INFORME DE METRICAS MODELOS DE CNN

RECETA

Antes de entrenar:

1) Datos: Hemos procedido a la descarga de datos, visualización de las clases para verificar outliers, en el preprocesado hemos normalizado los pixels. Hemos ajustado los valores para que estén en un rango entre 0 y 1 con el fin de mejorar la estabilidad del entrenamiento del modelo de aprendizaje automático o ayudar a la convergencia del algoritmo.

2) Objetivo (función de coste): "Categorical Crossentropy" La entropía cruzada categórica, tiene como objetivo minimizar esta función de costo durante el entrenamiento, así el modelo se ajusta para producir predicciones más calibradas, es decir, predicciones que reflejen con mayor precisión la probabilidad real de pertenencia a cada clase

3) Define Baselines: Valor de la pérdida al principio del entrenamiento ()

En modelos de la pagina huggingface, encontramos modelos de referencia con los siguientes datos:

Loss: 0.0138

Accuracy: 1.0

The following hyperparameters were used during training:

learning_rate: 0.0002

train_batch_size: 16

eval_batch_size: 8

seed: 42

optimizer: Adam with betas=(0.9,0.999) and epsilon=1e-08

lr_scheduler_type: linear

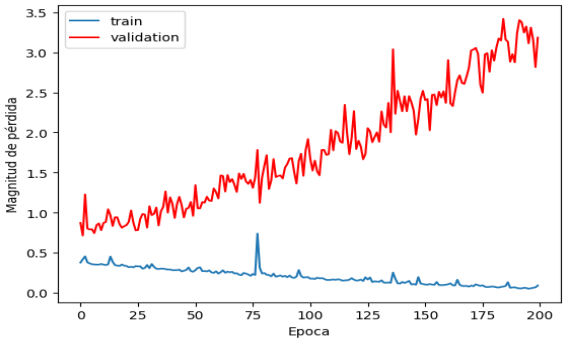
num_epochs: 4

Para conseguir un modelo simple partiendo de 4 filtros y con imágenes de 150 x 150 pixeles, ha sido necesario aplicar varias capas convolucionales y Max_Pooling2 para reducir al máximo el número de parámetros (1325). Entrenamiento con CPU, muy lento. Entrenamiento con CPU con error

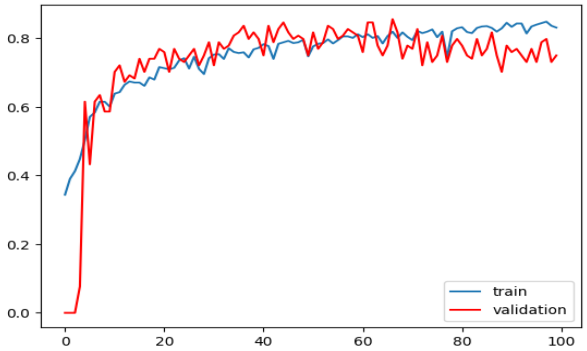
1. *Un modelo con el menor número de parámetros*

Modelo Simple	Total n.param.	Loss_train	Acc_Train	Loss_val	Acc_val	Loss_test	Acc_test
4C2D+inp_s(150,150,3)RR_p	1325	0.5764	0.7775	1.0421	0.6280	0.7120	0.7422
4C2D+inp_s(150,150,3)ear_st	1325	0.6248	0.7376	1.0579	0.6039	2.3054	0.6641

Evolución Loss



Evolucion Accuracy

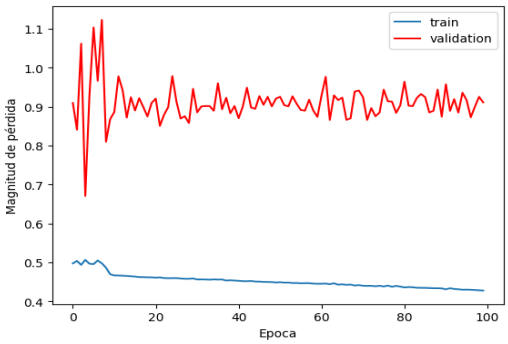


En los grafico de la evolucion de la funciòn de perdida podemos observar que hay que meter Regularizaciòn, ya que hay que conseguir que la curva de validaciòn baje y se junte con la curva de train.

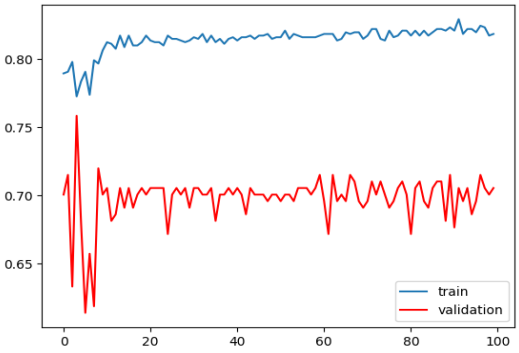
Cambios en el modelo inicial: se introducen cambios en el modelo inicial y de consiguen las siguientes mètricas

Modelo1 con Callbacks-east-ReRpl	Total n.param.	Loss_train	Acc_Train	Loss_val	Acc_val	Loss_test	Acc_test
4C2D+inp_s(150,150,3)eastopp	2697	0.5141	0.7823	0.9187	0.6957	0.6857	0.7109
4C2D+inp_s(150,150,3)ReRpl		0.4280	0.8186	0.9111	0.7053	0.6857	0.7109

Evolución Loss



Evolucion Accuracy

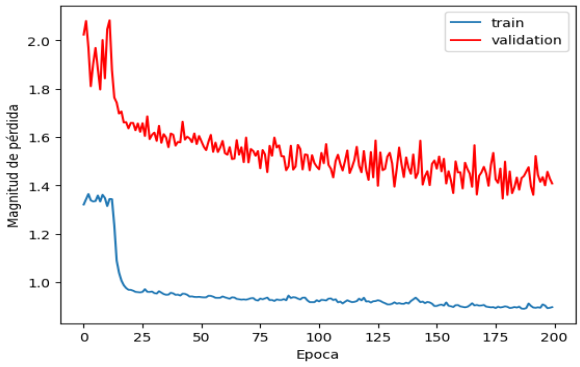


Cambios en el segundo modelo: aumento de neuronas en una capa convolucional y regularizacion L1 de 02

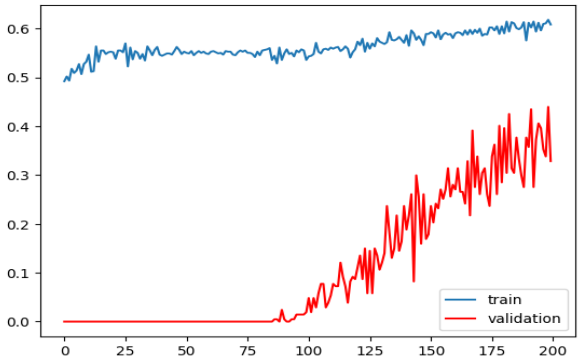
Modelo2 con Callbacks-east-ReRpl	Total n.param.	Loss_train	Acc_Train	Loss_val	Acc_val	Loss_test	Acc_test
4C2D+inp_s(150,150,3)eastopp	15143	1.3351	0.4764	1.9627	0.0000	0.9349	0.6094
4C2D+inp_s(150,150,3)ReRpl		0.8971	0.6082	1.4078	0.3285	0.9349	0.6094

Es importante indicar que a este modelo le falto entrenamiento, ya que se quedaba amitad de las epocas y se producía un error de uso GPU, se tuvo que intentar varias veces el entrenamiento y era muy lento.

Evolución Loss



Evolución Accuracy

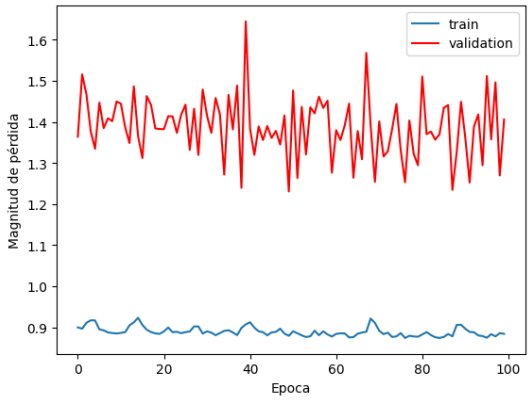


2. *Un modelo con más parámetros que número de datos.*

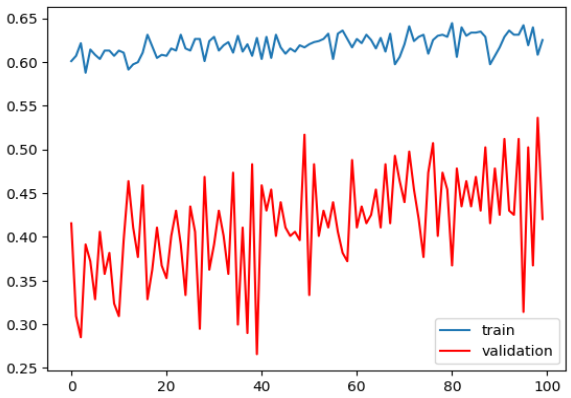
Para conseguir un modelo con más parámetros hemos cogido las imágenes de 150 x 150 pixeles, hemos aplicado 4 capas Conv2D y 4Max_Pooling2 y 2 capas densas con más neuronas consiguiendo un modelo más grande con un número de parámetros (173307). Entrenamiento con CPU, muy lento.

Modelo3 con Callbacks-east-ReRpl	Total n.param.	Loss_train	Acc_Train	Loss_val	Acc_val	Loss_test	Acc_test
4C2D+inp_s(150,150,3)eastopp	173307	0.9495	0.6034	1.5232	0.2464	1.0610	0.5234
4C2D+inp_s(150,150,3)ReRpl		0.8840	0.6252	1.4058	0.4203		

Evolución Loss



Evolución Accuracy



No obstante, se hayan aplicado diferentes técnicas de regularización, como dropout y regularización a distintos niveles y agregado más capas convolucionales con más neuronas, no se ha logrado que las curvas converjan durante el entrenamiento. De lo anterior se pueden extraer varias conclusiones: Es posible que el modelo esté experimentando un sobreajuste significativo a los datos de entrenamiento, a pesar de la aplicación de estas técnicas de regularización no generaliza bien a nuevos datos. Esto puede deberse a que hemos agregado una complejidad innecesaria en el modelo, o bien a falta de datos de entrenamiento suficientes o insuficiente regularización.