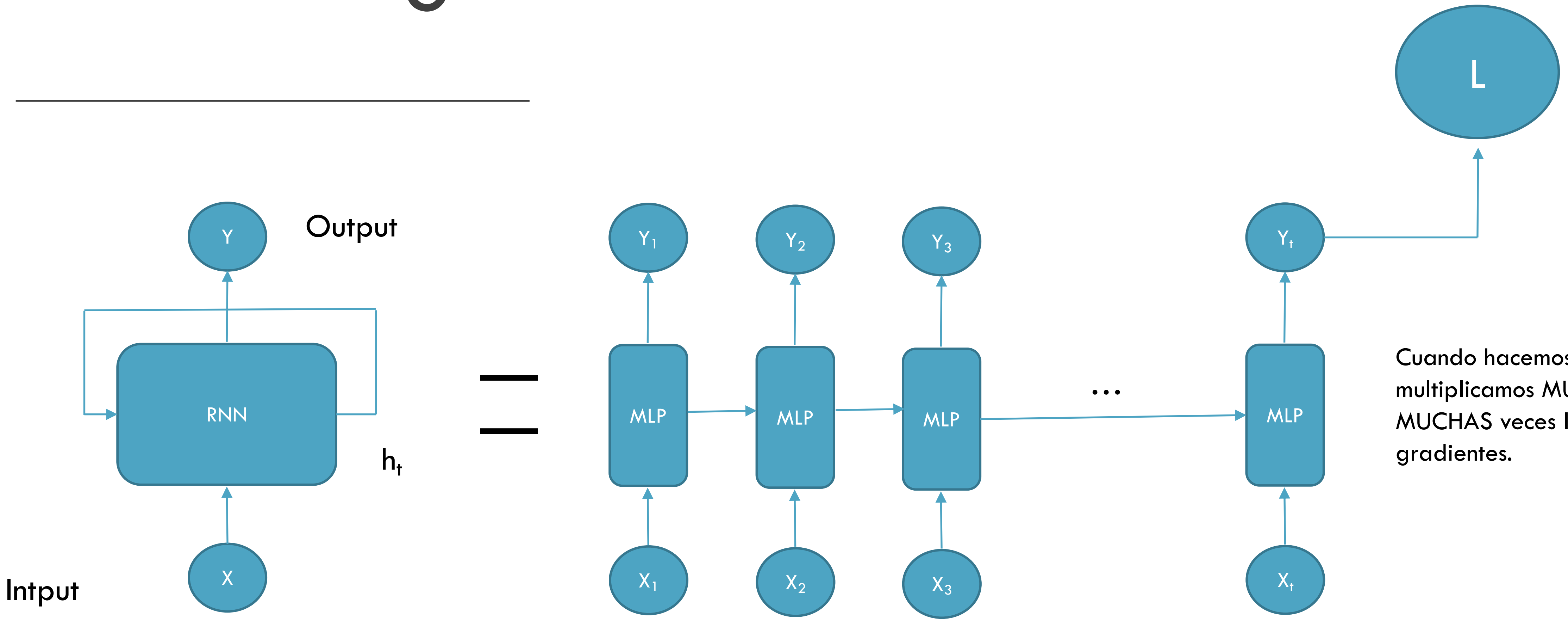


BBTT intuición (Backward pass)



Questions

Vanishing Gradient



Cuando hacemos BPTT multiplicamos MUCHAS MUCHAS veces los gradientes.

Qué pasa con los gradientes?

$$\lim_{n\to\infty} \prod a$$

Si $a < 0$

Si $a > 0$

Qué pasa con los gradientes?

$$\lim_{n \to \infty} \prod a$$

Si a < 0

Si a > 0

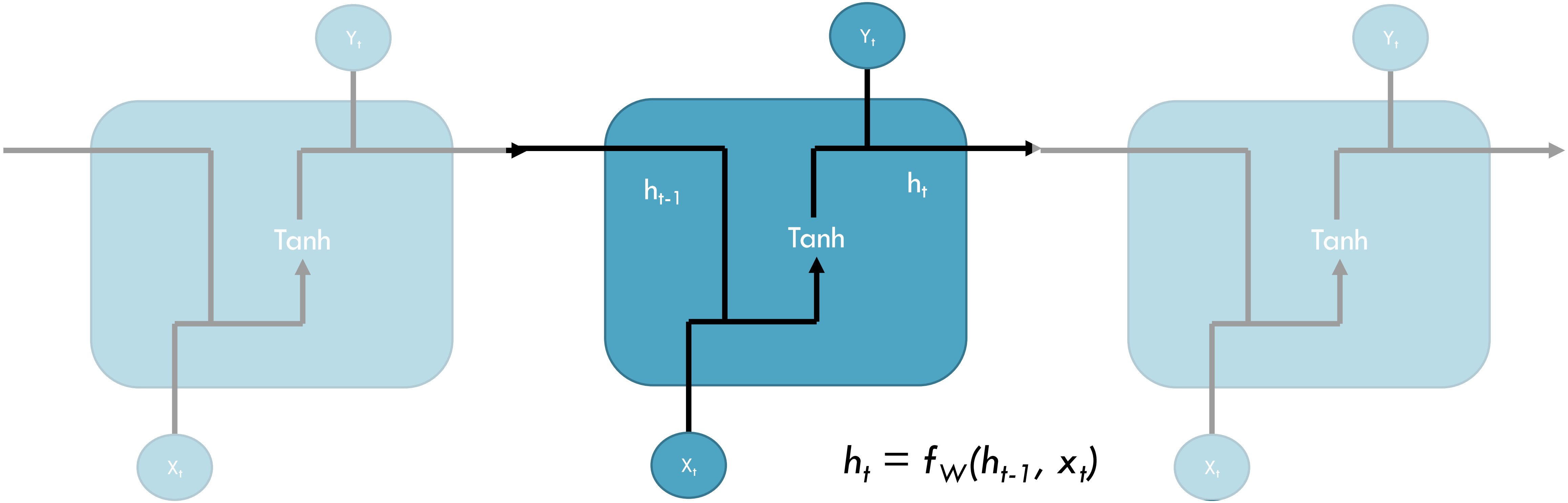
Entonces, necesitamos que los gradientes sean igual (o lo mas similarees posibles) a 1

Dummy solution: utilizar ReLU como función de activación, por qué?

LSTM (Long-Short Term Memory)

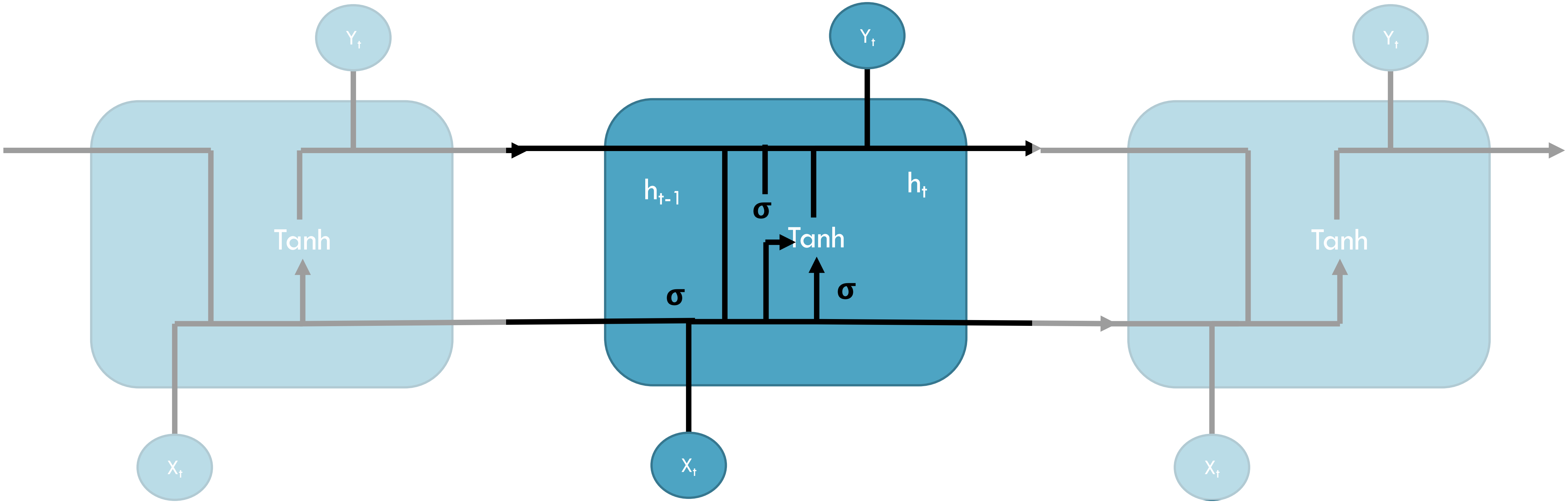
LSTM

LSTM (Long-Short Term Memory)

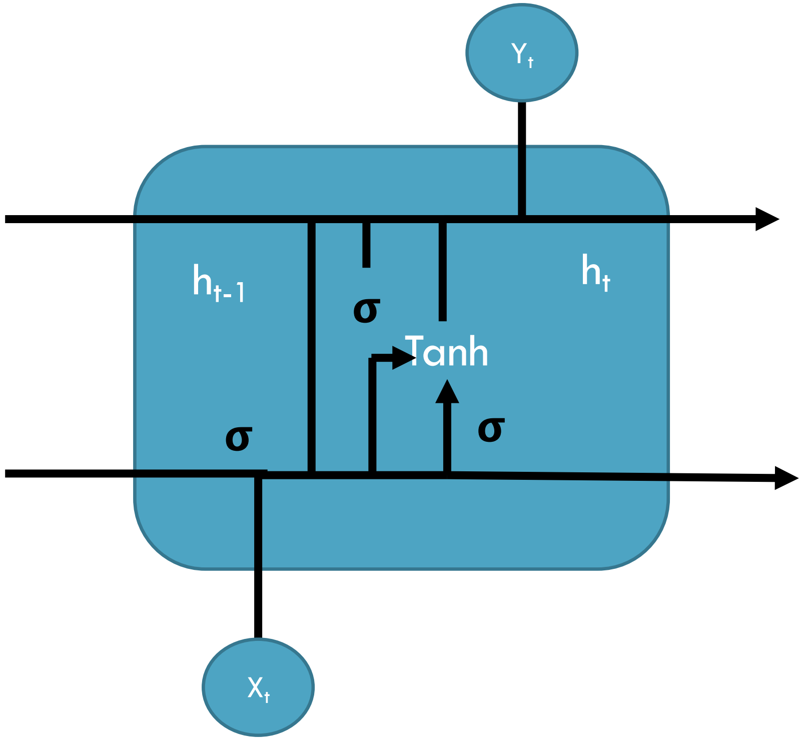


$$h_t = f_W(h_{t-1}, x_t)$$

LSTM (Long-Short Term Memory)



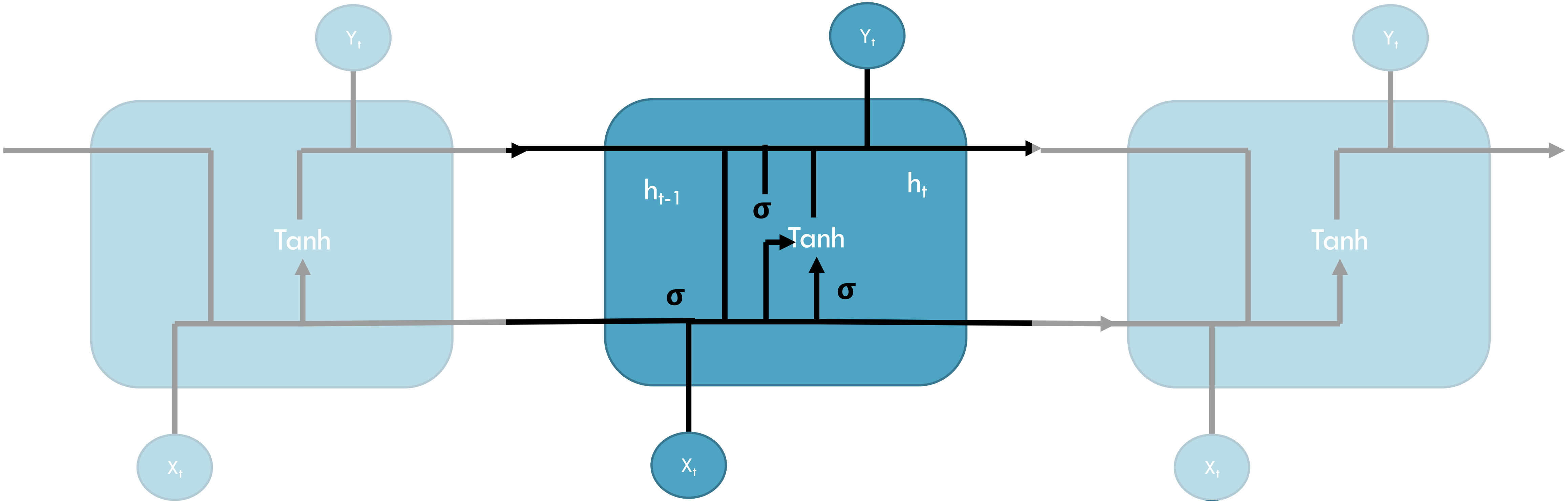
LSTM (Long-Short Term Memory)



LSTM cumplirá 4 diferentes propositos con **σ**:

1. Olvidar (forget)
2. Guardar (store)
3. Actualizar (Update)
4. Predecir (Output)

LSTM (Long-Short Term Memory)



LSTM (RNN) Resumen

1. Mantiene las celdas separadas en cada iteracción
2. Utiliza 4 puertas para controlar la información relevante

Olvida la información irrelevante.
Guarda lo la información importante
Actualiza los pesos de la matríz W
Predice el output final

RNN Aplicaciones

Choping music generated with LSTM:

https://www.youtube.com/watch?v=j60J1cGINX4

RNN Aplicaciones

Choping music generated with LSTM:

https://www.youtube.com/watch?v=j60J1cGINX4

Sentiment Classification

Python Notebook

RNN Aplicaciones

Choping music generated with LSTM:

https://www.youtube.com/watch?v=j60J1cGINX4

Sentiment Classification

Python Notebook

Traducción de textos

	eng	hin
0	give your application an accessibility workout	अपने अनुप्रयोग को पहुंचनीयता व्यायाम का लाभ दें
1	accerciser accessibility explorer	एक्सेर्साइसर पहुंचनीयता अन्वेषक
2	the default plugin layout for the bottom panel	निचले पटल के लिए डिफोल्ट प्लगइन खाका
3	the default plugin layout for the top panel	ऊपरी पटल के लिए डिफोल्ट प्लगइन खाका
4	a list of plugins that are disabled by default	उन प्लगइनों की सूची जिन्हें डिफोल्ट रूप से निष...
5	highlight duration	अवधि को हाइलाइट रकें
6	the duration of the highlight box when selecti...	पहुंचनीय आसंधि नोड को चुनते समय हाइलाइट बक्से ...
7	highlight border color	सीमांत बोर्डर के रंग को हाइलाइट करें
8	the color and opacity of the highlight border	हाइलाइट किए गए सीमांत का रंग और अपारदर्शिता।
9	highlight fill color	भराई के रंग को हाइलाइट करें
10	the color and opacity of the highlight fill	हाइलाइट किया गया भराई का रंग और पारदर्शिता।

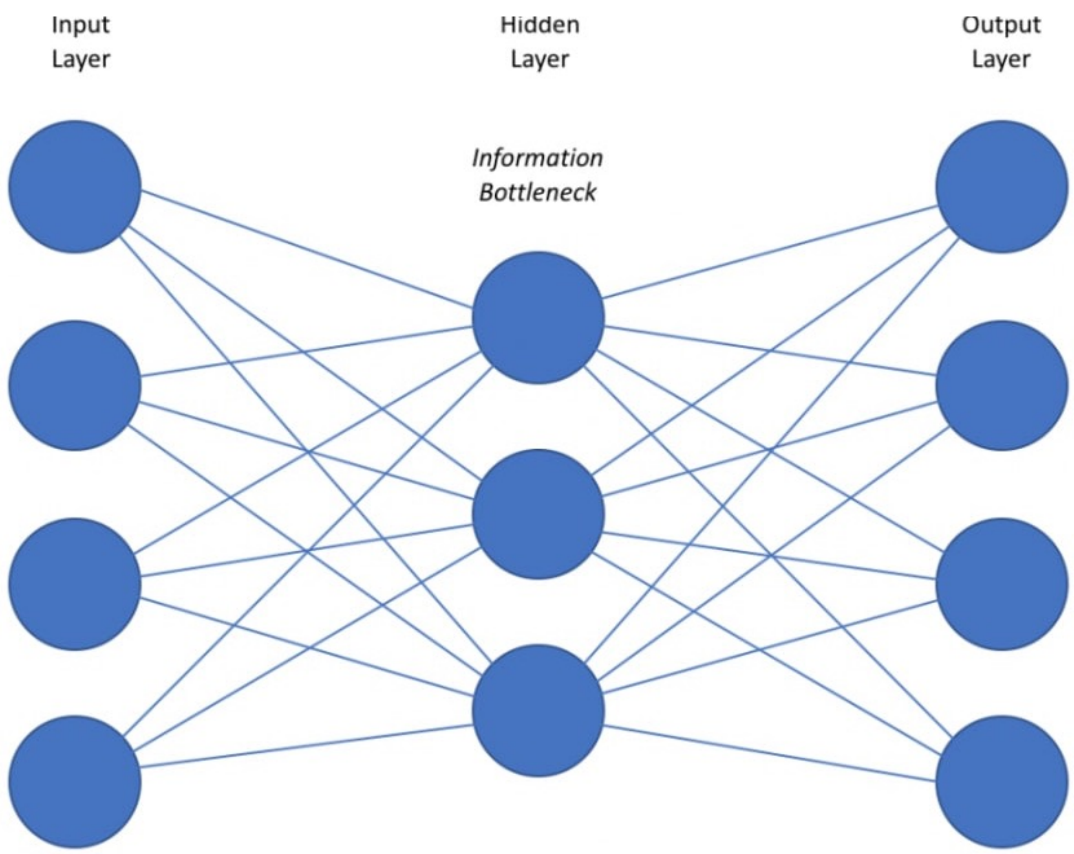
Questions

Autoencoder (Unsupervised DL)

Idea:

Tomar el input, pasarlo por una estructura de red neuronal y despues reconstruirlo

Decoder(Encoder(x)) = X



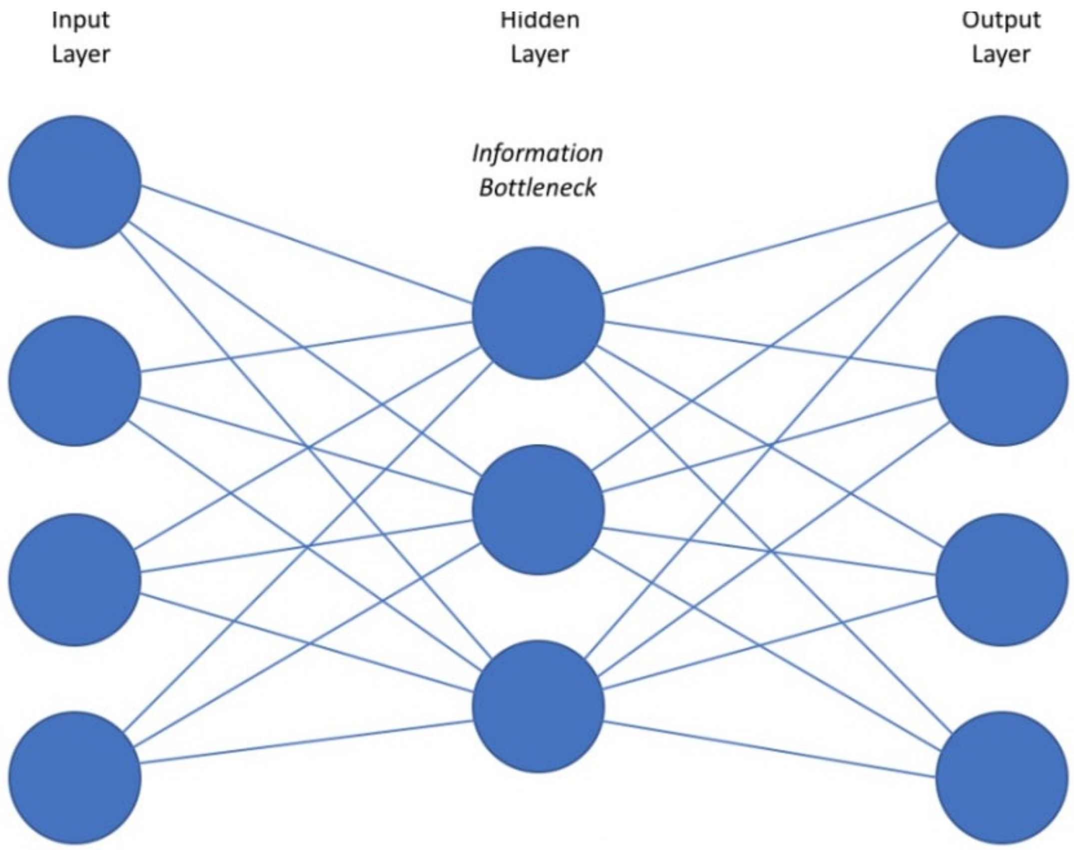
Encoder → Representation Layer → Decoder

Autoencoder (Unsupervised DL)

Idea:

Tomar el input, pasarlo por una estructura de red neuronal y despues reconstruirlo

Decoder(Encoder(x)) = X

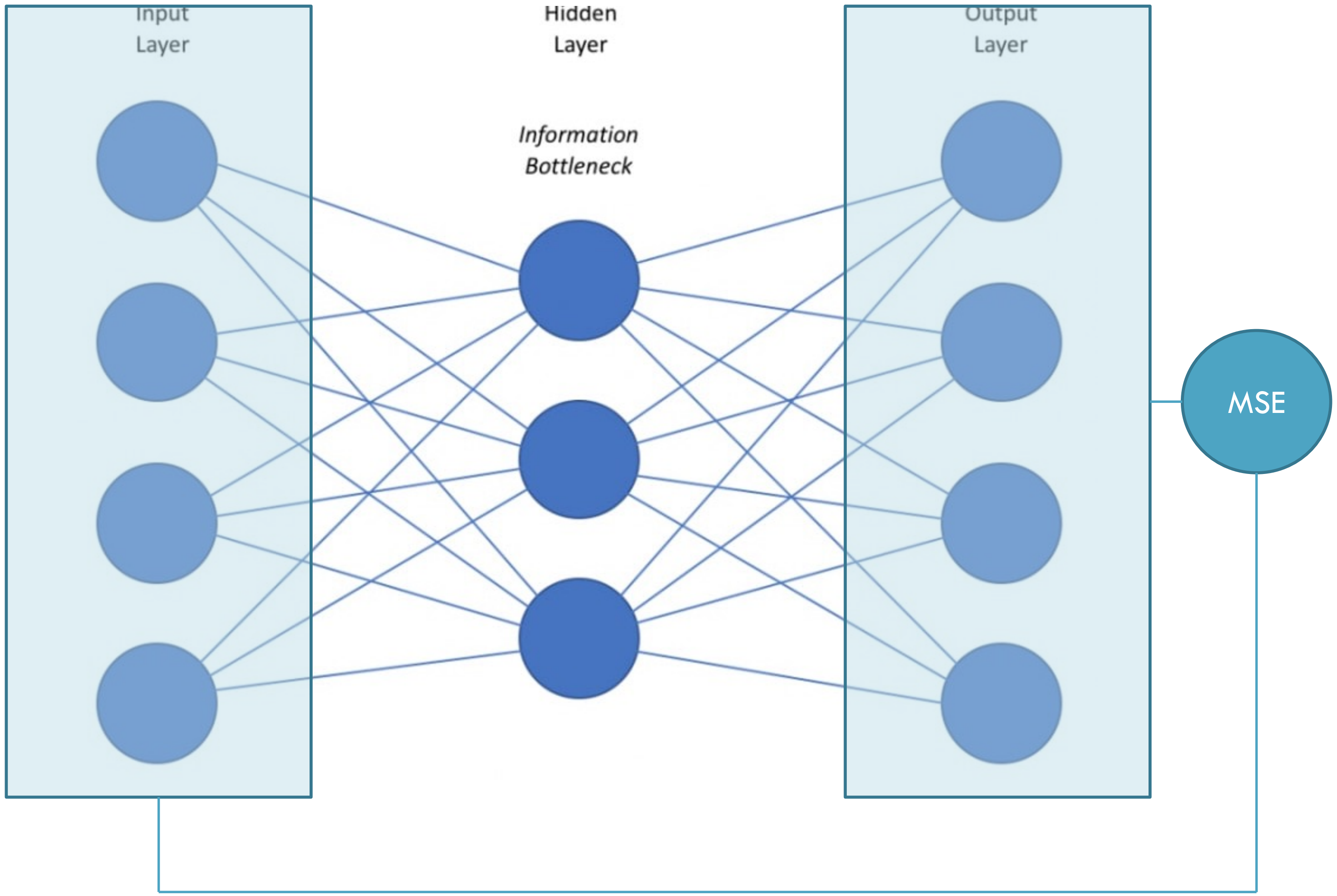


Encoder → Representation Layer → Decoder

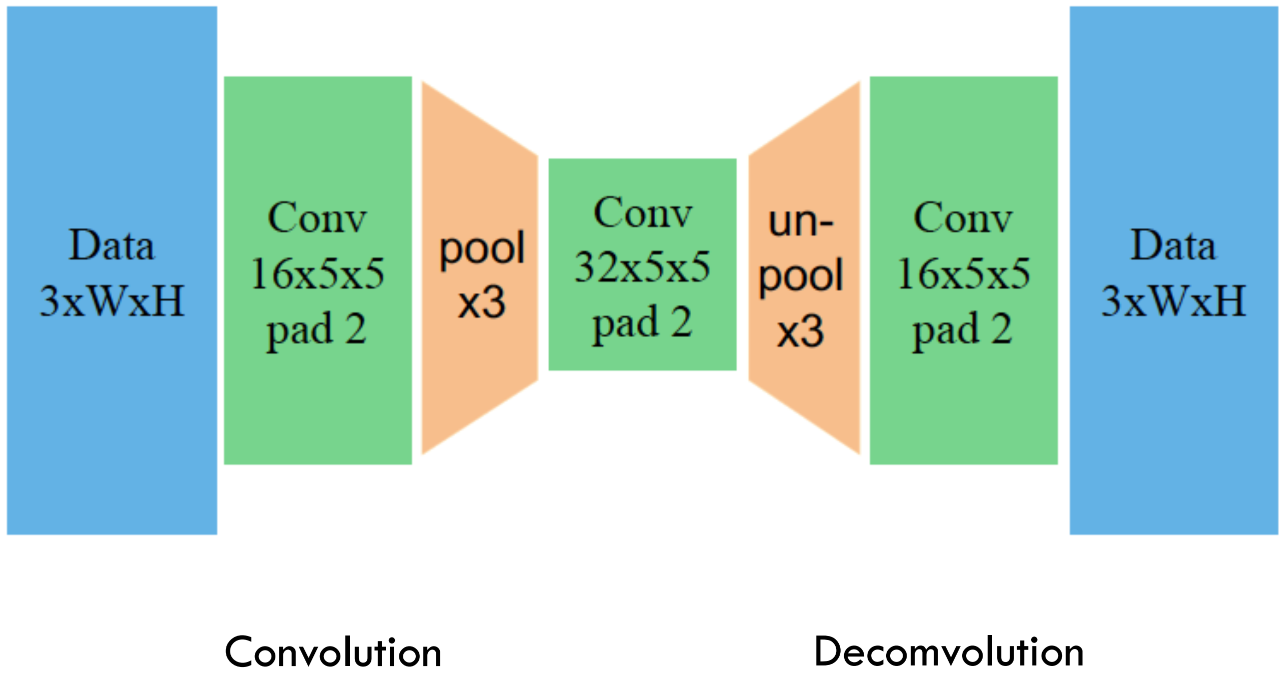
Cómo se entrena?

Backpropagation!

Minimizando el MSE entre el input y el output.

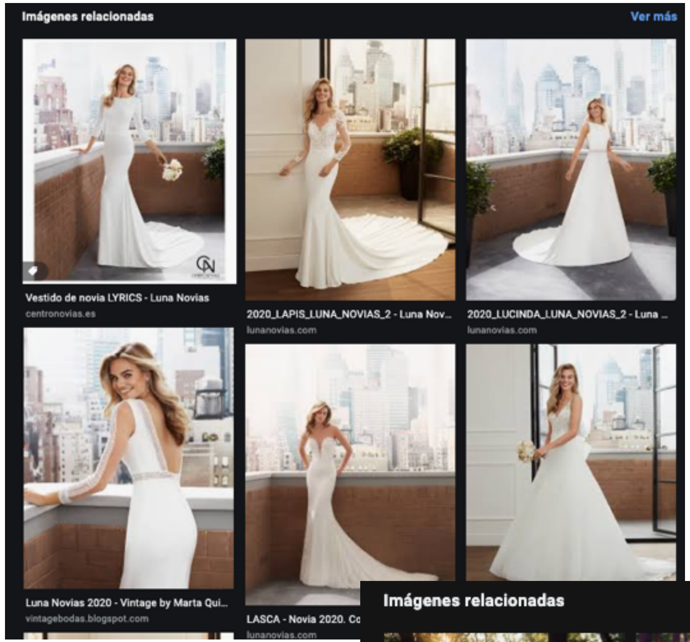


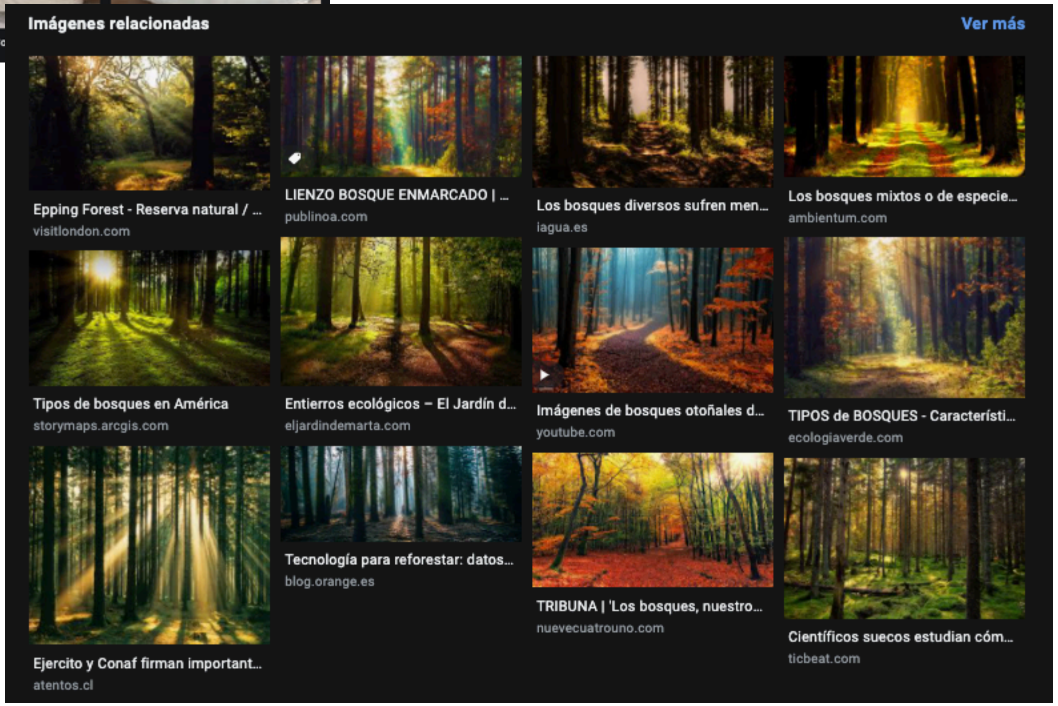
Qué pasa si el input es una imagen?



Autoencoder Aplicaciones

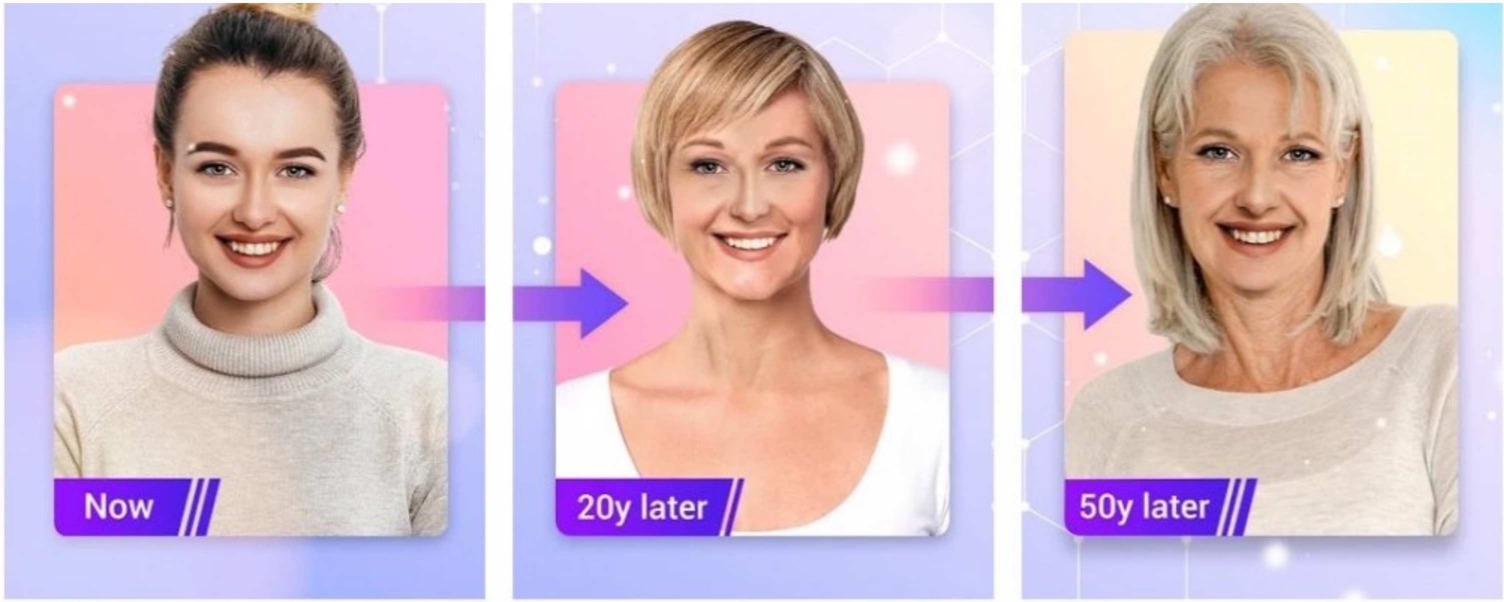
Image Clustering





Autoencoder
Aplicaciones

Style mixing



Tipos de Autoencoders

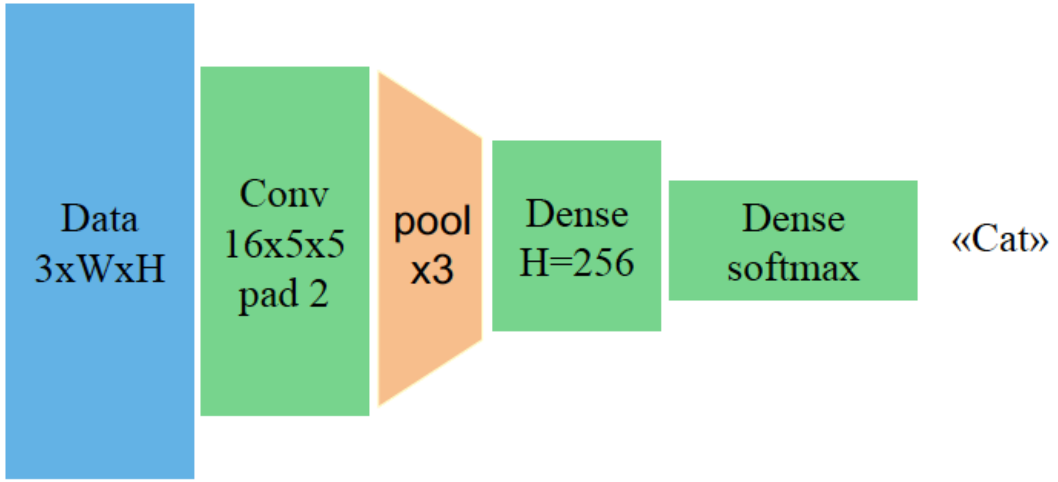
Podemos utilizar diferentes pre-entrenados AE

Sparse Autoencoder $L = \|X - Dec(Enc(X))\| + \sum_i |Enc_i(X)|$

variational Autoencoder $L = \left\|X - Enc\left(Dec(Noize(X))\right)\right\|$

Expanding Autoencoder $L = \|X - Dec(Enc(X))\|$

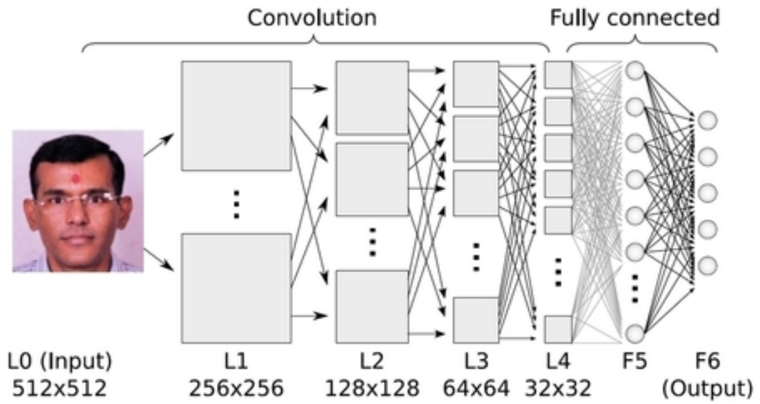
Denoizing Autoencoder $L = \left\|X - Enc\left(Dec(Noize(X))\right)\right\|$



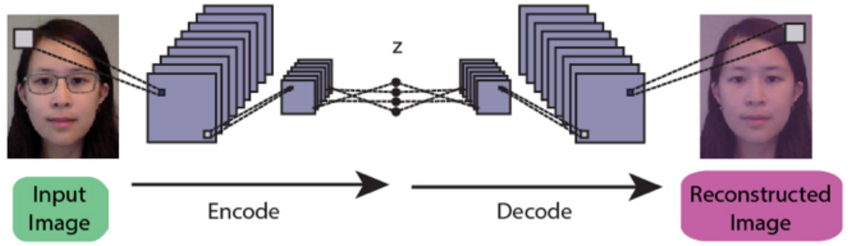
Generative Adversarial models 101

Objetivo:
Queremos generar nuevas imagenes “Falsas” que parezcan reales, vamos a dejarlo en rostros de personas.

CNN puede classificar imagenes de rostros



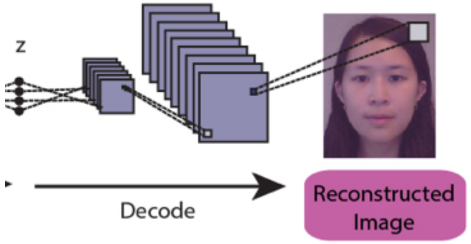
Autoencoder puede descomponer y reconstruir imagenes de rostros

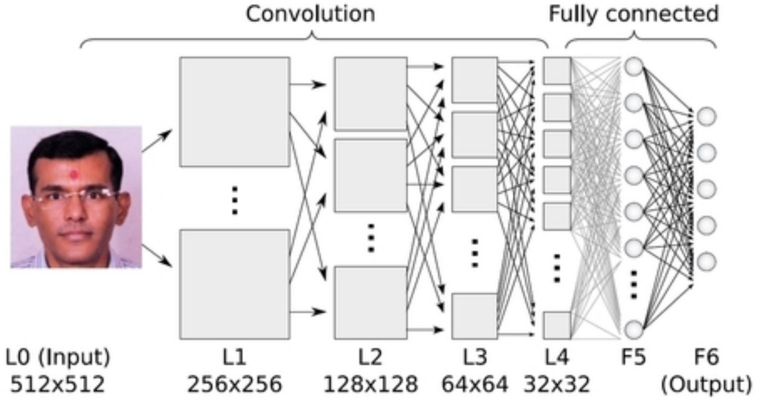


Objetivo:
Queremos generar nuevas imagenes “Falsas” que parezcan reales, vamos a dejarlo en rostros de personas.

CNN puede classificar
imagenes de rostros

Muestreo
aleatorio de
n objetos

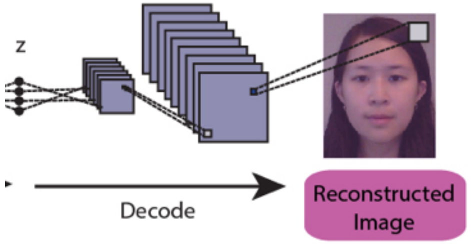


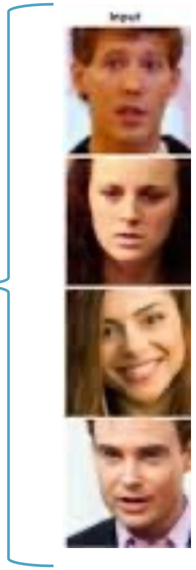


MSE

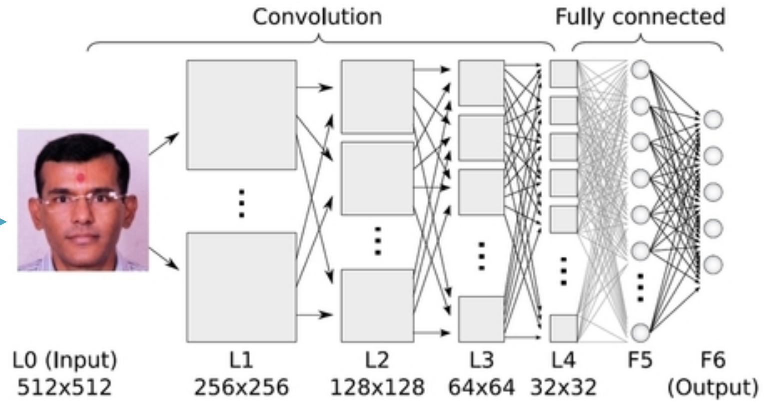
Objetivo:
Queremos generar nuevas imagenes "Falsas" que parezcan reales, vamos a dejarlo en rostros de personas.

Muestreo aleatorio de n objetos





CNN puede classificar imagenes de rostros



MSE

Una vez que este proceso converge, nuestro sampler se comvierte en un generador de rostros capaz de engañar a una red neuronal

Questions

Que vimos hoy?

Calentamiento

- SGD
- MLP y Backpropagation
- FNN
- NLP y sentiment Analysis

Computer vision

- Convoluciones
- Feature extraction
- Image classification

Arquitecturas avanzadas

- RNN
- Autoencoders
- GANs

La punta del iceberg

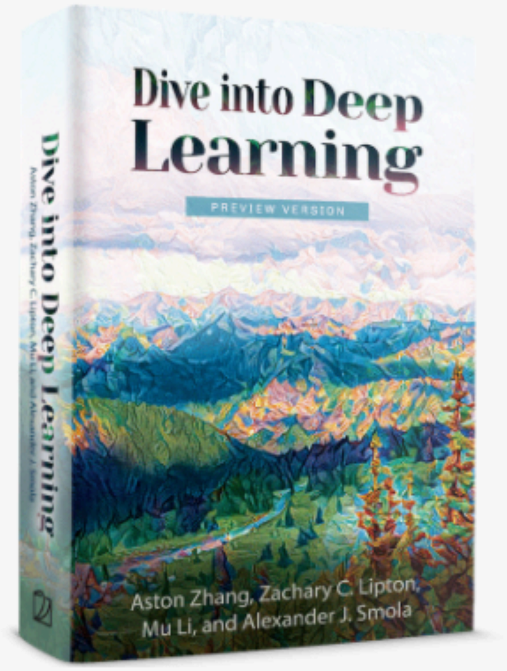
Otros paradígmas

Semi-supervised Learning

Transfer Learning

Reinforcement Learning

Recursos Adicionales



Dive into Deep Learning

An interactive deep learning book with code, math, and discussions

Provides NumPy/MXNet, PyTorch, and TensorFlow implementations

https://d2l.ai/

FREE COURSE

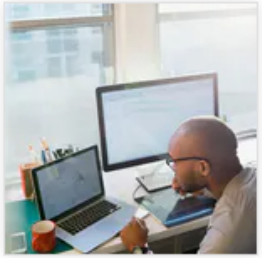
Intro to TensorFlow for Deep Learning

by TensorFlow

This course is a practical approach to deep learning for software developers

START FREE COURSE

coursera



Machine Learning

University of Washington

Especialización

4.6 (13,927) | 380K estudiantes | PLUS

Intermediate

Advanced Machine Learning

National Research University Higher School of Economics

Especialización

4.5 (3,482) | 290K estudiantes | PLUS

Advanced

https://www.udacity.com/course/intro-to-tensorflow-for-deep-learning--ud187

https://www.coursera.org/specializations/machine-learning
https://www.coursera.org/specializations/aml

Alexander Amini
75,900 suscriptores

Abhishek Thakur
32,000 suscriptores

TensorFlow
290,000 suscriptores

DeepMind
251,000 suscriptores

stanfordonline
201,000 suscriptores

Thank you