Uso de Redes Neuronales para MNIST y CIFAR

Toni Blasco Calafat
Universitat Politècnica de València
MIARFID

1. Introducción

Para la presente memoria se ha realizado una serie de experimentos sobre dos conjuntos de datos mediante técnicas de aprendizaje automático con las redes neuronales. Para ello, primeramente se ha realizado una red neuronal simple y posteriormente se ha ido incrementando el número de modificaciones de la red viendo como se comporta con el objetivo de maximizar el resultado de test.

Los resultados obtenidos han sido fruto de aplicar las redes neuronales a los conjuntos de datos de MNIST y CIFAR. En el primer caso, la base de datos MNIST, es una gran base de datos de dígitos manuscritos que se utiliza comúnmente para la capacitación de varios sistemas de procesamiento de imágenes y las pruebas en el campo del aprendizaje automático. En el caso de CIFAR, es un conjunto de datos utilizado para el reconocimiento de objetos en imágenes. En ambos casos se tiene:

1. 60.000 imágenes
2. 50.000 imágenes de entrenamiento
3. 10.000 de test

2. Primer acercamiento

En esta sección se comenta como se ha abordado el problema en el caso de ambos datasets para un primer acercamiento.

2.1. MNIST

En el caso de MNIST se ha creado un MLP con una capa oculta de 512 entradas tanto en la capa de entrada como en la capa oculta, ambas también con una función de activación relu y finalmente la capa de salida con un número de neuronas igual al número de clases, en este caso 10, y función de activación softmax para la clasificación.

El entrenamiento se ha realizado con un batch size de 128 imágenes y con 25 epochs, obteniendo así un 99.96 % de accuracy en entrenamiento y un 98.13 % en validación.

2.1.1. Variaciones del MLP

Una vez realizado esa primera versión se ha intentado variar parámetros de esa primera versión sin utilizar otro tipo de técnicas intentando aumentar el tanto por cierto de validación. Para ello, se ha variado el tamaño de batch size, el número de epochs o incluso se ha aumentado el número de capas.

2.2. CIFAR

Por otro lado, en el dataset de CIFAR se ha hecho uso de redes neuronales convolucionales(CNN) para la clasificación. Se ha partido de una CNN con 5 capas de convolución inicialmente con la secuencia de 32,64,128,256,512 neuronas y posteriormente se han añadido 512 neuronas en el allanamiento para la parte fully-connected. Para completar los parámetros requeridos por las CNN, se ha añadido en todas las capas un max-pooling de 2x2, además de un same padding y filtros kernel de convolución de 3x3. Añadir que a diferencia que en el caso anterior, si se ha hecho inicialmente uso de Batch Normalization y del

Gaussian Noise.

El entrenamiento ha sido realizado con un batch size de 100 y con 75 epochs, obteniendo con ello un accuracy de 99.57 % en entrenamiento y un 78.42 % en test. Dado que el resultado dista mucho de los objetivos se realizan una serie de cambios en las redes con el objetivo de mejorar el resultado.

3. Ampliación del modelo

Una vez realizado esa primera versión se ha intentado variar parámetros de esa primera versión sin utilizar otro tipo de técnicas intentando aumentar el tanto por cierto de validación. Para ello, se ha variado el tamaño de batch size, el número de epochs o incluso se ha aumentado el número de capas.

3.1. MNIST

Para el caso de MNIST se han añadido dos capas ocultas, pasando a tener 3 capas ocultas donde en la capa de entrada se tienen 1024 neuronas, mientras que, la segunda y tercera tienen 512 y la cuarta capa 128 neuronas, todas ellas con función de activación relu. Finalizando como en el caso anterior, en la capa de salida con 10 neuronas y una softmax.

Este crecimiento de la red neuronal obtiene unos resultados de 100 % de accuracy en entrenamiento, mientras que, en test se encuentra un accuracy de 98.22 %. Estos resultados indican que la red neuronal es capaz de aprenderse todos los datos de test, pero que cuando vienen datos nuevos no es capaz de generalizar correctamente y por eso vemos que aunque la red haya aumentado considerablemente y tiene un 100 % en entrenamiento no hay apenas diferencia con el caso anterior en la validación.

En el caso de augmentar en número de batch size o el número de epochs se obtiene el mismo resultado y por tanto se ha optado por aplicar otro tipo de técnicas con la intención de aumentar el valor en validación.

3.2. CIFAR

En el caso de CIFAR se ha hecho uso del data augmentation para intentar obtener mejores resultados. Además también se ha añadido una función callback como en el caso de MNIST con diferentes Learning Rate dependiendo de la iteración. El resultado obtenido es de 85.88 % en entrenamiento y de 84.02 % en test, dado que no consigue el objetivo de la práctica se aplica un aumento de la red neuronal pasando a tener otra capa convolucional a la inicial con otros filtros kernels de 3x3 padding same, y a esa capa añadirle Batch Normalization junto con Gaussian Noise (se entra en detalle que son estás técnicas en la siguiente sección) con función de activación Relu, y maxpooling de 2 x 2. Pasando a tener un resultado de 87.89 % en entrenamiento y 86.86 % teniendo menos de un 15 % de error, para conseguir mejores resultados se comenta en la siguiente sección cuales han sido las estrategias aplicadas.

4. Uso de técnicas de regulalización

En esta sección se añaden técnicas de batch normalization, regularizaciones, learning rate, data augmentation y otro tipo de técnicas como dropout. A continuación se detalla como se añaden y que resultados obtiene la red con la inclusión de estas técnicas.

4.1. MNIST

En el caso de MNIST se ha optado por una red neuronal de una capa de entrada de 1024 neuronas con función de activación relu y dos capas ocultas tambien con 1024 neuronas y funcion de activación relu, finalizando como en los enteriores casos con la capa de salida de 10 neuronas y softmax.

4.1.1. BATCH NORMALIZATION

Este parámetro nos permite entrenar la red de forma más rápida y estable, se encarga de centrar y normalizar cada mini-batch que le llega a nuestra red con una media y desviación estándar calculadas con el mini-batch. Los resultados obtenidos han sido de 98.54 % validación y 100 % en test observamos un incremento del porcentaje de válidación pero sigue siendo todavía muy poco respecto al caso anterior.

4.1.2. GaussianNoise

A continuación se añade a las capas de la red neuronal un Gaussian Noise después de aplicarse un batch normalization. El ruido gaussiano es un ruido estadístico con una distribución gaussiana (normal). Significa que los valores de ruido se distribuyen de forma gaussiana normal. El ruido gaussiano se añade a la imagen original. Se han obtenido unos resultados de 98.33 % test 98.57 % validación. Se observa como el añadir este parámetro al entrenamiento hace que aunque no se aprenda todo el test, obteniendo un resultado algo menor que en los casos anteriores en el test pero, sin embargo, mayor porcentaje en validación.

Posteriormente, se ha añadido delante del batch normalization obteniendo ahora unos resultados de 97.78 % en entrenamiento y 98.37 % en test, viendo como los resultados han sido mejores en el caso anterior se ha decidido escoger la otra modalidad para el siguiente paso.

4.1.3. Learning-Rate

Pasamos ahora a añadir a nuestra red neuronal una tasa de aprendizaje o learning rate. La tasa de aprendizaje es un hiperparámetro que controla cuánto cambiar el modelo en respuesta al error estimado cada vez que se actualizan los pesos del modelo. Elegir la tasa de aprendizaje es un reto, ya que un valor demasiado pequeño puede dar lugar a un proceso de entrenamiento largo que podría atascarse, mientras que un valor demasiado grande puede provocar el aprendizaje de un conjunto subóptimo de pesos demasiado rápido o un proceso de entrenamiento inestable.

La estrategia utilizada ha sido la de variar la tasa de aprendizaje, primeramente se ha hecho con un LR de 0.1, posteriormente uno de 0.01 y finalmente otro de 0.001. En la tabla se muestran los resultados:

LR	acc TEST	acc VAL
0.1	98.80 %	98.64 %
0.01	98.19 %	98.45 %
0.001	95.39 %	97.04 %

Cuadro 1: Resultados con diferentes variaciones de parámetros.

Se pueden observar grandes variaciones en el resultado final, se observa como en el primer caso se obtiene mayor porcentaje y por tanto se hará uso de ese valor para la siguiente experimentación.

4.1.4. Data augmentation

En este caso se hace uso del aumento de datos para obtención de resultados, la estrategia seguida ha sido realizar varias trasformaciones afines con el objetivo de mejorar los resultados previos. A continuación se muestra la transformación realizada y el resultado.

Primera transformación En este caso se ha añadido una amplitud de la imagen con un width shift range de 0.1, un height shift range de 0.1, y un horizontal flip=False, obteniéndose un 95.88 % test y un 98.66 % en validación

Segunda transformación En esta transformación se ha añadido un valor de rotación de 90 sobre los otras afines anteriormente dichas. En este caso los resultados obtenidos son de 89.76 % en test y 96.46 % en validación.

Rotación	acc TEST	acc VAL
0	95.88 %	98.66 %
90	89.76 %	96.46 %
5	0.9582 %	0.9888 %
10	0.9557 %	0.9878 %

Cuadro 2: Resultados con diferentes variaciones de parámetros.

4.1.5. Variación final

Viendo que no se llega al objetivo de la práctica, se ha decidido juntar todas las variaciones anteriores y realizar diversos cambios con el objetivo de obtener mejores resultados, a continuación se detalla cuales son

los cambios realizados. Finalmente se ha decidido realizar una variación ya con todas las modificaciones hechas y escogidas probando diferentes tipos, como el tipo de batch y número de epochs viendo como se comporta, a continuación en la tabla se muestran las experimentaciones.

LR	Batch	acc TEST	acc VAL
0.1	50	98.80 %	98.64 %
0.01	50	98.19 %	98.45 %
0.001	100	95.39 %	97.04 %

Cuadro 3: Resultados con diferentes variaciones de parámetros.

En el primer modelo con 50 iteraciones y la red reducida, es decir, 3 capas ocultas pero con solo 512 neuronas por capa, obteniéndose un 96.68 % en test y un 98.95 % en validación, viendo que no se consiguen los resultados esperados se opta por otro modelo.

Se opta por variar el ruido gaussiano de la red reducida, en este caso se pasa a un 0.2 en la capa intermedia y un 0.1 en la capa final, se entrena con 50 epochs como en el caso anterior y se obtiene un 97.96 % en test y 99.18 % en validación.

Se pasa ahora a una red ampliada, de 1024 neuronas por capa, con 100 epochs y con el mismo tamaño de batch de 100 y con los parámetros anteriores obteniéndose un 98.08 % en test y un 99.18 % en validación. Aunque obtiene un mejor resultado sigue sin obtenerse los mejores resultados por lo que se pasa a entrenar otra red neuronal.

Con una red con un ruido gausiano de 0.1 en todas sus capas se obtiene un 98.29 % de accuracy en test y 99.14 % en validación.

Viendo que no es suficiente se pasa ahora a eliminar la parte de rotación del data augmentation y con los mismos parámetros que los anteriores(100 iteraciones y 100 de batch size junto a 3 capas ocultas de 1024) se obtiene un 99.19 % en test y un 99.28 % en validación. Es el mejor resultado hasta el momento pero se ha intentado afinar mas en la red, hasta que con un momentum de 0.5 en el regularizador y 200 epochs y con una reducción en el batch size pasando de 100 a 64 se ha obtenido un 99.35 % en test y 99.31 % en validación, con estos valores se obtiene menos de un 7 % de error que era el objetivo principal de la parte de MNIST

4.2. CIFAR

Para el caso de CIFAR se han realizado diversas modificaciones para poder llegar al 10 % de error en validación. Inicialmente, se ha aumentado el número de cambios en el Data augmentation, se ha incluido un Datagen tanto para entrenamiento como para validación, posteriormente se ha cambiado el optimizador pasando de un SGD a un Adam, aunque también se han hecho pruebas con un RMSProp sin dar mejores resultados, se ha variado para diversos Learning-rate, hasta finalmente quedarse con un LR de 0.001, además se ha cambiado la forma de variar en LR en entrenamieto, pasando a usar un ReduceLROnPlateau, con un min_lr=0.005 y factor=0.5, además se ha aumentado el número de epochs a 250 y con uso de un batch size = 128, conseguiendo finalmente un resultado de 97.55 % de accuracy en entrenamiento y un 90.05 % en validación. Remarcar que la mayor diferencia para obtener este resultado ha sido la inclusión del Data augmentation, en el set de validación y en el de entrenamiento.

Optimizador	LR	min_lr	train acc	validation acc
Adam	1e-3	0.00005	0.9766	90.74 %
Adam	1e-4	0.000005	0.9806	85.96 %
Adagrad	1e-3	0.00005	87.6 %	76.7 %

Cuadro 4: Resultados con diferentes variaciones de parámetros.

Con estos resultados se ha obtenido el mejor resultado de todas las experimentaciones realizadas, remarcar que aunque se han realizado diversas modificaciones, como la variación en los parámetros del data augmentation, tanto del rotate como del horizontal shift, además de variaciones de optimizadores con diferentes parametros, RMSProp y SGD entre otros, no se ha conseguido conseguir el 8 % de error requerido en la práctica, se indican como mejoras futuras la inclusión de redes neuronales residuales y técnicas como el cut-out para el data augmentation.