UNIVERSITÀ DEGLI STUDI DI CATANIA
DIPARTIMENTO DI MATEMATICA ED INFORMATICA

Relazione progetto Machine Learning

BINARY CLASSIFICATION USING ALEXNET

STUDENTI:

Lorenzo **ABBATE**
Cristian **DANIELE**

Anno Accademico 2019/2020

Capitolo 1

Introduzione

Il progetto nasce con lo scopo di costruire una rete neurale in grado di riconoscere, presa in input la foto di un desk, se su quel desk è stato tolto o spostato un qualsiasi oggetto.
La rete potrebbe avere ampio impiego in ambito forense, per vedere se un oggetto è stato rimosso e spostato dalla scena di un crimine ad esempio.
La rete è quindi stata allenata su due dataset, un primo contenente solo immagini di desk *integri*, senza quindi manomissioni. Un secondo dataset è stato costruito inserendo solo immagini di desk *manomessi*. Sono quindi stati spostati di volta in volta vari oggetti.
E' stato utilizzato il modello *alexnet* fornito da pythorch, non preallenato.

Capitolo 2

Ambiente di sviluppo

L'intero progetto è stato sviluppato su *colab* in quanto ci ha permesso di collaborare facilmente ed di aver accesso ad hardwar specifico per il machinelearning. In partiocolar modo la scheda video che ci è stata riservata per il training è stata una
Tesla P100-PCIE-16GB (UUID: GPU-484503e4-6d64-c69f-01b3-49822745b161).
Sull'ambiente *colab* è stato fatto il *mount* della cartella google drive, in modo da avere libero accesso (permanente) a file e cartelle.
Non è stato possibile utilizzare la sezione *files* all'interno di colab stesso poichè dopo qualche ora venivano cancellati tutti i dati salvati.

Capitolo 3

Realizzazione dataset

Il dataset conta di 7324 foto, suddivise nel seguente modo:

- 4882 impiegate per il training
- 2442 impiegate per il testing

Il metodo di acquisizione delle immagini è avvenuto mediante l'acquisizione di frame da video di circa un minuto ciascuno.

3.1 Acquisizione frame

Del codice python è stato scritto per prendere in input un video e salvare i frame estratti sulla cartella *Google Drive* prima montata. Nello specifico è stato utilizzato OpenCV prima per importare il video (anch'esso salvato sul drive), poi per estrarre da esso i frame, nominarli in modo adeguato e salvarli all'interno del drive.
Sono state create quindi due cartelle (una per ogni classe) contenenti rispettivamente le immagini del desk integro e di quello alterato.

3.2 Dataset integro

Un primo dataset è stato creato riprendendo il desk integro.Una *label* 0 è stata fatta corrispondere al desk con tutti gli oggetti presenti nella stessa posizione. Il video è stato acquisito con diverse condizioni di luminosità, in modo da rendere più vario il dataset.

3.3 Dataset alterato

In questo caso sono stati girati più video. Ogni video era diverso dall'altro per posizione degli oggetti, o mancanza di alcuni di essi. Questa volta una *label* 1 identificava i desk appartenenti a questa classe.
Anche in questo caso le acquisizioni sono state fatte con diverse condizioni di luce, in modo da variare il più possibile il dataset.

Capitolo 4

Preparazione dei dati e del modello

Una volta costruito il dataset sono state applicate delle modifiche alle immagini per rendere il training più efficace.
Si è reso inoltre necessara una modifica al modello, che vedremo in seguito.

4.1 Preparazione dei dati

Le manipolazioni alle immagini hanno previsto le seguenti fasi:

- RandomHorizontalFlip: ogni immagine, con probabilità di 1/2, è stata ruotata lungo l'asse delle ordinate.
- Resize: è stato fatto un resize di ogni immagine di 256px
- RandomCrop: ogni immagine è stata ritagliata in maniera random in una più piccola di dimensione di 224px
- Normalize: alexnet richiede immagini a colori normalizzate sottraendo determinati valori medi RGB e dividendo per determinate deviazioni standard. Questa normalizzazione si ottiene in pratica applicando una trasformazione particolare.

4.2 Scelta del modello

Il modello da noi scelto per affrontare il problema è stato *Alexnet*, rete convoluzionale creata da Krizhevsky, e pubblicata con Ilya Sutskever Geizrey Hinton.
La rete conta di 8 livelli, i primi 5 di tipo convoluzionale, gli ultimi 3 di tipo *fully connected*. La funzione di attivazione scelta è stata la ReLU.

4.3 Modifica ad alexnet

Alexnet, per sua natura, prevede come ultimo layer un softmax che classifichi 1000 classi. Il nostro problema era però basato sulla classificazione di due classi.
Abbiamo dovuto ridefinire l'ultimo layer di tipo *Linear* per restituire 2 valori nel seguente modo:

model.classifier[6]=nn.Linear(4096,2)

Capitolo 5

Esperimento

Il modello è stato allenato su 150 epoche, utilizzando una batch size di 24 immagini e 6 thread. Il training è durato circa 3 ore. I risultati ottenuti, nonostante non avessimo utilizzato il modello preallenato sono stati ottimi, abbiamo infatti ottenuto un'accuracy del 98.98%. Di seguito venngono riportati i grafici prodotti durante la fase di training. Nella figura 5.1 notiamo come la funzione di loss scende all'aumentare delle iterazioni. Lo stesso andamento lo notiamo in figura 5.2, per quanto riguarda il test set. Nelle figure 5.3 - 5.4 possiamo vedere come l'accuracy si comporta e nella figura 5.5 si possono vedere riassunti i valori di True Positive, True Negative, False Positive, False Negative.

5.1 Conclusioni

Siamo molto soddisfatti del risultato ottenuto.
Sfruttando una rete già esistente, ma allenandola completamente sul nostro dataset abbiamo ottenuto un'accuracy molto alta. Questo potrebbe essere dovuto ad un leggero overfitting causato dal metodo di acquisizione delle immagini.
Catturare le immagini da video, ha infatti lo sconveniente di creare diverse foto simili fra di loro. Si potrebbe pensare di acquisire immagini singolarmente, in modo da eumentare l'entropia del dataset, e in modo da rendere più difficile l'apprendimento della rete.

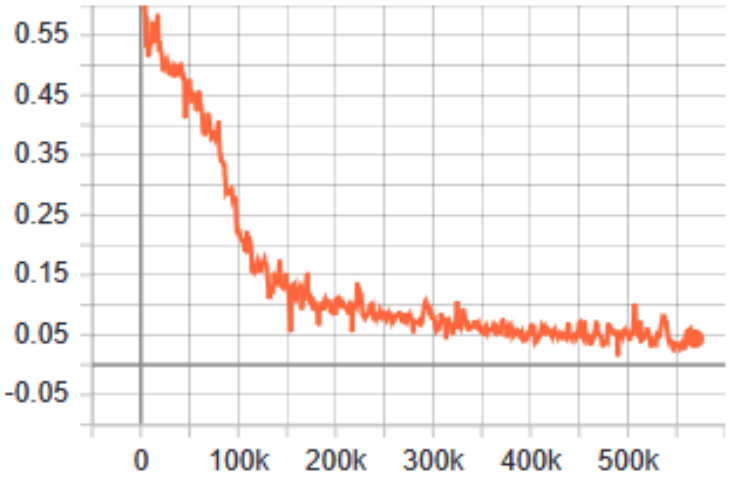

Figura 5.1: Andamento della funzuone di loss nel training set

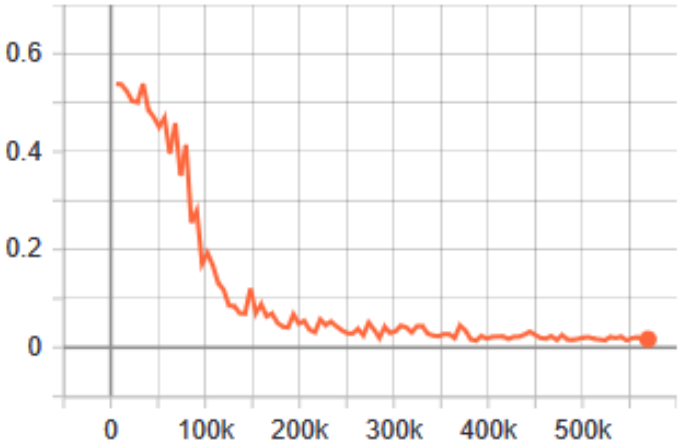

Figura 5.2: Andamento della funzuone di loss nel test set

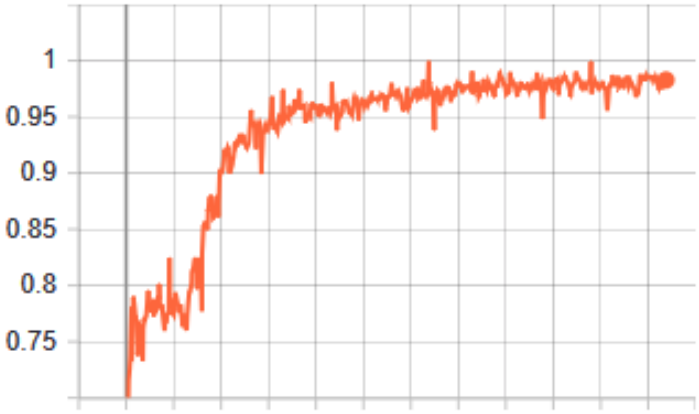

Figura 5.3: Andamento dell'accuracy nel training set

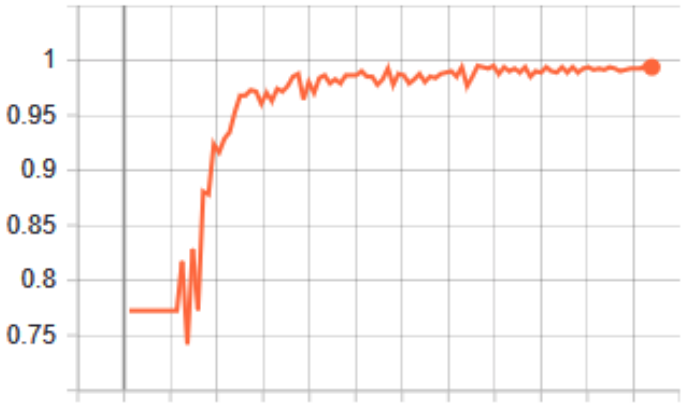

Figura 5.4: Andamento dell'accuracy nel test set

```
Accuracy di Alexnet: 99.14%
true +  0.98984375

true -  1.0

false +  0.0

false -  0.036011080332409975
```

Figura 5.5: Valori per la valutazione della bontà del modello

Capitolo 6

Demo

E' stata realizzata una demo che permette di accedere facilmente ai metodi creati per allenare e testare la rete. Si ha inoltre la possibilità di allenare nuovamente il modello (definendo learning rate, e numero di epoche), o caricare il modello da noi precedenetmente allenato.
E' possibile quindi testare il modello sul relativo dataset di test. La classe prevede inoltre un metodo che, preso in input il path di un'immagine, restituisce l'immagine stessa e la label predetta. In figura 6.1 possiamo vedere i risultati di una simulazione di prova, allenando una rete su 2 sole epoche.
La label predetta è quella corretta, con accuracy del 72.93%.

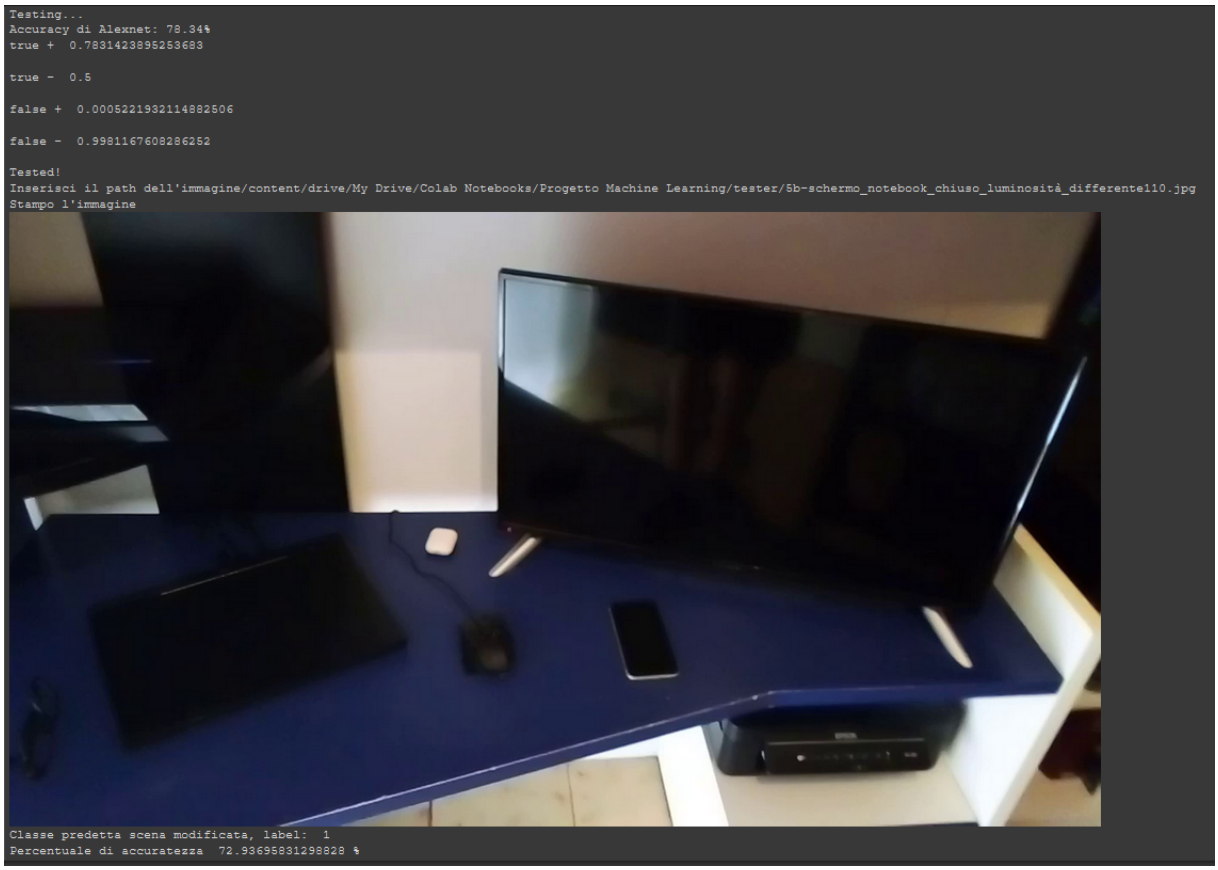

Figura 6.1: Screenshot demo