Proyecto Redes Neuronales 2020-2

Clasificador de Dibujos

Monreal Gamboa Francisco Manuel Ramírez García Diana Isabel

Universidad Nacional Autónoma de México
Facultad de Ciencias

12 de junio de 2020

Índice

Introducción

Introducción

¿Cómo nos comunicamos?

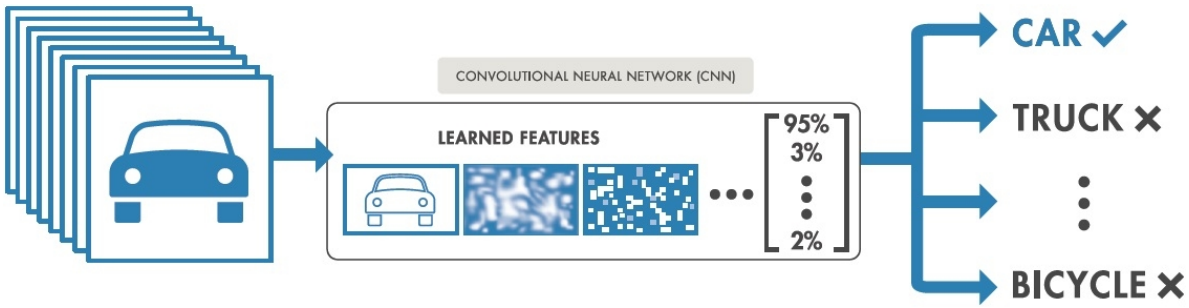
Cuando necesitamos comunicarnos y el idioma se transforma en muros, necesitamos buscar otras opciones tal como las señas y los dibujos.

Objetivo

Objetivo

¿Cómo lo resolvemos?

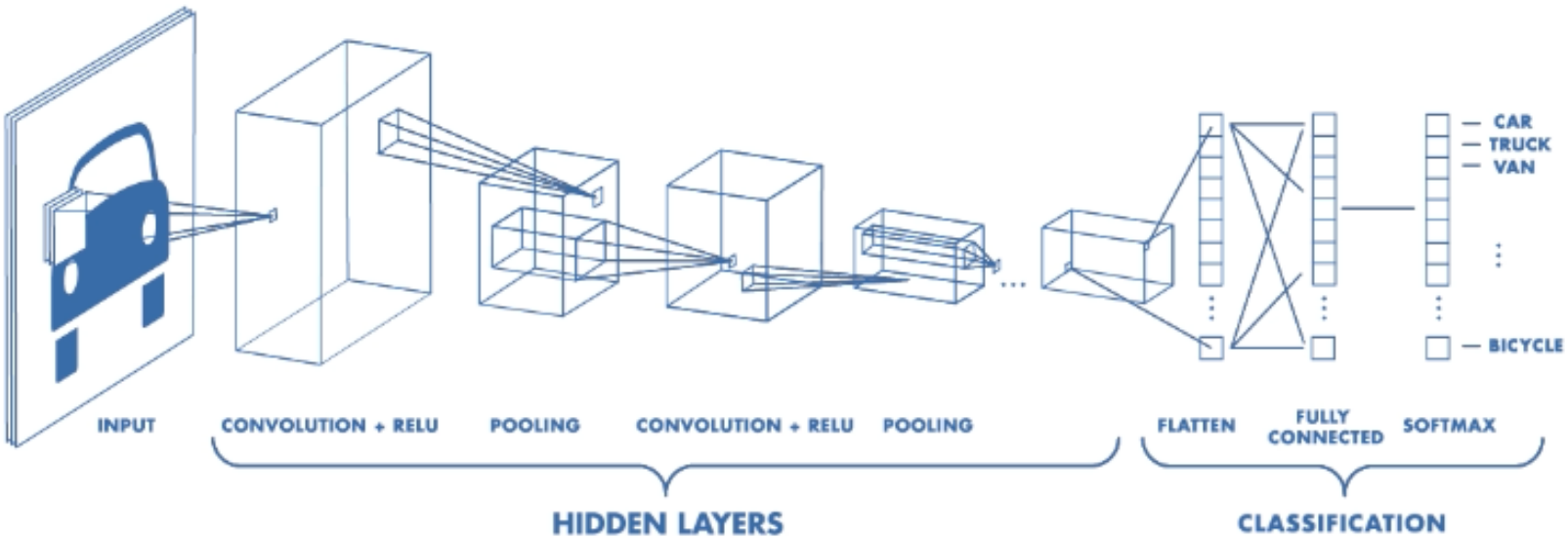
Buscamos generar una red neuronal convolusional tal que tenga la capacidad de identificar distintos objetos por medio de dibujos escenciales o en otras palabras garabatos.



Modelo

Base de Datos

Modelo


CAR
TRUCK
VAN
BICYCLE
INPUT
CONVOLUTION + RELU
POOLING
CONVOLUTION + RELU
POOLING
FLATTEN
FULLY CONNECTED
SOFTMAX
HIDDEN LAYERS
CLASSIFICATION

Modelo

Modelo

In [30]:

```
import numpy as np
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout, Flatten, Activation
from tensorflow.keras.layers import Convolution2D, MaxPooling2D
from tensorflow.keras.utils  import to_categorical
import matplotlib
%matplotlib inline
from matplotlib import pyplot as plt

#creamos nuestro modelo
modelo = tf.keras.models.Sequential([
  tf.keras.layers.Convolution2D(16, (3, 3),padding='same',input_shape=x_tr.shape[1:], activatio
n='relu'),
  tf.keras.layers.MaxPooling2D(pool_size=(2,2)),
  tf.keras.layers.Convolution2D(32, (3, 3), activation='relu'),
  tf.keras.layers.MaxPooling2D(pool_size=(2,2)),
  tf.keras.layers.Convolution2D(64, (3, 3), activation='relu'),
  tf.keras.layers.MaxPooling2D(pool_size=(2,2)),
  #aplanamos los datos del modelo hasta este punto
  tf.keras.layers.Flatten(),
  tf.keras.layers.Dense(128, activation='relu'),
  tf.keras.layers.Dropout(0.2),
  tf.keras.layers.Dense(120, activation='softmax')
])


modelo.summary()
```

Modelo

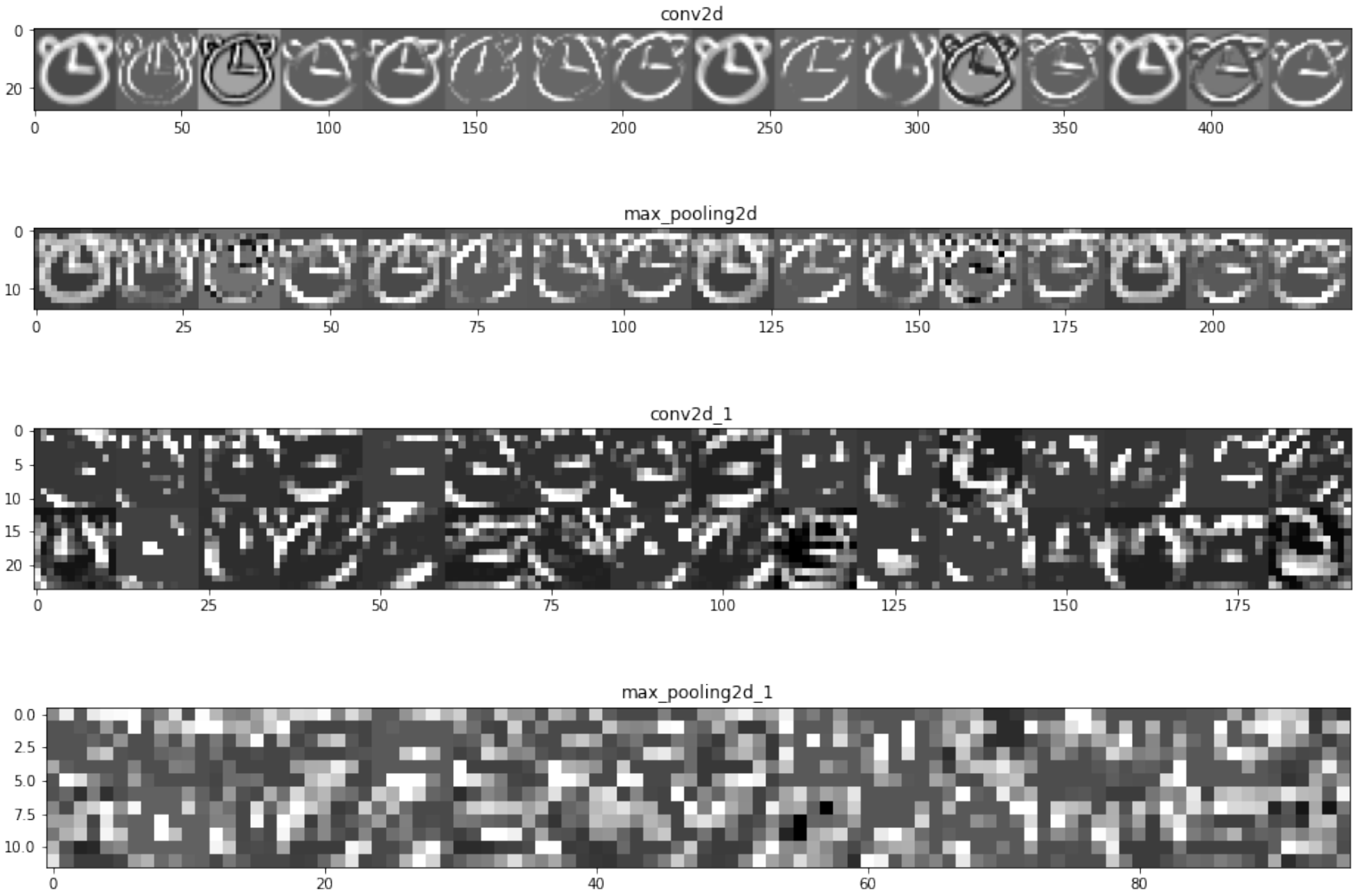
In [31]:
```
#Compilamos nuestro modelo
modelo.compile(optimizer='adam',
               loss='categorical_crossentropy',
               metrics=['top_k_categorical_accuracy'])

#Entrenamos el modelo
H = modelo.fit(x = x_tr, y = y_tr, validation_split=0.1, batch_size = 256, verbose=1, epochs=7)
```

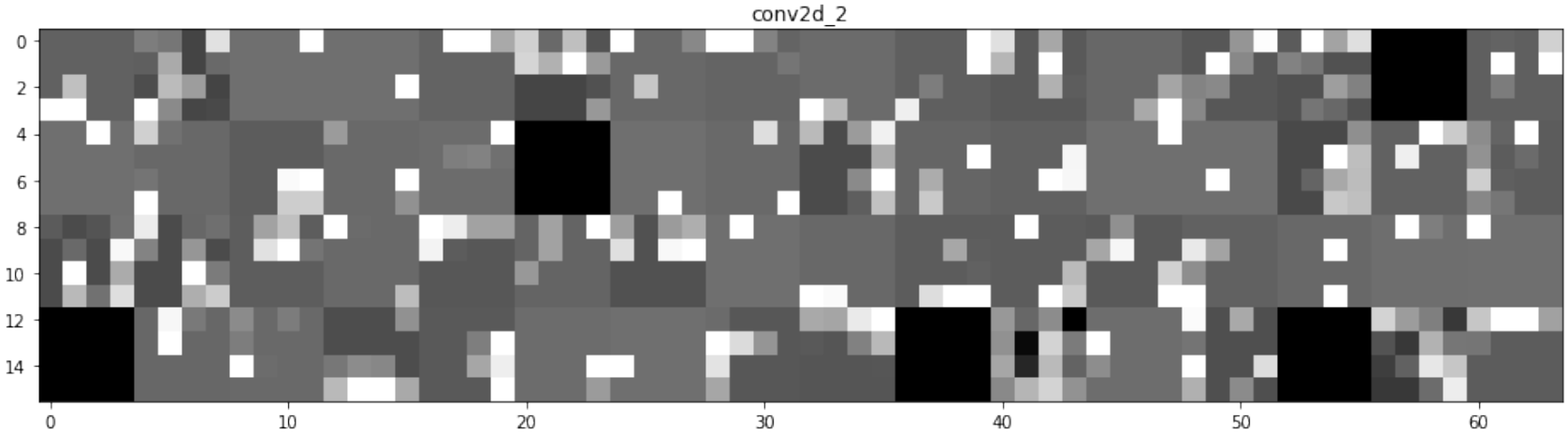
```
Epoch 1/7
1350/1350 [==============================] - 185s 137ms/step - loss: 2.3559 - top_k_categorical_
accuracy: 0.7030 - val_loss: 1.5826 - val_top_k_categorical_accuracy: 0.8484
Epoch 2/7
1350/1350 [==============================] - 185s 137ms/step - loss: 1.6315 - top_k_categorical_
accuracy: 0.8420 - val_loss: 1.3720 - val_top_k_categorical_accuracy: 0.8765
Epoch 3/7
1350/1350 [==============================] - 184s 136ms/step - loss: 1.4727 - top_k_categorical_
accuracy: 0.8640 - val_loss: 1.2899 - val_top_k_categorical_accuracy: 0.8864
Epoch 4/7
1350/1350 [==============================] - 185s 137ms/step - loss: 1.3853 - top_k_categorical_
accuracy: 0.8751 - val_loss: 1.2186 - val_top_k_categorical_accuracy: 0.8953
Epoch 5/7
1350/1350 [==============================] - 184s 136ms/step - loss: 1.3232 - top_k_categorical_
accuracy: 0.8822 - val_loss: 1.1868 - val_top_k_categorical_accuracy: 0.8976
Epoch 6/7
1350/1350 [==============================] - 184s 136ms/step - loss: 1.2781 - top_k_categorical_
accuracy: 0.8874 - val_loss: 1.1341 - val_top_k_categorical_accuracy: 0.9039
Epoch 7/7
1350/1350 [==============================] - 185s 137ms/step - loss: 1.2405 - top_k_categorical_
accuracy: 0.8916 - val_loss: 1.1155 - val_top_k_categorical_accuracy: 0.9054
```

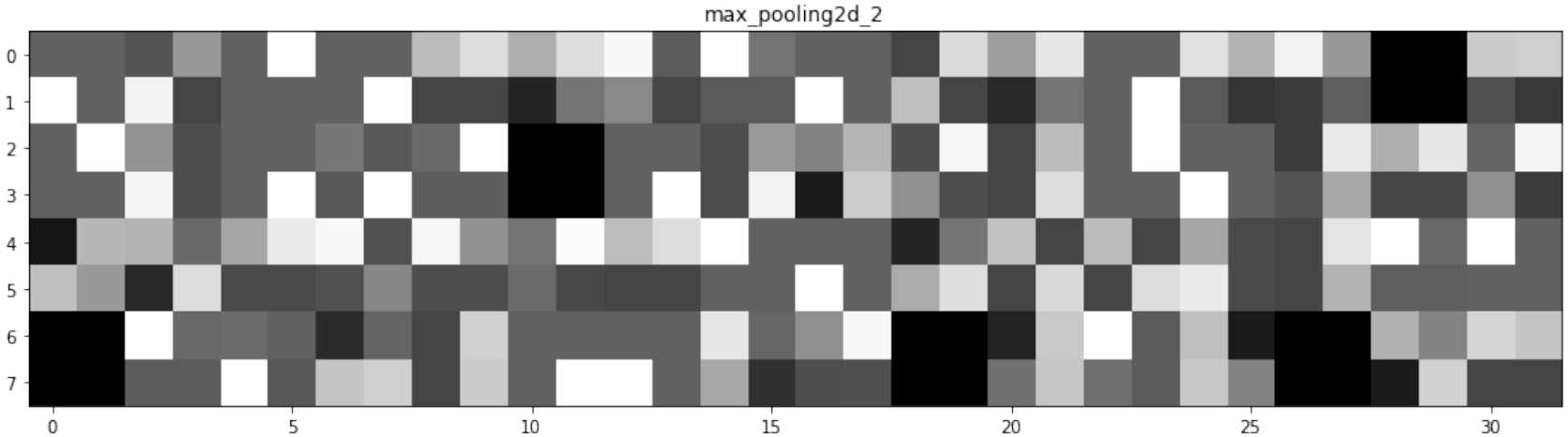
Resultados

Desarrollo de las Capas

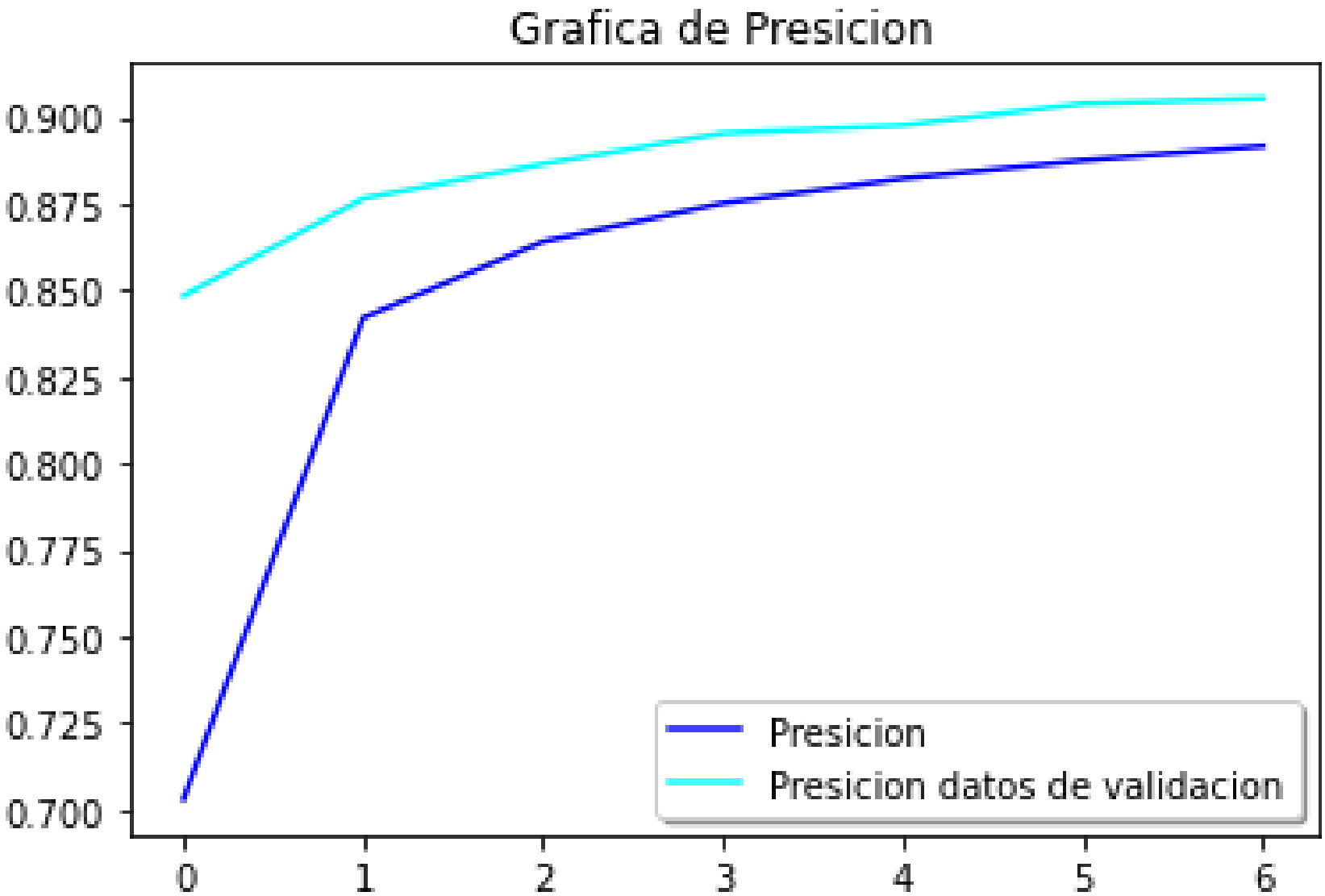


Desarrollo de las Capas

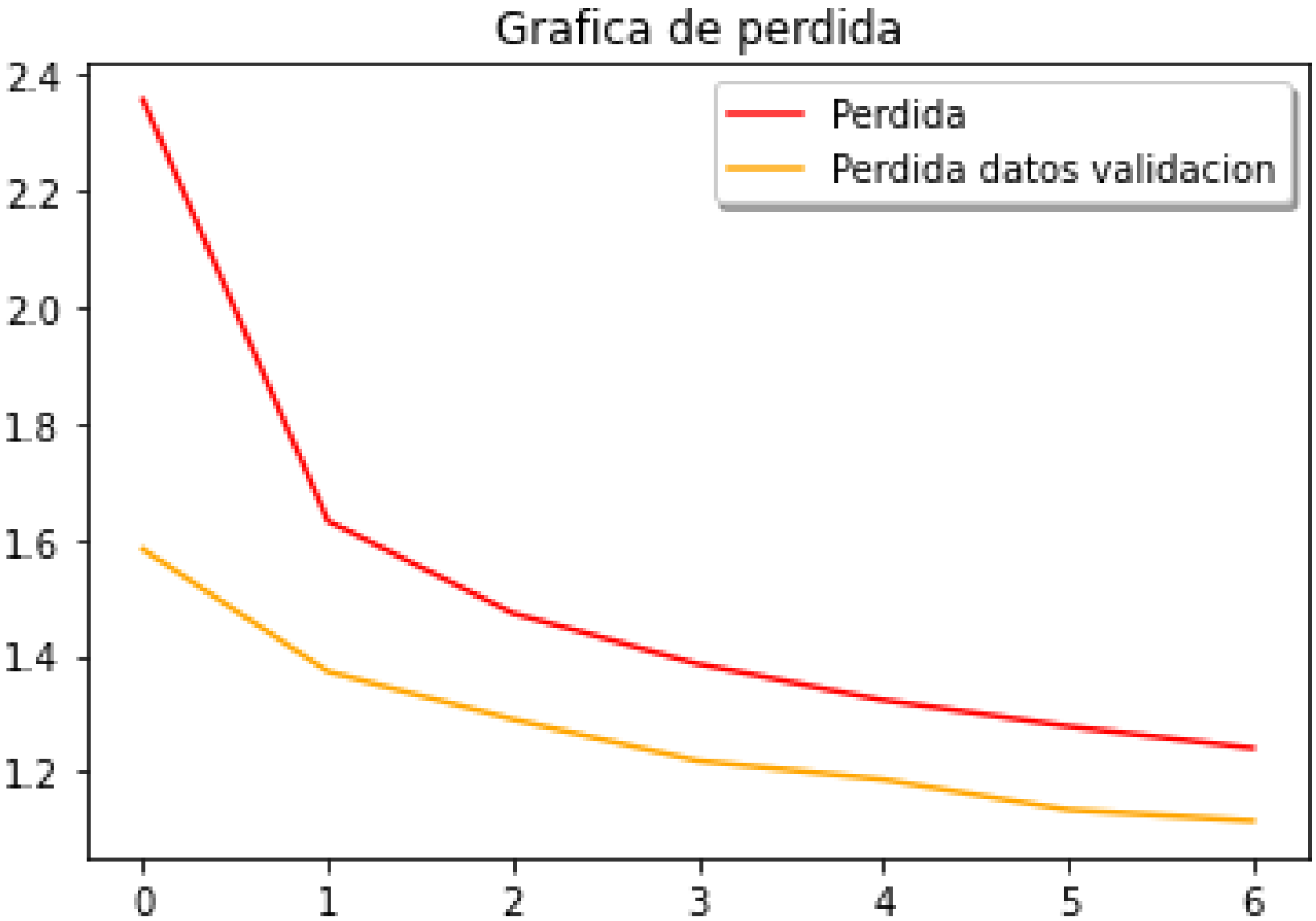




Precisión


Grafica de Presicion
0.900
0.875
0.850
0.825
0.800
0.775
0.750
0.725
0.700
0
1
2
3
4
5
6
Presicion
Presicion datos de validacion

Pérdida


Grafica de perdida
Perdida
Perdida datos validacion
2.4
2.2
2.0
1.8
1.6
1.4
1.2
0
1
2
3
4
5
6

Comparación

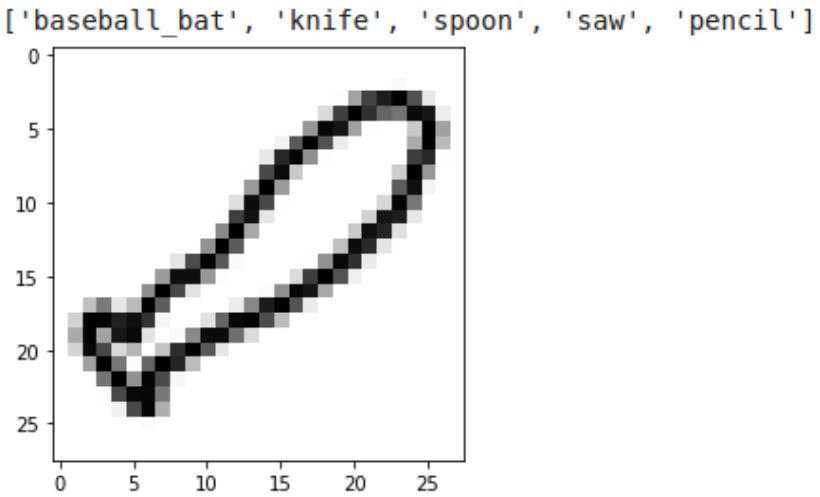


Figure: Imagen digital

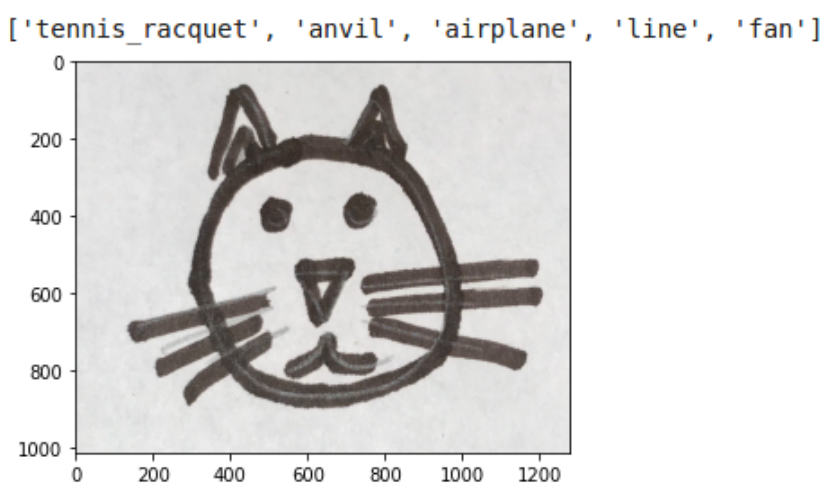


Figure: Fotografía

Interfaz Gráfica

Clasificador de 100 imagenes

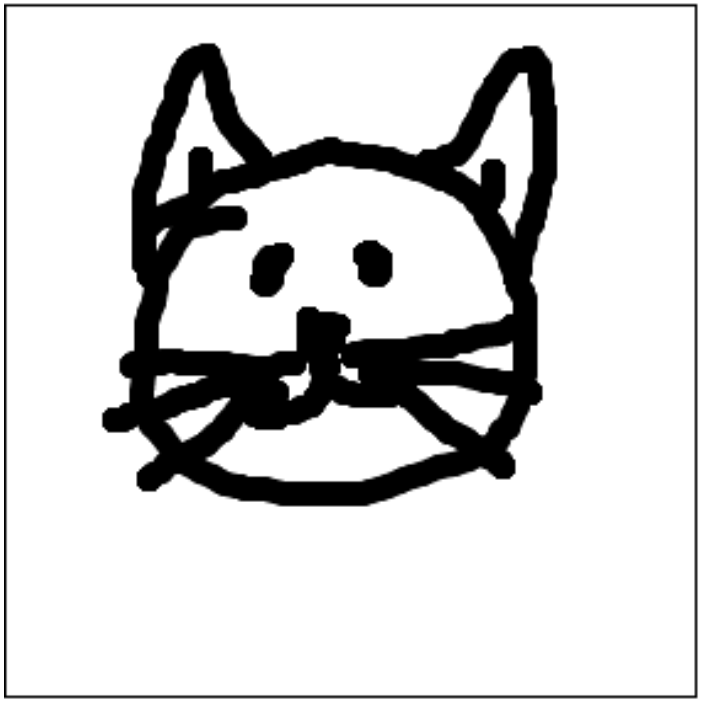
Borrar

butterfly (93.16%),
grapes (2.1399999999999997%),
flower (1.49%),
ceiling_fan (1.37%),
fan (0.79%),
spider (0.21%),
candle (0.16999999999999998%),
lightning (0.13%),
sun (0.1%),
coffee_cup (0.06%)

Interfaz Gráfica

Clasificador de 100 imagenes

Borrar



cat (95.46%),
lightning (1.1900000000000002%),
cloud (0.69%),
face (0.5%),
flower (0.32%),
cookie (0.25%),
diving_board (0.16%),
anvil (0.15%),
coffee_cup (0.13999999999999999%),
grapes (0.11%)

Conclusión

Base de Datos