Sistemas Inteligentes para la Gestión en la Empresa

Técnicas de visualización en *Deep Learning*

Felipe Peiró

Juan Carlos Serrano

Pedro Manuel Gómez-Portillo

Índice

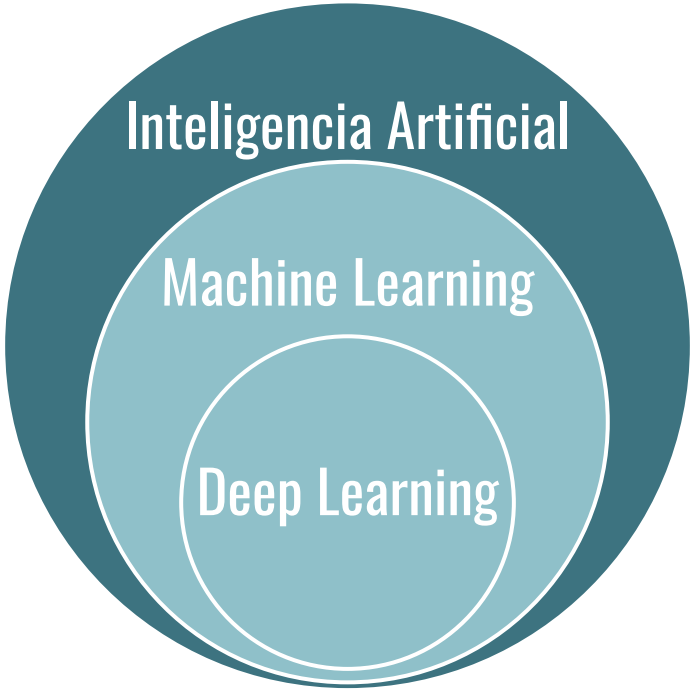
- **Introducción.**
- **Soluciones disponibles.** Playground, TensorBoard, PlotNeuralNet, NN-SVG, Keras.js, Netron
- **Conclusiones.**

Introducción

Introducción I

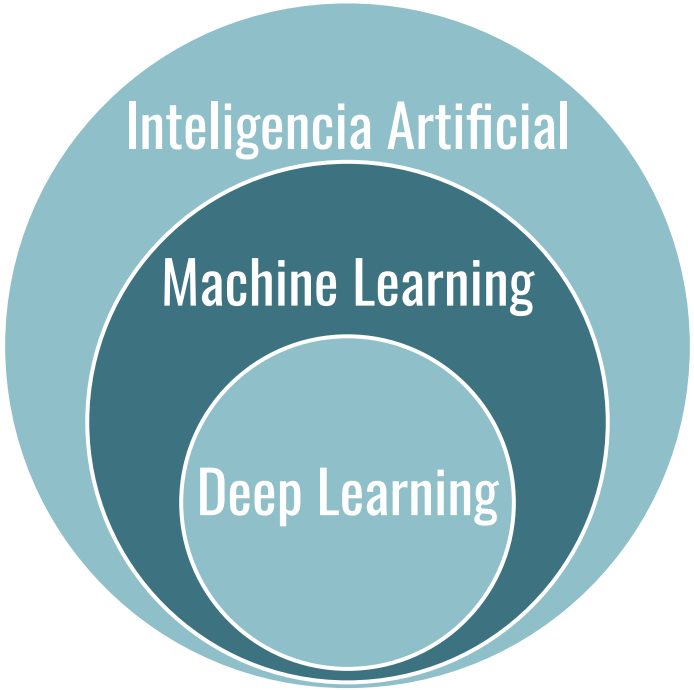
- El *Deep Learning* es muy popular
- Muy bueno para tareas relacionadas con los sentidos
  - Natural para los humanos pero no para las máquinas
    - Procesamiento de Lenguaje Natural
    - Visión por Computador

Introducción II



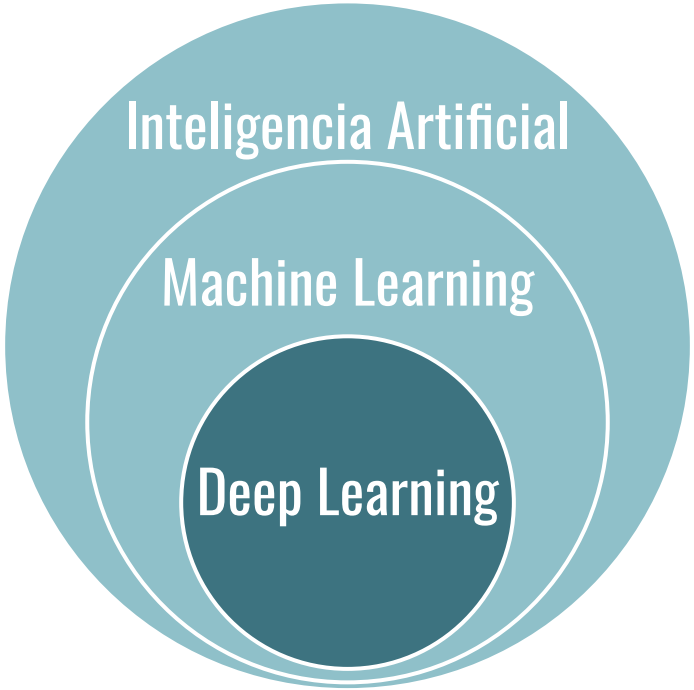
- Inteligencia Artificial
  - Años 50
  - Algoritmos de búsqueda
  - Razonamiento simbólico
  - Razonamiento lógico

Introducción II



- Machine Learning
  - Sin programación específica
  - Ap. supervisado, ap. no supervisado y Deep Learning

Introducción II



- Deep Learning
  - Redes neuronales artificiales similares al cerebro humano

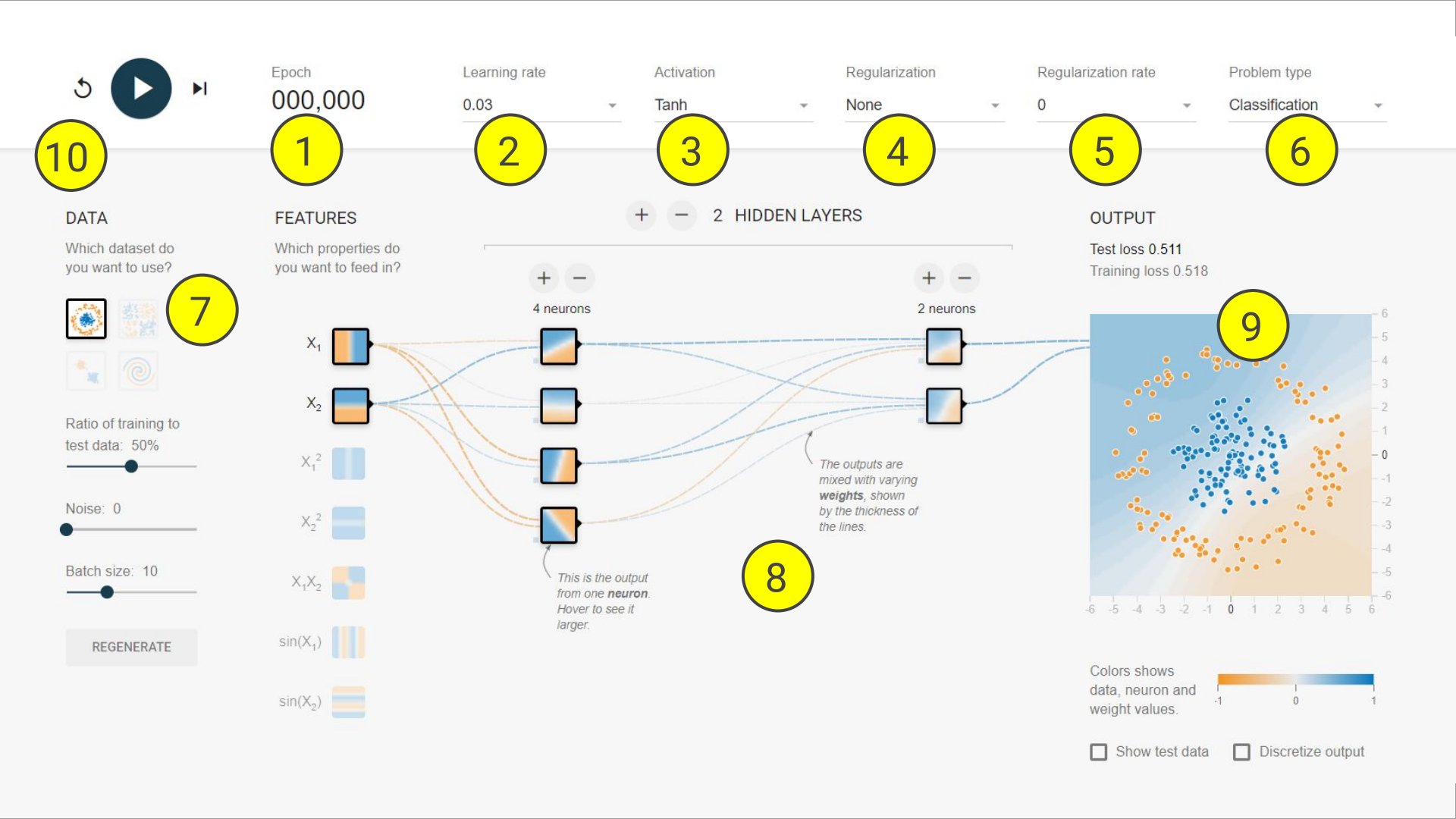
Introducción III - Deep Learning

- TensorFlow, Keras, Pytorch, Stickit-learn, Lasagne, DSSTINE, MXNet, DL4J, Microsoft Cognitive Toolkit
- **Democratización** → todo el mundo tiene acceso
  - Ayudar a visualizar y entender
- Alternativas y aplicación a la práctica

Soluciones disponibles

Playground I

- Herramienta de TensorFlow
- Centrada en ser didáctica
  - Nombre → *jugar*
- No permite subir datos propios


Epoch
000,000
Learning rate
0.03
Activation
Tanh
Regularization
None
Regularization rate
0
Problem type
Classification
10
1
2
3
4
5
6
DATA
Which dataset do you want to use?
7
Ratio of training to test data: 50%
Noise: 0
Batch size: 10
REGENERATE
FEATURES
Which properties do you want to feed in?
X1
X2
X1²
X2²
X1X2
sin(X1)
sin(X2)
2 HIDDEN LAYERS
4 neurons
2 neurons
This is the output from one neuron. Hover to see it larger.
The outputs are mixed with varying weights, shown by the thickness of the lines.
8
OUTPUT
Test loss 0.511
Training loss 0.518
9
Colors shows data, neuron and weight values.
-1
0
1
Show test data
Discretize output

PlayGround III

Demo

https://playground.tensorflow.org/

TensorBoard I

- Integrada en TensorFlow
  - Facilitar a sus usuarios entender y optimizar sus redes neuronales
- Pueden usarse datos propios

TensorBoard II - Uso

Código en R

```
callback_tensorboard("path/to/logs/") #fit()

[ … ]

tensorboard("path/to/logs/")
```

TensorBoard SCALARS GRAPHS INACTIVE

Show data download links

Ignore outliers in chart scaling

Tooltip sorting method: default

Smoothing

0,6

Horizontal Axis

STEP RELATIVE WALL

Runs

Write a regex to filter runs

logs/.

TOGGLE ALL RUNS

logs:./logs

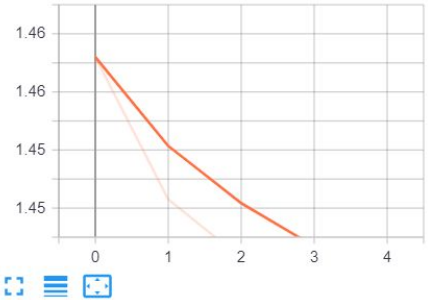
Filter tags (regular expressions supported)

epoch_acc 1

epoch_acc

epoch_loss 1

epoch_loss



TensorBoard SCALARS GRAPHS INACTIVE

Search nodes. Regexes supported.

Fit to Screen

Download PNG

Run (1) logs/.

Session runs (0)

Upload Choose File

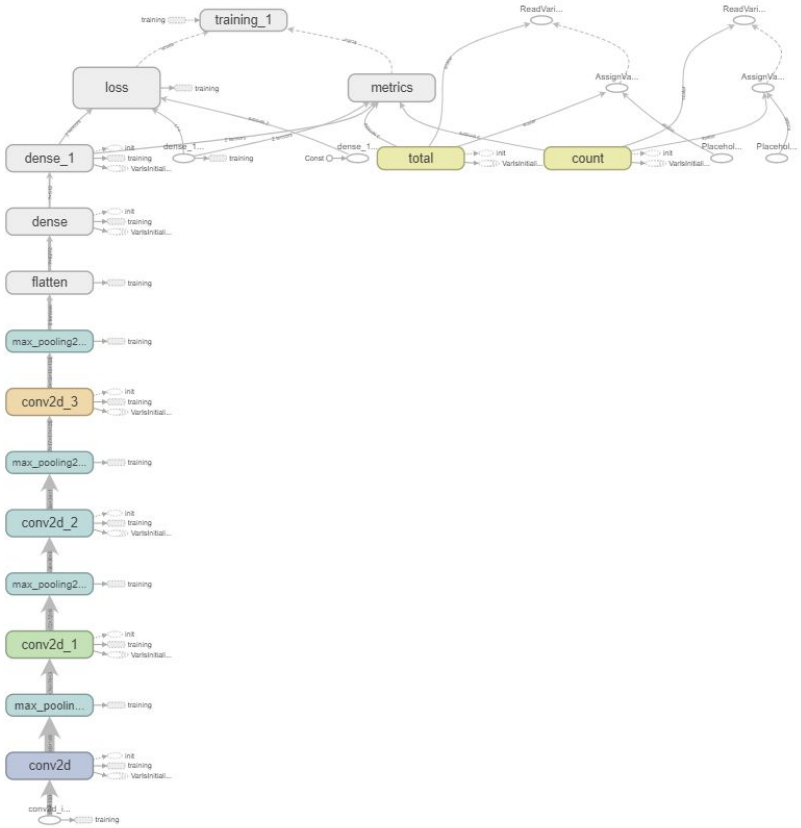
Trace inputs

Color
- Structure
- Device
- XLA Cluster
- Compute time
- Memory
- TPU Compatibility

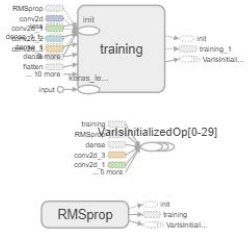
Close legend.

Graph (* = expandable)
- Namespace* ?
- OpNode ?
- Unconnected series* ?
- Connected series* ?
- Constant ?
- Summary ?
- Dataflow edge ?
- Control dependency edge ?
- Reference edge ?

Main Graph



Auxiliary Nodes



PlotNeuralNet I

- Paquete LaTeX
- Permite generar diagramas de redes neuronales en PDF
  - Perfecta para informes
- Relativamente complicado y lioso

PlotNeuralNet II - Uso

- Importar la librería al proyecto
- Comenzar con `\begin{tikzpicture}`
- Capas, conexiones, líneas

PlotNeuralNet III - Ejemplo

Demo

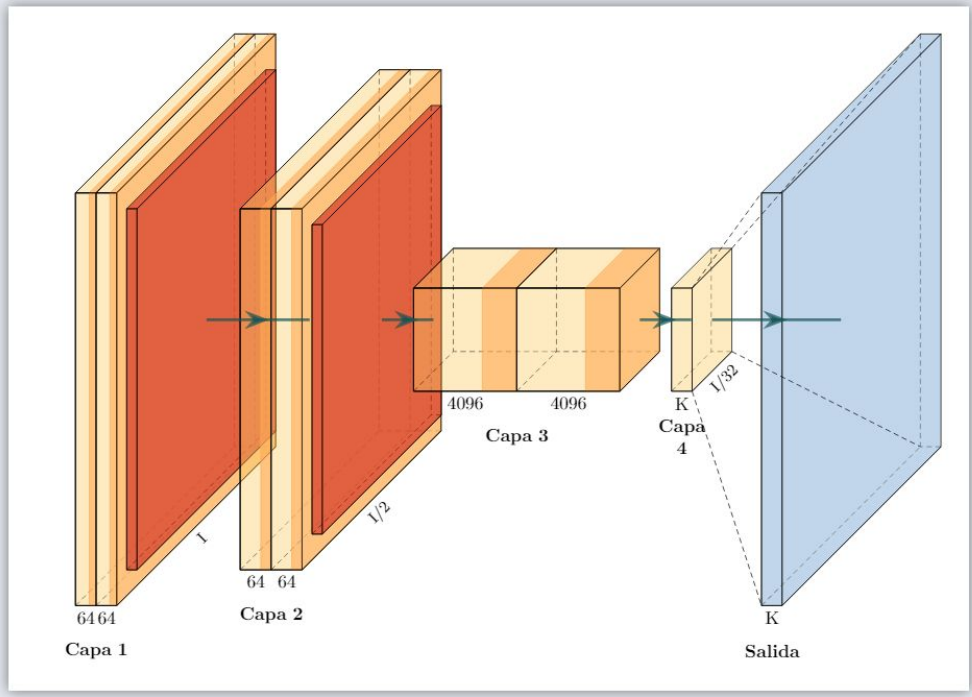
https://www.overleaf.com/project/5cf93dab9e01b27a8fb981b5

Menu | PlotNeuralNet-ejemplo | Review | Share | Submit | History | Chat

Source | Rich Text

Recompile

```
\documentclass[border=15pt, multi, tikz]{standalone}
\usepackage{import}
\subimport{../../layers/}{init}
\usetikzlibrary{positioning}

\def\ConvColor{rgb:yellow,5;red,2.5;white,5}
\def\ConvReluColor{rgb:yellow,5;red,5;white,5}
\def\PoolColor{rgb:red,1;black,0.3}
\def\DcnvColor{rgb:blue,5;green,2.5;white,5}
\def\SoftmaxColor{rgb:magenta,5;black,7}

\begin{document}

\begin{tikzpicture}
\tikzstyle{connection}=[ultra thick,every node/.style={sloped,allow upside down},draw=\edgecolor,opacity=0.7]
% ------------------- Capas
% Capa 1
\pic[shift={(0,0,0)}] at (0,0,0) {RightBandedBox={name=cr1,caption=Capa 1,
        xlabel={{"64","64"}},zlabel=I,fill=\ConvColor,bandfill=\ConvReluColor,
        height=40,width={2,2},depth=40}};
% Pool 1
\pic[shift={(0,0,0)}] at (cr1-east) {Box={name=p1,
        fill=\PoolColor,opacity=0.5,height=35,width=1,depth=35}};

% Capa 2
\pic[shift={(2,0,0)}] at (p1-east) {RightBandedBox={name=cr2,caption=Capa 2,
        xlabel={{"64","64"}},zlabel=I/2,fill=\ConvColor,bandfill=\ConvReluColor,
        height=35,width={3,3},depth=35}};
% Pool 2
\pic[shift={(0,0,0)}] at (cr2-east) {Box={name=p2,%
        fill=\PoolColor,opacity=0.5,height=30,width=1,depth=30}};

% Capa 3
\pic[shift={(1,0,0)}] at (p2-east) {RightBandedBox={name=cr6_7,caption=Capa 3,%
        xlabel={{"4096","4096"}},fill=\ConvColor,bandfill=\ConvReluColor,%
```



NN-SVG I

- Herramienta web
- Generar manualmente diagramas de redes neuronales
- Muy sencillo y parametrizable
- 3 estilos → esferas, 2D y 3D

NN-SVG II - Demo

Demo

http://alexlenail.me/NN-SVG/index.html

Keras.js I

- Herramienta web
- Despliega una red neural en el navegador
  - Permite usar GPU
- Muchos ejemplos
  - Muestran las capas en tiempo real

Keras.js II - Demo

Demo

https://transcranial.github.io/keras-js/#/mnist-cnn

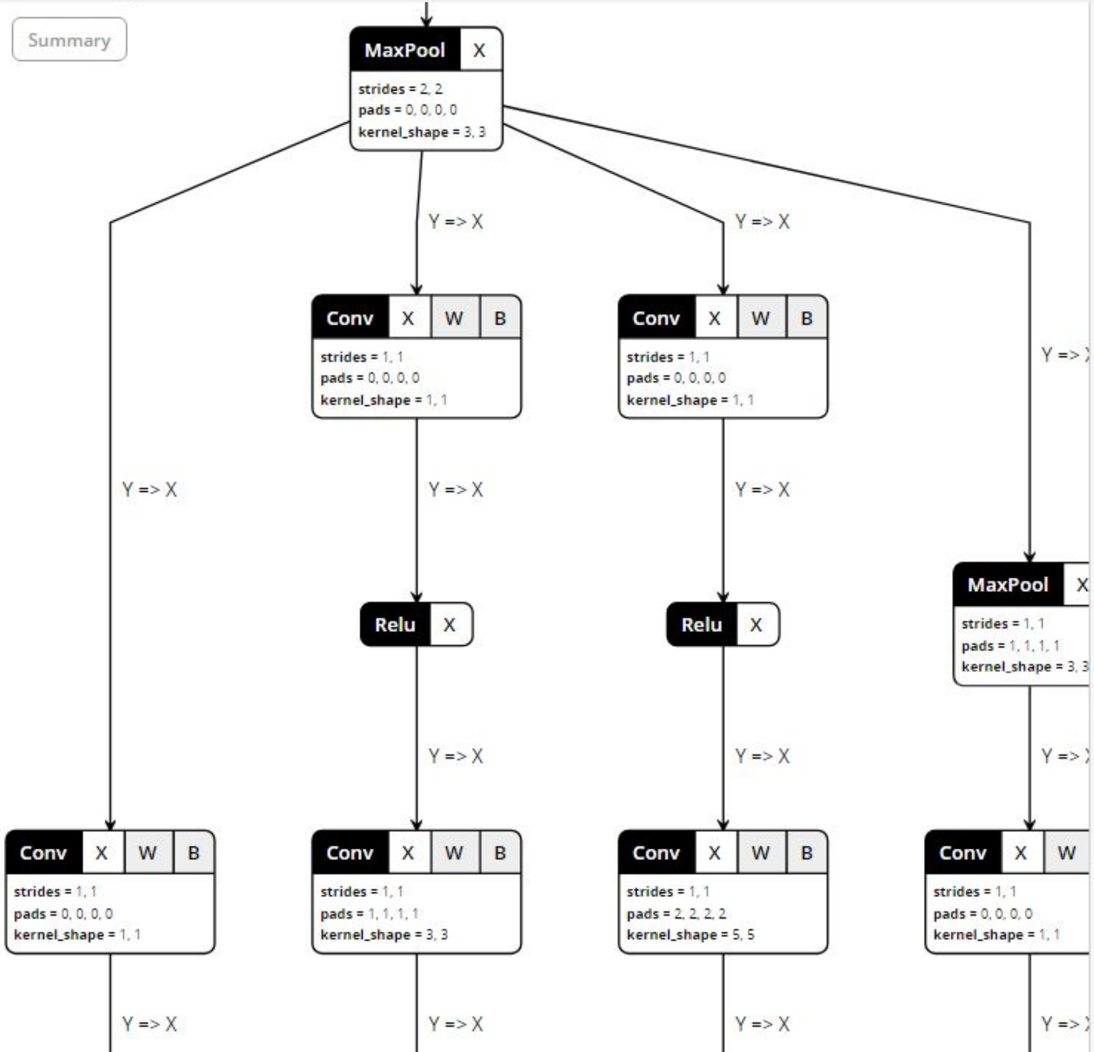
Netron I

- Herramienta para visualizar redes neuronales
- Desarrollado por Lutz Roeder
  - Visual Studio
- Gratuita

Netron II

- Soporta la mayoría de frameworks
  - Importar los modelos
- Permite consultar el API y la documentación de las capas que utiliza el modelo

File Edit Help



DOCUMENTATION

MaxPool

MaxPool consumes an input tensor X and applies max pooling across the the tensor according to kernel sizes, stride sizes, and pad lengths. max pooling consisting of computing the max on all values of a subset of the input tensor according to the kernel size and downsampling the data into the output tensor Y for further processing. MaxPool consumes an input tensor X and applies max pooling across the the tensor according to kernel sizes, stride sizes, and pad lengths. max pooling consisting of computing the max on all values of a subset of the input tensor according to the kernel size and downsampling the data into the output tensor Y for further processing. MaxPool consumes an input tensor X and applies max pooling across the the tensor according to kernel sizes, stride sizes, and pad lengths. max pooling consisting of computing the max on all values of a subset of the input tensor according to the kernel size and downsampling the data into the output tensor Y for further processing. MaxPool consumes an input tensor X and applies max pooling across the the tensor according to kernel sizes, stride sizes, and pad lengths. max pooling consisting of computing the max on all values of a subset of the input tensor according to the kernel size and downsampling the data into the output tensor Y for further processing.

Attributes

`auto_pad` : string

auto_pad must be either SAME_UPPER, SAME_LOWER or VALID. Where SAME_UPPER or SAME_LOWER mean pad the input so that the ouput size match the input.In case of odd number add the extra padding at the end for SAME_UPPER and at the begining for SAME_LOWER. VALID mean no padding. DEPRECATION NOTE: auto_pad is only intended to support legacy uses, and for framework authors, one is explicitly encouraged to use explicit padding specified in the pads attribute.

`kernel_shape` : list of ints

The size of the kernel along each axis.

`pads` : list of ints

Padding for the begining and ending along each axis, it can take any

Conclusiones

Conclusiones

- La Inteligencia Artificial es muy compleja
  - Se necesitan herramientas de visualización
- Multitud de alternativas
- Mejor comprensión de nuestros modelos

¿Preguntas?