Classificazione di Immagini

Francesco Bizzarri

Settembre 2022

Elaborato assegnato per l'esame finale del corso di Intelligenza Artificiale.

1 Introduzione

In questo progetto si utilizza e si analizzano le prestazioni di un classificatore multi-layered perceptron per il riconoscimento e la classificazione di immagini sul dataset CIFAR-10.

2 Il dataset

Il dataset CIFAR-10 consiste in 60000 immagini a colori di dimensioni 32x32. Ogni immagine appartiene ad una delle 10 classi:

ID	CLASSE
0	Aeroplano
1	Automobile
2	Uccello
3	Gatto
4	Cervo
5	Cane
6	Rana
7	Cavallo
8	Barca
9	Camion

Il dataset è diviso in 6 batches, 5 dedicati all'apprendimento e uno per i test. Ogni batch contiene dunque 10000 immagini. Il batch dedicato ai test contiene esattamente 1000 immagini per ogni classe. Mentre le restanti immagini sono divise casualmente tra gli altri batches. Nel singolo training batch potrebbero quindi esserci più immagini di una certa classe rispetto ad un'altra. Ma se sommati insieme contengono esattamente 5000 immagini di ogni classe.

3 Il classificatore

Per questo progetto è stato utilizzato un classificatore multi-layered perceptron disponibile nella libreria (open source) scikit-learn per il linguaggio di programmazione Python. Questa rete neurale utilizza una funzione di attivazione *softmax* per l'output layer. La funzione di attivazione per gli hidden layers è invece selezionabile nel costruttore del classificatore. Così come il resto degli iper-parametri quali, ad esempio:

- Profondità e larghezza degli hidden layers
- Learning rate
- Solver per l'ottimizzazione dei pesi
- Dimensione minibatches
- Numero di epoche per il training

L'input layer è composto da 3072 unità. Ogni unità rappresenta il valore di ogni singolo pixel dell'immagine. L'output layer invece è composto da 10 unità, una per ogni classe di possibile appartenenza.

4 Analisi dei risultati

La rete usata per per gli esperimenti è stata addestrata utilizzando tutti i train batches e ha 2 hidden layers, ognuno composto da 400 neuroni.

4.1 Accuratezza

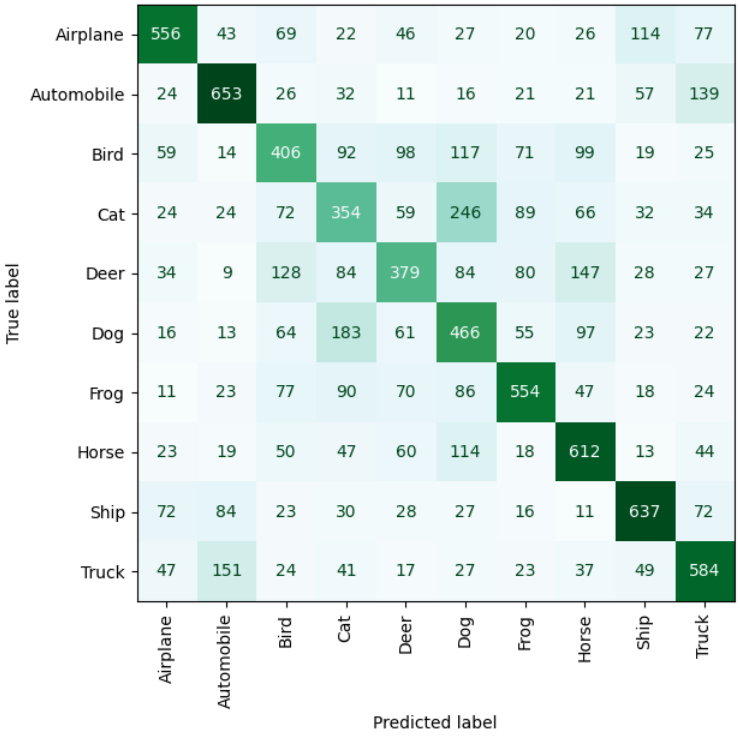
Il parametro più significativo per valutare la bontà delle predizioni è quello dell'accuratezza, ovvero la percentuale di classificazioni corrette sul test batch.

$$accuratezza : 52\%$$

É interessante analizzare anche i parametri *precision*, *recall*, e *f1-score*:

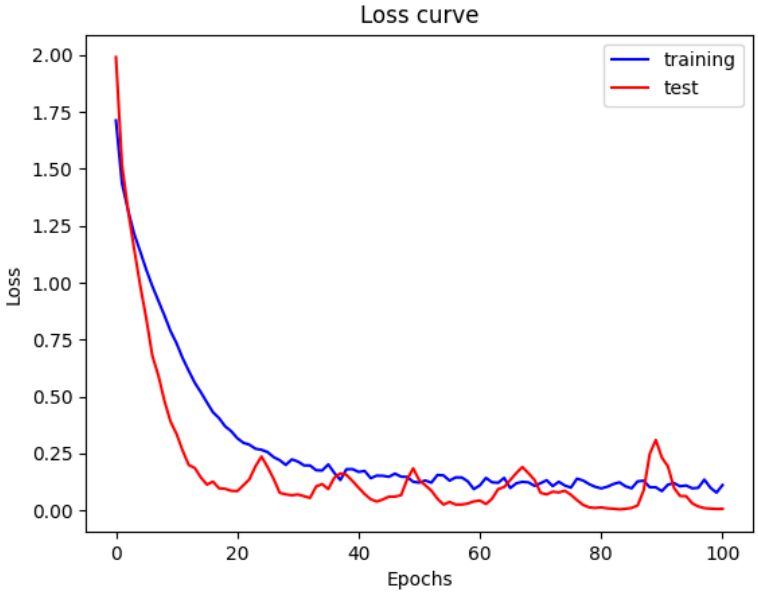
	PRECISION	RECALL	F1	ISTANZE
Aeroplano	0.64	0.56	0.60	1000
Automobile	0.63	0.65	0.64	1000
Uccello	0.43	0.41	0.42	1000
Gatto	0.36	0.35	0.36	1000
Cervo	0.46	0.38	0.41	1000
Cane	0.39	0.47	0.42	1000
Rana	0.59	0.55	0.57	1000
Cavallo	0.53	0.61	0.57	1000
Barca	0.64	0.64	0.64	1000
Camion	0.56	0.58	0.57	1000
AVG	0.52	0.52	0.52	

4.2 Matrice di confusione



Come possiamo vedere dalla matrice di confusione, gli errori più comuni commessi dalla rete sono classificare i gatti come cani e viceversa.
Performa al meglio, invece, nella classificazione di barche e automobili.

4.3 Loss



La rete neurale utilizzata per gli esperimenti è stata addestrata in 100 iterazioni. Come si vede dal grafico, però, già dopo 50 epoche non ci sono sostanziali miglioramenti.

5 Per riprodurre gli esperimenti

Questi risultati sono stati tutti ottenuti pre-processando i dati da mandare in input alla rete. Infatti i valori dei pixel sono stati normalizzati in modo da avere $\mu = 0$ e $\sigma = 1$. Senza normalizzazione la stessa rete, a parità di parametri, non supera di molto l'accuratezza del 10%, ovvero non classifica meglio di un classificatore casuale.
É stata utilizzata la funzione di attivazione ReLu per gli hidden layers e il solver Adam per l'ottimizzazione. Tutto il codice utilizzato è disponibile qui.

6 Conclusioni

Lavorare su questo progetto mi ha fatto imparare molte cose. Intanto ho capito quanto la normalizzazione dei dati sia fondamentale quando si lavora con reti

neurali. In secondo luogo mi è stato utile a prendere dimestichezza con strumenti pratici come la libreria scikit-learn.
Ho trovato i risultati sperimentali molto interessanti, in particolare è stato curioso vedere come l'errore più comune della rete sia confondere cane e gatto, che in effetti potrebbe rispecchiare l'errore più comune commesso da un cervello umano.

6.1 Problemi e miglioramenti

Credo che il multi-layered perceptron non sia lo strumento migliore per problemi di classificazioni di immagini. Intanto perché, come i risultati sperimentali dimostrano, l'accuratezza non è alta.
L'assenza di layers di pooling potrebbe essere in generale problematica in caso di dataset con immagini più grandi (ricordiamo che in questo caso le immagini erano molto piccole, 32x32). Inoltre reti con layers convoluzionali si prestano meglio a risolvere problemi di computer vision come questo perché possono essere spazio-invarianti. Ovvero possono classificare oggetti presenti nell'immagine indipendentemente dalla loro posizione o grandezza. Per come è architetturata la rete utilizzata in questo progetto invece la causa della bassa accuratezza potrebbe essere ricondotta anche alla diversa posizione dei soggetti da classificare all'interno delle immagini del dataset. Per curiosità è stata testata la stessa rete anche su un problema di classificazione diverso, sul celebre dataset Iris. In questo caso il training non è effettuato con immagini, ma con dati tabulari. In questo contesto la rete classifica con una accuratezza prossima al 100%.