Emotion Recognition with CNNs
Deep Learning Assignment

Andrea Mansi UniUD - 137857
Christian Cagnoni UniUD - 137690

I° Semestre 2021/2022

Contents

1 Introduzione

In tale report verrà brevemente descritto il processo seguito e riportati i risultati ottenuti. È importante segnalare che nel codice sorgente fornito sono state escluse le porzioni di codice riguardanti le varie prove alternative, ritenute inferiori in termini di performance. È quindi inclusa solo la versione finale del codice (e del modello).

2 Data Preprocessing

Dopo una superficiale analisi del dataset di partenza è possibile notare chiaramente come le varie classi (emozioni) siano sbilanciate. A tal proposito si è supposto che questo potesse essere un problema. Per ovviare a questo problema è stato deciso di tentare di mitigare questo problema eseguendo una serie di trasformazioni al dataset. Nello specifico si sono provate le seguenti soluzioni:

1. Estendere (tramite trasformazioni) il dataset facendo combaciare le frequenze assolute di ogni classe (applicando trasformazioni a quelle classi presenti in quantità inferiore). Sfortunatamente questo approccio non ha portato a migliorie (in termini di accuracy), nonostante si ipotizzasse che così facendo si mitigasse la bassa accuracy per le classi meno presenti.

2. Ridurre (tramite scarto randomico) il dataset facendo combaciare le frequenze assolute di ogni classe a quella della classe meno presente. L'ipotesi alla base di questo tentativo era che lavorare su una porzione inferiore del dataset bilanciata (però composta solo da immagini iniziali) potesse migliorare il risultato. In questo caso si è osservato una riduzione dell'accuracy, probabilmente causata dal fatto che è stata scartata una grossa quantità di dati dal dataset iniziale (vista l'estrema differenza tra la classe più e meno frequente).

3. Estendere tutte le classi tramite trasformazioni (mantenendo sbilanciate le classi). Questo approccio è stato considerato quello definitivo in quanto si è osservato un aumento dell'accuracy di circa 2% nel modello finale (rispetto al dataset originale).

Le trasformazioni applicate sono cambi di prospettiva e rotazioni casuali. Si noti come si ipotizza che tali trasformazioni non siano ottimali per immagini di dimensione 48x48.

La natura di tali trasformazioni porta a una perdita di informazioni: le immagini perdono dettagli e porzioni di esse diventano inutili. Si è deciso di non effettuare "crop" sulla base dell'ipotesi che si sarebbe introdotto del rumore nel dataset: immagini che non includono volti. Si potrebbero considerare e testare ulteriori trasformazioni, al fine di indivuare quali di esse siano le più efficaci.

3 Il Modello

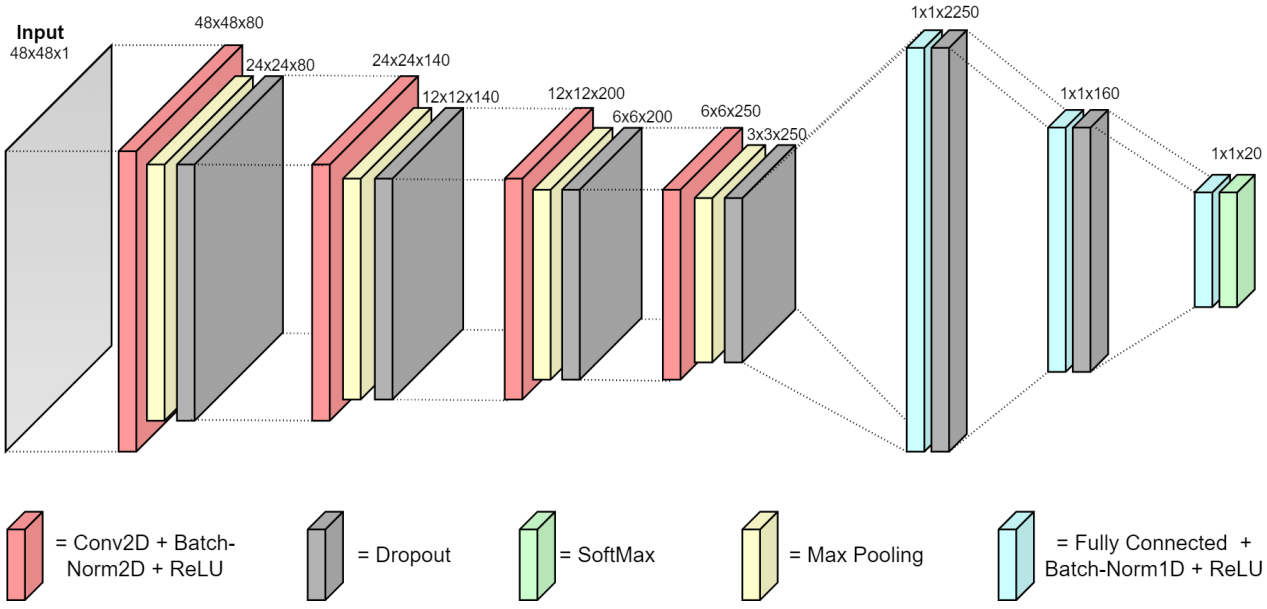
Lo sviluppo del modello è partito dall'utilizzo dell'architettura fornita dall'esercizio 6 di laboratorio, successivamente affinata. Nell'affinare il modello di partenza non ci si è completamente concentrati nel raggiungere un'accuracy più alta possibile, bensì ci si è concentrati nello sperimentare più varianti di essa. Vengono di seguito riportati i risultati del modello finale.

- **Accuracy**: 66%
- Train time: 5.4 ore; epoche: 350 (Nvidia RTX 3070Ti - Intel i7 9700K @5.0Ghz)

Segue una tabella riportante accuracy, precision e recall per le singole classi:

	Accuracy	Precision	Recall
Anger (0)	76 %	55 %	61 %
Disgust (1)	0 %	0 %	0 %
Fear (2)	45 %	59 %	43 %
Happiness (3)	97 %	83 %	94 %
Sadness (4)	52 %	55 %	50 %
Surprise (5)	94 %	81 %	90 %
Neutral(6)	75 %	58 %	71 %

Segue una rappresentazione grafica riassuntiva del modello finale:



Un'ulteriore rappresentazione più dettagliata è fornita insieme alla relazione.

Durante la fase di affinamento sono state provate diverse versioni del modello. Verranno riportate in modo coinciso alcune considerazioni e risultati per ciascuna di esse.

- Inizialmente il modello di partenza presentava un'accuracy di solo 35-40%. La prima modifica è stata aggiungere un terzo layer convolutivo + max pooling. Tale aggiunta ha portato a un'aumento delle performance. Successivamente è stata provata sia la configurazione con 4 che con 5 layer. Si è osservato un miglioramento con la versione a 4 layer convolutivi ma un peggioramento con la versione a 5. Si suppone che il peggioramento della configurazione a 5 layer sia dovuto al fatto che i tensori raggiungessero una dimensione di 1x1. Tale dimensione non rappresenta più nessun pattern che possa caratterizzare l'immagine iniziale. A tal proposito si è scelto di fissare a 4 il numero di layer convolutivi (più max pooling e ReLU).

- Un'ulteriore caratteristica su cui si è posta attenzione è stata la dimensione dei singoli layer convolutivi. Sono state provate diverse configurazioni, nello specifico si sono aumentati i numeri di canali in output di ciascun layer convolutivo fino a trovare un punto di stabilità (in termini di guadagno di accuracy).

- Relativamente ai layer fully-connected è stato seguito un approccio simile: sono state provate diverse configurazioni con diversi parametri.

- Fissato il numero di layer convolutivi e le rispettive dimensioni sono state provate diverse dimensioni per i kernel. La configurazione finale è rispettivamente 5-5-3-3. Si è notato come kernel di dimensioni 5 nei layer 3 e 4 portassero a una riduzione troppo drastica delle dimensione dell'output con un conseguente degrado delle performance. Si ipotizza che con la graduale riduzione della dimensione dell'input sia adeguato ridurre anche la dimensione del kernel per evitare che sia troppo elevata.

- Come funzione di attivazione finale è stata utilizzata una SoftMax in quanto adeguata per la classificazione multiclasse.

- L'optimizer è stato sostituito passando da SGD a Adam. Tale modifica ha comportanto un aumento di accuracy dell'1-2%.

- Al fine di mitigare il rischio di overfitting e velocizzare il tempo di train sono stati introdotti dei passi di dropout (con p=25%) come da immagine.

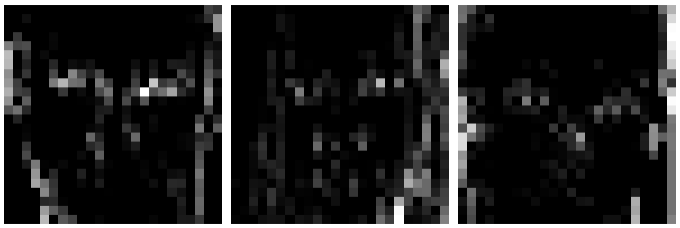
- È stato inoltre deciso di aggiungere un passaggio di normalizzazione dopo ogni layer. Questa scelta ha comportanto un notevole miglioramento della accuracy al costo di un tempo di train peggiorato considerevolmente.

- Per ogni layer convolutivo è stato aggiunto un padding al fine di mitigare la riduzione in dimensione dell'input. Tale modifica ha comportato un leggero miglioramento.

- Relativamente al tempo di train sono state provati diversi valori di batchsize e learning rate.

Al fine di visualizzare l'output dei vari layer della rete è stata definita una funzione apposita. Seguono tre esempi di immagini ritenute significative.

- Tre immagini 24x24 in output dopo il primo layer convolutivo e il primo max-pooling. È possibile notare come le immagini evidenziano i lineamenti dei volti nell'immagine in input.



4 Conclusioni e Possibili Migliorie

Ci riteniamo relativamente soddisfatti delle performance ottenute dal modello, nonostante il livello di accuracy non sia molto elevato. Il risultato risulta abbastanza in linea con i lavori pubblicati da parte di altri autori sulla relativa pagina di Kaggle (60-80%).

L'unica grave problematica risulta la totale inutilità del modello nel riconoscere la classe "Disgust". Tale problematica è sicuramente causata dal fatto che il dataset contiene poche immagini appartenenti a tale categoria, di conseguenza esse vengono probabilmente trattate come del rumore da parte della rete. Questa problematica potrebbe venir risolta adottando un bilanciamento del dataset più intelligente (si noti che le alternative provate e descritte in precedenza hanno migliorato tale situazione ma nel complesso la rete risultava meno performante).

Analizzando a posteriori il risultato ottenuto seguono alcune considerazioni su ciò che si ritiene migliorabile:

- Si pensa che concentrandosi su una fase di pre-processing più corposa e intelligente dell'input possa beneficiare le performance finali. Come annunciato in precedenza si potrebbero valutare e testare diverse trasformazioni. Nello specifico si ipotizza che una fase di pre-processing delle immagini che possa portare in risalto i lineamenti dei volti possa aumentare la qualità dell'input della rete (ad esempio con tecniche simili all'edge detection) come da immagine d'esempio:

- Una possibilità per migliorare le performance complessive è quella di applicare tecniche/meccanismi di attenzione (attention layers/models).
- Eseguire la fase di training in maniera più controllata e strutturata al fine di monitorare (ed evitare) fenomeni di under e overfitting.