Implementación de una CNN utilizando el set de datos CIFAR10

Ronny Jiménez
Fabricio Quirós

Agenda

- Objetivos
- Conjunto de Datos
- Propuestas evaluadas
- Solución definitiva
- Resultados
- Conclusiones
- Referencias

Objetivos del proyecto

General

- Implementación de un algoritmo de aprendizaje profundo, utilizando redes neuronales convolutivas para la detección de 10 tipos diferentes de objetos en el dataset CIFAR10

Específicos

- Obtención del conjunto de datos CIFAR10 y visualización de ellos
- Determinar preprocesado basado en estudio del set de datos
- Implementación de un modelo de red neuronal convolutiva
- Entrenamiento utilizando el modelo seleccionado
- Optimización / *tunning*
- Validación

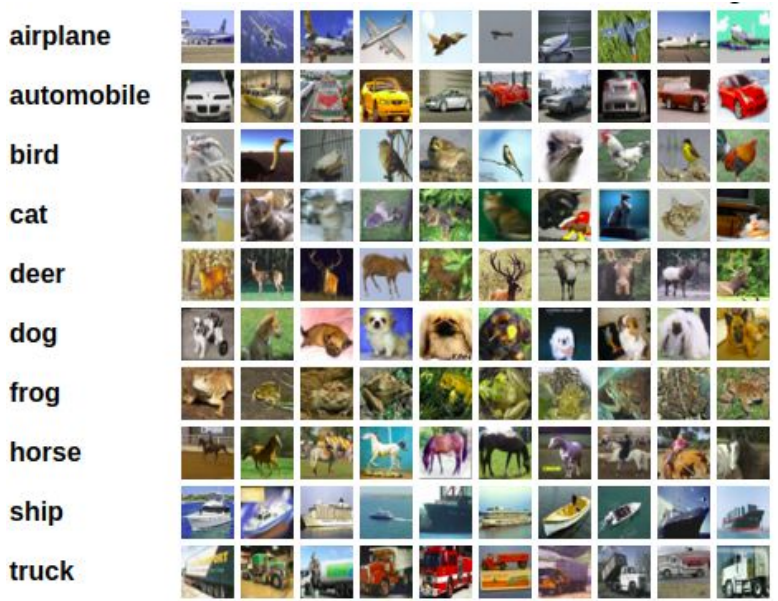
Conjunto de datos CIFAR10

10 Categorias etiquetadas

Del 0 al 9

Diferentes objetos

Baja resolución: 32x32



Pre-procesamiento efectuado

- Reajuste de la forma de los datos de entrenamiento y pruebas.
- Normalización de los datos
- Categorización de las clases tanto de entrenamiento como las de prueba.
- PCA: underfitting.

Solución propuesta

- 4 etapas de convolución
- 2 etapas de pooling
- 3 dropouts
- 1 aplanamiento
- 2 etapas de capas totalmente conectadas

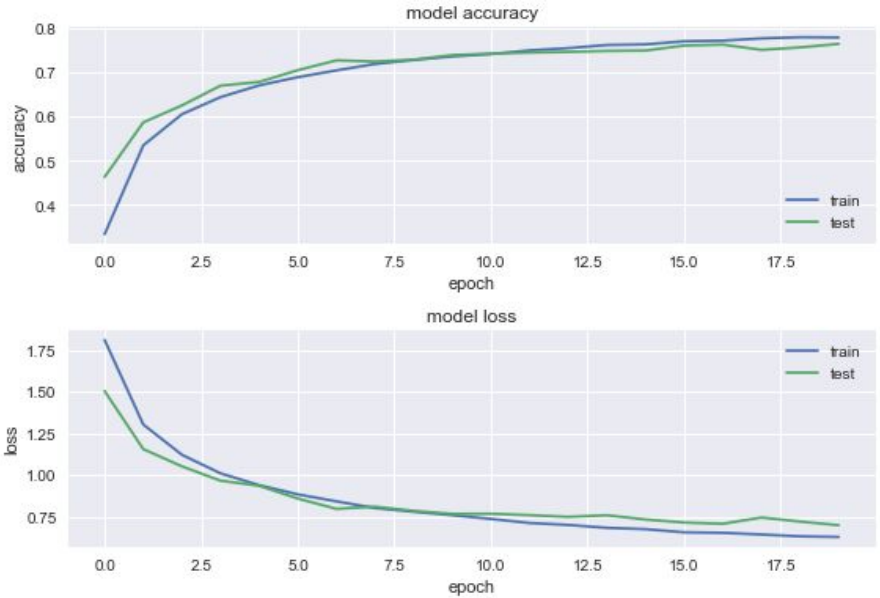
Modelo propuesto

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 32, 32, 32)	896
conv2d_2 (Conv2D)	(None, 30, 30, 32)	9248
max_pooling2d_1 (MaxPooling2	(None, 15, 15, 32)	0
dropout_1 (Dropout)	(None, 15, 15, 32)	0
conv2d_3 (Conv2D)	(None, 15, 15, 64)	18496
conv2d_4 (Conv2D)	(None, 13, 13, 64)	36928
max_pooling2d_2 (MaxPooling2	(None, 6, 6, 64)	0
dropout_2 (Dropout)	(None, 6, 6, 64)	0
flatten_1 (Flatten)	(None, 2304)	0
dense_1 (Dense)	(None, 512)	1180160
dropout_3 (Dropout)	(None, 512)	0
dense_2 (Dense)	(None, 10)	5130

Total params: 1,250,858
Trainable params: 1,250,858
Non-trainable params: 0

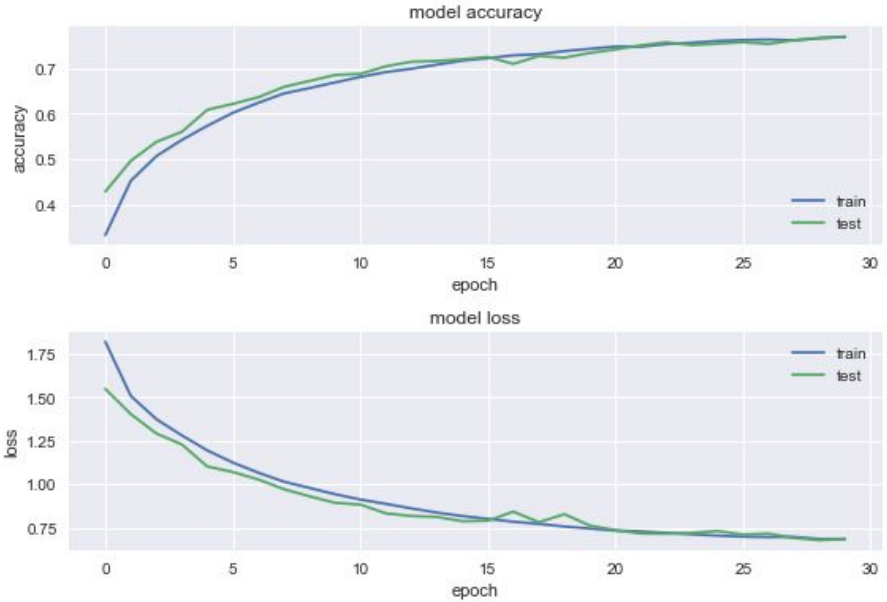
Resultados - Pruebas parte 1

- Hiper-parámetros:
  - Tamaño del batch: 64
  - Epochs: 20
  - Optimizador: Nadam
- Resultados del entrenamiento:
  - Duración: 421.31 min / 25278.89 sec
  - Precisión: 77.87%
  - Pérdidas: 62.76%
- Resultados de la evaluación (testing):
  - Duración: 1.21 min / 69.57 sec
  - Precisión: 76.41 %
  - Pérdidas: 69.81%



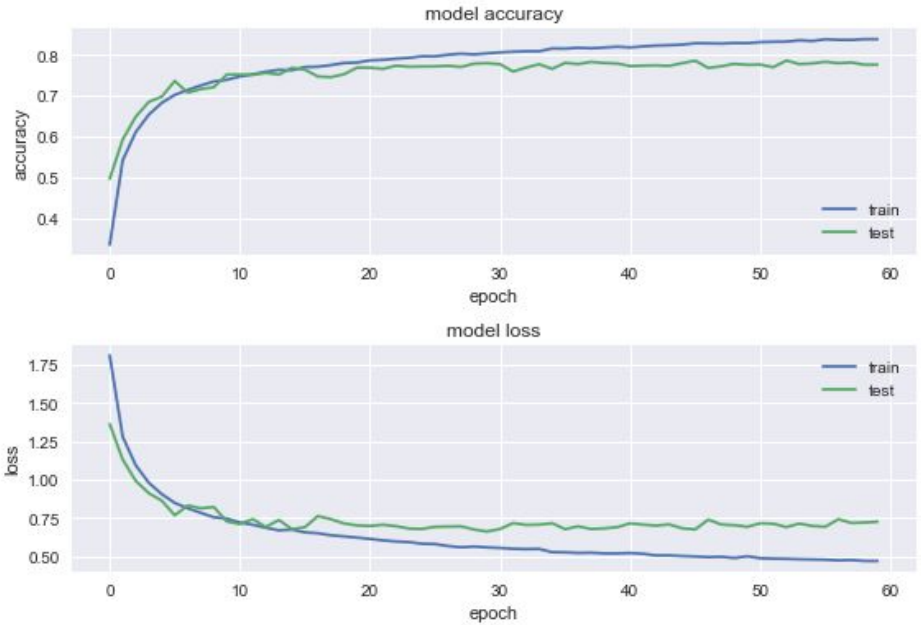
Resultados - Pruebas parte 2

- Hiper-parámetros:
  - Tamaño del batch: 32
  - Epochs: 30
  - Optimizador: RMSprop
- Resultados del entrenamiento:
  - Duración: 651.38 min / 39082.93 sec
  - Precisión: 77%
  - Pérdidas: 68.49%
- Resultados de la evaluación (testing):
  - Duración: 1.16 min / 69.57 sec
  - Precisión: 76.9 %
  - Pérdidas: 68.55%



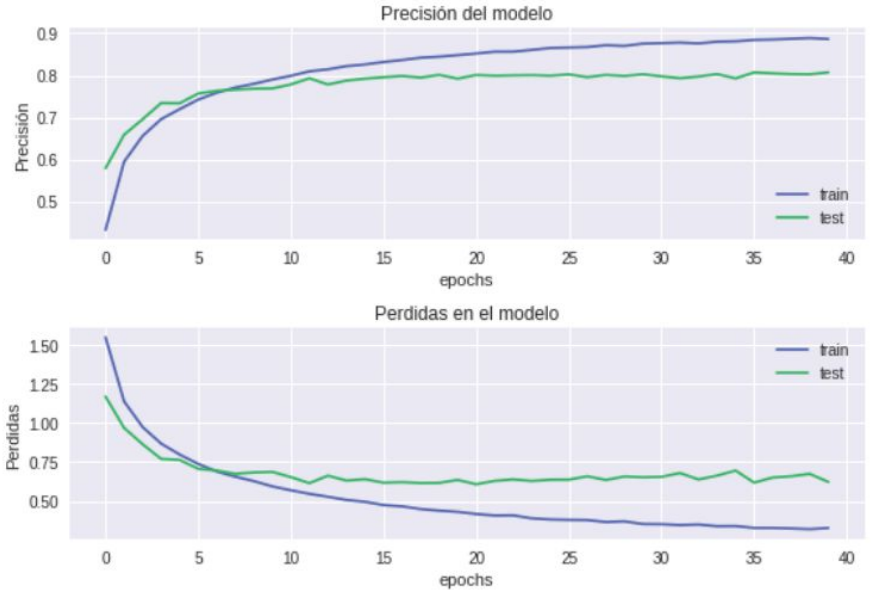
Resultados - Pruebas parte 3

- Hiper-parámetros:
  - Tamaño del batch: 64
  - Epochs: 60
  - Optimizador: Nadam
- Resultados del entrenamiento:
  - Duración: 1214.4 min / 72864.11 sec
  - Precisión: 83.88%
  - Pérdidas: 47.13%
- Resultados de la evaluación (testing):
  - Duración: 1.25 min / 74.95 sec
  - Precisión: 77.61 %
  - Pérdidas: 72.69%



Resultados - Pruebas parte 4

- Hiper-parámetros:
  - Tamaño del batch: 64
  - Epochs: 40
  - Optimizador: Adam
  - Learning rate: 0.001
- Resultados del entrenamiento:
  - Duración: 30.73 min / 1844.02 sec
  - Precisión: 88.62%
  - Pérdidas: 32.69%
- Resultados de la evaluación (testing):
  - Duración: 0.03 min / 2.01 sec
  - Precisión: 80.67 %
  - Pérdidas: 62.28%



Resultados finales

Se realizó un cambio en la arquitectura, incrementando la cantidad de filtros de cada capa convolucional, además se movieron las funciones de activación entre cada etapa de convolución

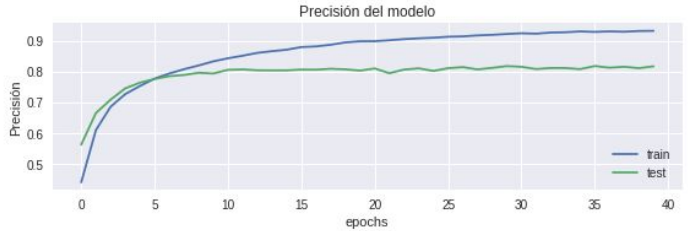
Layer (type)	Output Shape	Param #
conv2d_5 (Conv2D)	(None, 32, 32, 32)	896
activation_7 (Activation)	(None, 32, 32, 32)	0
conv2d_6 (Conv2D)	(None, 30, 30, 64)	18496
activation_8 (Activation)	(None, 30, 30, 64)	0
max_pooling2d_3 (MaxPooling2	(None, 15, 15, 64)	0
dropout_4 (Dropout)	(None, 15, 15, 64)	0
conv2d_7 (Conv2D)	(None, 15, 15, 128)	73856
activation_9 (Activation)	(None, 15, 15, 128)	0
conv2d_8 (Conv2D)	(None, 13, 13, 256)	295168
activation_10 (Activation)	(None, 13, 13, 256)	0
max_pooling2d_4 (MaxPooling2	(None, 6, 6, 256)	0
dropout_5 (Dropout)	(None, 6, 6, 256)	0
flatten_2 (Flatten)	(None, 9216)	0
dense_3 (Dense)	(None, 512)	4719104
activation_11 (Activation)	(None, 512)	0
dropout_6 (Dropout)	(None, 512)	0
dense_4 (Dense)	(None, 10)	5130
activation_12 (Activation)	(None, 10)	0

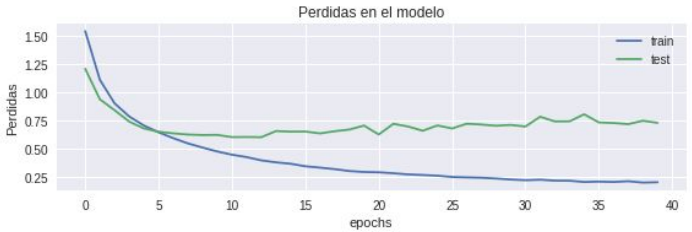
Total params: 5,112,650
Trainable params: 5,112,650
Non-trainable params: 0

Resultados finales

- Hiper-parámetros:
  - Tamaño del batch: 64
  - Epochs: 40
  - Optimizador: Adam
  - Learning rate: 0.001
- Resultados del entrenamiento:
  - Duración: 95.57 min / 5734.22 sec
  - Precisión: 93.16%
  - Pérdidas: 20.45%
- Resultados de la evaluación (testing):
  - Duración: 0.12 min / 7.06 sec
  - Precisión: 81.63 %
  - Pérdidas: 72.89%

```
Duración del entrenamiento: 95.57 minutos : 5734.22 segundos
Test loss: 0.7289218351840973
Test accuracy: 81.63 %
Perdidas en la parte del testing: 0.7289218351840973
Precisión del testing: 81.63 %
Tiempo de testing: 0.12 minutos : 7.06 segundos
```





Conclusiones

- Un pre-procesamiento basado en el PCA del dataset en cuestión mostró underfitting, debido a la baja resolución de las imágenes.
- La reducción del tamaño del batch, así como el incremento en la cantidad de epochs fue favorable al momento de someter la CNN al set de pruebas.
- Se mostró un alto overfitting, posiblemente debido a no tener suficiente cantidad de datos y la gran cantidad de parámetros que se producen después de cada convolución.
- Incrementar el tamaño de los filtros de la convolución produjo más parámetros, pero mejores resultados en el entrenamiento, siempre manteniendo resultados desfavorables en la parte de validación

Referencias

- https://www.cs.toronto.edu/~kriz/cifar.html
- https://keras.io/
- https://github.com/fmezacr/patrones/blob/master/CNN/CNN_MNIST_KERAS.ipynb
- https://towardsdatascience.com/cifar-10-image-classification-in-tensorflow-5b501f7dc77c