Redes Neuronales Artificiales: MNIST y CIFAR

Ramon Ruiz Dolz

December 2018

1 Introducción

Este trabajo consiste en el diseño y entrenamiento de redes neuronales. Concretamente se ha trabajado con dos tipos de redes neuronales, el *multi-layer perceptron* (MLP) y las redes convolucionales (CNN). Se han entrenado múltiples MLPs probando y aplicando distintas técnicas sobre el dataset MNIST. MNIST es un conjunto de datos formado por números escritos a mano, este dataset se utiliza comunmente para comprobar el correcto funcionamiento de una red neuronal a muy alto nivel, debido a la simpleza de este. En la Figure 1 podemos observar algunos ejemplos extraidos de MNIST.

Figure 1: Ejemplo de MNIST

Por otra parte, se han entrenado múltiples CNNs sobre el dataset CIFAR10.

En este caso, el dataset CIFAR10 consiste en 60000 imágenes de 32x32 separadas en 10 clases distintas. De esta forma, la red convolucional debe ser capaz de discriminar entre estas 10 clases distintas al recibir una imagen como entrada.

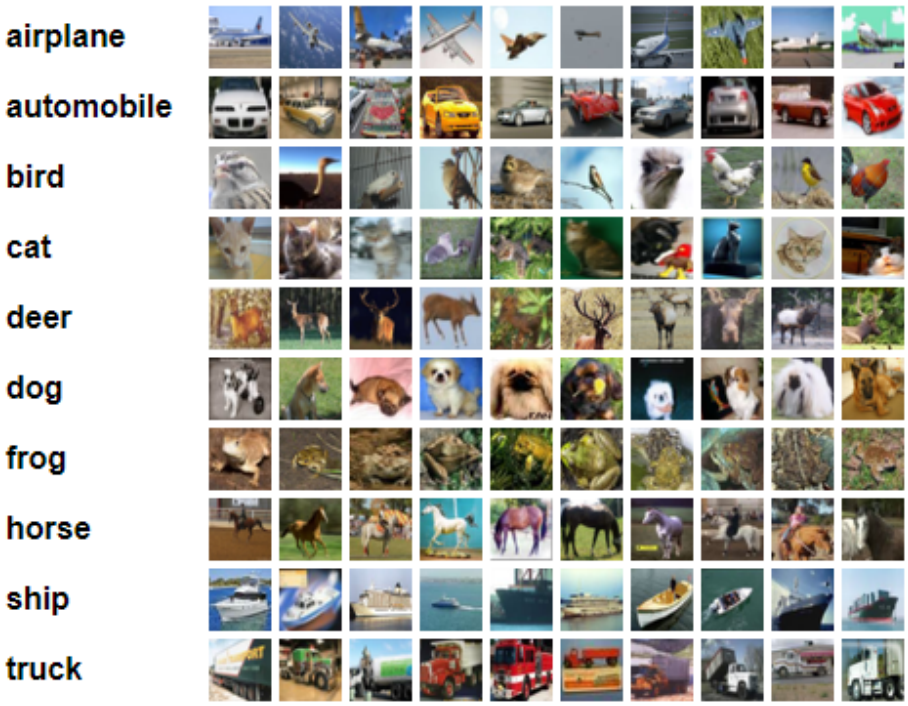

Figure 2: Ejemplo de CIFAR10

En la Figure 2 se puede observar un pequeño ejemplo de las distintas clases existentes en CIFAR10 y algunas fotos de ejemplo de cada clase.

En las siguientes secciones se han detallado las distintas arquitecturas de MLP y CNN entrenadas y la precisión obtenida en cada una de ellas.

2 Multi-Layer Perceptron

Para la realización de esta práctica se han entrenado tres MLP distintos tratando de analizar la repercusión del ruido gausiano y del *batch normalization*. Tras lanzar una serie de experimentos previos, he decidido trabajar con una red de la siguiente estructura debido a los mejores resultados obtenidos.

La red cuenta con tres capas ocultas más la de entrada y salida. Estas capas, formadas por 512 neuronas están conectadas a una capa de *batch normalization* y esta a otra capa de ruido gaussiano. Finalmente se comunican con la capa de activación, en este caso una función ReLU. Por otra parte, la capa de salida consiste en una función softmax que nos proporciona las probabilidades de

pertenecer a cada una de las posibles clases.

Los experimentos se han configurado con batches de tamaño 100 y 200 epochs por red a entrenar. Además se ha hecho uso de la técnica de data augmentation, es decir, generar más muestras de train a partir del dataset básico.

Por otra parte, para tratar de evitar que la red sobreentrenase los datos de entrenamiento también se ha hecho uso de un callback provisto por la librería Keras. El callback *ReduceLRonPlateau* consiste en reducir el *learning rate* al detectar que el modelo ha dejado de mejorar. Concretamente se ha configurado para que pasados 10 epochs sin mejora alguna, el *learning rate* sea reducido en un 50% y no pueda bajar de 0.005.

A continuación se muestran los resultados obtenidos con las tres redes neuronales empleadas para esta práctica.

Neural	**Extra Layers**	**Accuracy (%)**
MLP1	Gaussian Noise	99.18
MLP2	Batch Normalization	98.78
MLP3	Gaussian Noise Batch Normalization	99.42

Table 1: Results from MLP experiment

A modo de curiosidad, me gustaría mostrar el comportamiento que ha mostrado la red MLP3 en el proceso de entrenamiento.

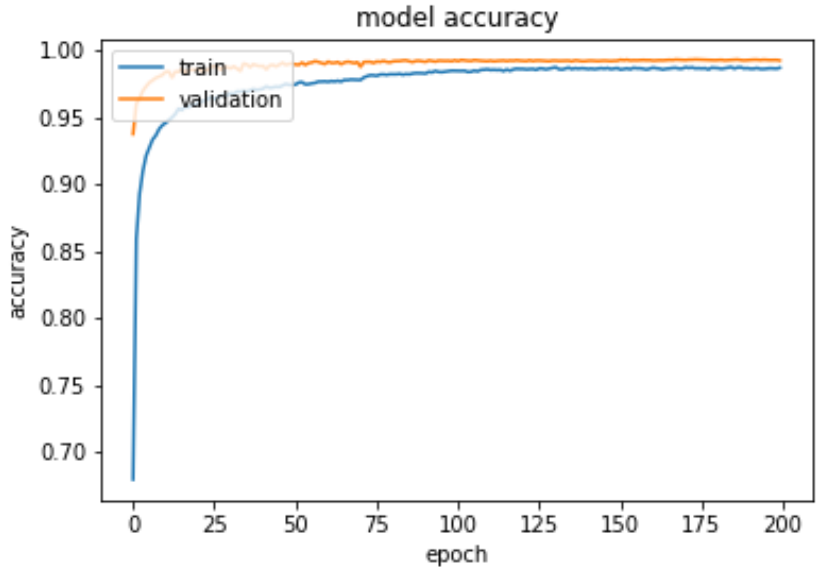

Figure 3: Training process of MLP3

Combinando estas las capas de *gaussian noise* y *batch normalization* con la técnica de data augmentation se ha conseguido evitar el *overfitting* hasta el punto de obtener mejores resultados con el conjunto de test que con el de entrenamiento.

3 Convolutional Neural Networks

En la segunda parte de la practica se han entrenado dos redes convolucionales. Para trabajar con imágenes de mayor complejidad, las CNN vienen muy bien a la hora de realizar tareas como por ejemplo clasificación. La estructura de la red que se ha empleado es la siguiente.

Concretamente la red consta de cinco bloques convolucionales, cada uno de estos bloques esta compuesto por una capa convolucional seguida de *batch normalization* y ruido gausiano, otra capa convolucional seguida de *batch normalization* y ruido gausiano de nuevo, y un operador de max pooling al final. Tras estos cinco bloques convolucionales, la red cuenta con una capa formada por 512 unidades y función de activación relu y la salida viene dada por una capa con función softmax que definirá las probabilidades de pertenecer a cada una de las diez clases existentes en CIFAR10.

Los experimentos con esta red neuronal se han configurado de la misma forma que el experimento anterior. En este experimento con redes convolucionales lo que se ha intentado ha sido mejorar la *accuracy* del modelo mejorando el proceso de *data augmentation* mediante la técnica de mixup. Esta técnica consiste en mezclar las clases de las imágenes de entrada y por lo tanto, una imagen puede ser de dos o más clases a la vez en sus debidas proporciones. En la Table 2 podemos observar los resultados obtenidos con ambas redes.

Neural	**Data Augmentation**	**Accuracy (%)**
CNN1	Basic	90.44
CNN2	Mixup	90.75

Table 2: Results from CNN experiment

Del mismo modo que en el experimento anterior, al añadir el mixup al proceso de *data augmentation*, se ha conseguido que la precisión del modelo con el conjunto de prueba supere a la precisión en el conjunto de entrenamiento. Lo que quiere decir que se ha podido controlar el problema del *overfitting*.

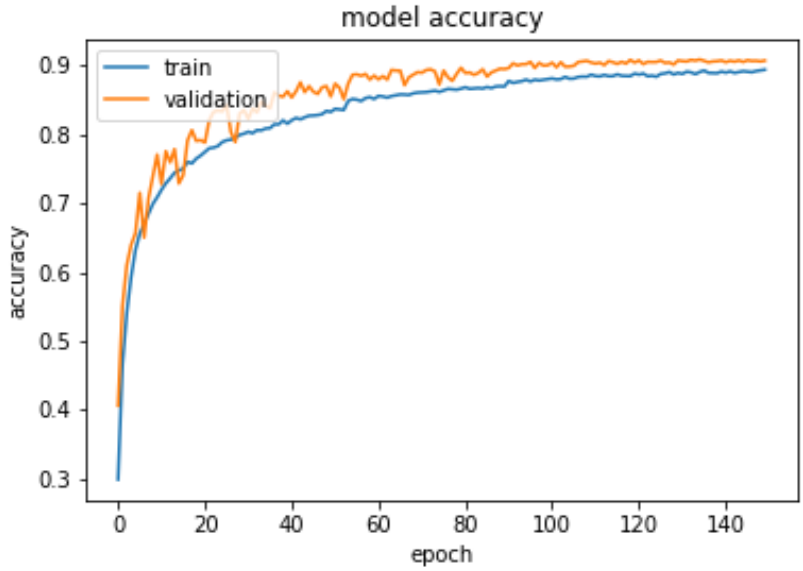

Figure 4: Training process of CNN2

4 Conclusion

Con la realización de esta practica ha sido posible emplear gran parte de las técnicas aprendidas en la teoría. También se ha visto la repercusión que puede tener en el modelo el uso de *data augmentation* en el caso de disponer de pocos datos así como hacer uso de técnicas como *batch normalization* para regularizar las salidas y acelerar el entrenamiento. Otra técnica que ha sido determinante en ambos experimentos para alcanzar la precisión requerida para la máxima nota ha sido la aplicación de ruido gaussiano entre capas evitando así la convergencia prematura de la red.