Práctica 5

Inteligencia Artificial

Pedro Tamargo Allué (758267)
27-12-2019

Resultados de entrenamiento de redes neuronales con Keras

Para la realización de la práctica se ha entrenado distintos tipos de redes neuronales, con diferente topología.

Perceptrón de salida

En el código de ejemplo proporcionado se puede observar que se han realizado pruebas con un Perceptrón con función de activación sigmoidal, su tasa de acierto en entrenamiento es del 90% y la de acierto en validación de 90.68%, no habiéndose producido sobreajuste.

Sobre el Perceptrón anterior se ha probado a cambiar su función de activación a una función softmax, consiguiendo tasas de acierto en entrenamiento de 90.16% y 90.82% en validación.

También, se ha realizado una prueba con una función de activación ReLU, pero como el Perceptrón es una capa de salida, esta no funciona bien, ya que está diseñada como función de transición. Sus resultados en entrenamiento son 55.3% y 54.25% en validación.

Perceptrón con una capa oculta

Dado que los resultados anteriores no nos permitían obtener una precisión mayor al 91% en validación, se ha estudiado la posibilidad de obtener mayor precisión con un perceptrón con una capa oculta.

Como método experimental para la elección de funciones de activación para el desarrollo de las pruebas, se van a estudiar diferentes combinaciones de las funciones ReLU, softmax y sigmoidal en las distintas capas de la red.

Se ha estudiado una red neuronal con 10 neuronas en la capa oculta y 10 en la capa de salida y con funciones de activación sigmoidal en ambas capas. Sus resultados han sido 82.59% en entrenamiento y 83.36% en validación.

Se han estudiado los resultados de una red neuronal idéntica en neuronas a la anterior, cambiando la función de activación de la capa oculta a ReLU, manteniendo la salida sigmoidal. Los resultados han sido 9.87% en entrenamiento y 9.8% en validación.

Como última prueba de las funciones de activación, se ha utilizado la misma red neuronal definida anteriormente, manteniendo la función de activación ReLU en la capa oculta y la función softmax en la de salida. Los resultados han sido 91.57% en entrenamiento y 91.74% en validación.

Todas las redes estudiadas a continuación van a utilizar función ReLU en la capa oculta y softmax en la capa de salida, ya que en el estudio previo se ha demostrado ser la combinación que mejores resultados ha proporcionado.

En la siguiente tabla podemos observar los resultados de utilizar las funciones de activación descritas anteriormente sobre la red neuronal de una capa oculta, variando el número de neuronas de la capa oculta.

Tipo de red	% acierto entrenamiento	% acierto validación
Perceptron con capa oculta (10) 'relu' y salida 'softmax'	91,57	91,74
Perceptron con capa oculta (30) 'relu' y salida 'softmax'	92,59	92,87
Perceptron con capa oculta (50) 'relu' y salida 'softmax'	92,93	92,89
Perceptron con capa oculta (70) 'relu' y salida 'softmax'	93,28	93,32
Perceptron con capa oculta (90) 'relu' y salida 'softmax'	93,22	93,23
Perceptron con capa oculta (100) 'relu' y salida 'softmax'	93,34	93,38
Perceptron con capa oculta (500) 'relu' y salida 'softmax'	93,86	93,87
Perceptron con capa oculta (750) 'relu' y salida 'softmax'	94,12	94,13

Tabla que contiene los resultados del entrenamiento de las redes neuronales descritas.

Se puede apreciar que conforme se aumentan el número de neuronas en la capa oculta, la red proporciona mayor precisión en la validación. También, se puede observar que no existe sobreajuste en esta red ya que los resultados de entrenamiento y los de validación son muy parecidos.

Perceptrón con dos capas ocultas

Tras la realización de las pruebas anteriores podemos observar que los resultados obtenidos mejoran respecto a los de un perceptrón de una sola capa (salida), no obstante, la máxima precisión que puede obtener la red neuronal en validación es menor que 95%, por lo tanto, se va a proceder a realizar pruebas con un perceptrón con dos capas ocultas con funciones de activación ReLU, ya que se ha demostrado anteriormente que obtenían mejores resultados en capas ocultas que la función sigmoidal.

La siguiente tabla contiene los resultados de realizar distintas pruebas variando el número de neuronas en las capas ocultas.

Ncapa 1	Ncapa2	NcapaSalida	% entren.	% validacion	Variacion
10	10	10	92	92,04	0,04
10	50	10	92,31	92,46	0,15
10	90	10	92,31	92,55	0,24
10	100	10	92,3	92,46	0,16
30	10	10	93,95	93,74	0,21
30	70	10	93,81	93,78	0,03
30	100	10	94,04	94,05	0,01
50	70	10	94,13	93,83	0,3
50	100	10	94,6	94,45	0,15
70	70	10	94,55	94,36	0,19
70	100	10	94,55	94,44	0,11
90	50	10	94,58	94,46	0,12
90	70	10	94,83	94,64	0,19
90	100	10	94,84	94,69	0,15
100	100	10	94,72	94,65	0,07

Se puede apreciar que, conforme aumenta el número de neuronas de ambas capas, obtenemos una mayor precisión en validación, no obstante, también podemos remarcar que existe cierto sobreajuste en la red, ya que cada vez la distancia entre la precisión en entrenamiento y la de validación aumenta, siendo de un 0.3% su máxima en la tabla anterior (caso 50,70,10).

Para intentar reducir este sobreajuste, se han aplicado técnicas como earlystopping y dropout sobre los casos (90,70,10) [caso 1] y (90,100,10) [caso2], ya que estos han sido los que mayor porcentaje de acierto han obtenido y presentan sobreajuste.

Aplicando earlystopping con $patience = 5$ con algoritmo de optimización SGD y función de coste categorical_crossentropy, en 20 épocas, obtenemos los siguientes resultados:

Caso	% acierto entrenamiento	% acierto test
Caso 1	94.76	94.62
Caso 2	94.72	94.55

Aplicando dropout sobre las distintas capas obtenemos los siguientes resultados:

Caso	Capas Dropout	% acierto entren.	% acierto test
Caso 1	1	95	94.86
Caso 1	2	94.58	94.43
Caso 1	1,2	94.26	94.33
Caso 2	1	94.83	94.77
Caso 2	2	94.53	94.55
Caso 2	1,2	94.54	94.62

Podemos concluir que en el caso 2, aplicando dropout en la primera capa, obtenemos un buen resultado respecto a los entrenamientos anteriores, sin sobreajuste.

Sobre el caso 2 se ha procedido a probar distintos algoritmos de optimización y de coste:

Algoritmo de optimización	Algoritmo de coste	Método para evitar sobreajuste	% acierto entrenamiento	% acierto test
SGD	MSE (Error cuadrático medio)	-	73.52	75.13
RMSProp	Categorical_crossentropy	-	99.7	97.76
RMSProp	Categorical_crossentropy	Earlystopping(5)	99.28	97.45
RMSProp	Categorical_crossentropy	Dropout(0.2,-)	99.36	97.96
RMSProp	Categorical_crossentropy	Dropout(-,0.2)	99.6	97.85
RMSProp	Categorical_crossentropy	Dropout(0.2,0.2)	99.31	97.94
RMSProp	MSE	-	99.43	97.99
Adagrad	Categorical_crossentropy	-	98.52	97.54
Adagrad	Categorical_crossentropy	Earlystopping(5)	98.95	97.63
Adagrad	Categorical_crossentropy	Dropout(0.2,-)	98.57	97.54
Adagrad	Categorical_crossentropy	Dropout(-,0.2)	98.65	97.39
Adagrad	Categorical_crossentropy	Dropout(0.2,0.2)	98.38	97.47
Adagrad	MSE	-	98.76	97.38

Se puede apreciar que, obtenemos un porcentaje de acierto con los datos de test bastante alto (entorno al 97% en todos los casos, menos SGD con MSE). Pero observamos que existe sobreajuste, aplicando las técnicas proporcionadas para reducir el sobreajuste podemos reducirlo.

Se va a proceder a estudiar sobre los casos: SGD con Dropout (0.2,-), RMSProp con Dropout (0.2,-) y Adagrad con Dropout (0.2,0.2) el efecto de trabajar con lotes ($batch$) menores, ya que hasta ahora las pruebas han sido realizadas con $batch = 128$.

Caso de estudio	Batch	% acierto entrenamiento	% acierto test
(SGD,Dropout(0.2,-))	128	94.72	94.56
(SGD,Dropout(0.2,-))	32	98.11	97.16
(SGD,Dropout(0.2,-))	16	99.06	97.53
(RMSProp, Dropout(0.2,-))	128	99.36	97.96
(RMSProp, Dropout(0.2,-))	32	99.10	97.80
(RMSProp, Dropout(0.2,-))	16	98.79	97.54
(Adagrad, Dropout(0.2,0.2))	128	98.38	97.47
(Adagrad, Dropout(0.2,0.2))	32	98.45	97.43
(Adagrad, Dropout(0.2,0.2))	16	98.5	97.47

Se puede apreciar que para el algoritmo de optimización SGD, cuanto menor es el tamaño del batch mejores resultados proporciona, aunque, mostrando sobreajuste.

Para los algoritmos RMSProp y Adagrad no obtenemos una mejora tan significativa como con el algoritmo anterior.

Redes convolucionales

Las redes convolucionales son un tipo de red neuronal que se fundamentan en la aplicación de un conjunto de filtros sucesivos sobre una entrada. Son un tipo de red muy utilizada en visión por computador, clasificación de imágenes…

Tras la ejecución del fichero proporcionado junto con el enunciado de la práctica, obtenemos los siguientes resultados de precisión (%) de la red convolucional con los datos de test: 99.23%.