Bortolussi Lorenzo SM3201370 5/5/2025

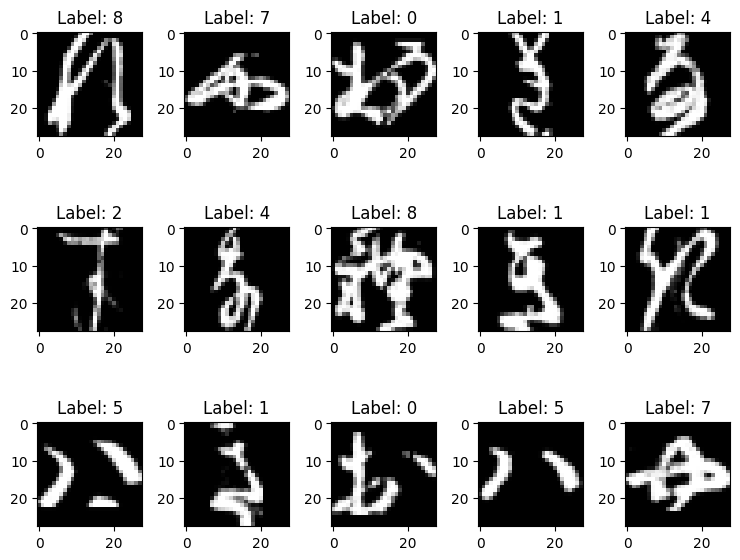
**CHALLENGE 3 – REPORT**

In questa challenge si ha esplorato le capacita delle reti neurali nel riconoscimento di immagini di caratteri appartenenti al dataset KMNIST. Nel mettere a confronto reti neurali convoluzionali (CNN) e completamente connesse lineari (FCNN), si ha considerato più modelli e diversi ottimizzatori, con l’intento di verificare come questi cambiamenti modificassero le prestazioni dei modelli considerati. Si ha poi approfondito un singolo modello, analizzando varie grandezze di interesse come l’evoluzione della loss durante l’allenamento.

Per l’implementazione dei modelli si ha utilizzato la libreria Torch.

**ESPLORAZIONE DEI DATI**

Prima di cominciare, osserviamo i dati con cui si dovrà lavorare:



Si tratta di immagini di dimensioni 28x28, con un singolo canale di colore, mentre le etichette risultano dividersi in 10 classi distinte. Si decide di utilizzare 32 elementi per batch durante l’allenamento.

**DESCRIZIONE DEI MODELLI**

I modelli presi in considerazione sono i seguenti:

* **FCNN\_v1** : rete neurale lineare fully connected con un hiden layer
* **FCNN\_v2** : rete neurale lineare fully connected con tre hidden layers
* **CNN\_v1** : rete neurale con due layer convoluzionali
* **CNN\_v2** : rete neurale con due layer convoluzionali seguiti da pooling
* **CNN\_v3** : rete neurale con due layer convoluzionali preceduti da pooling

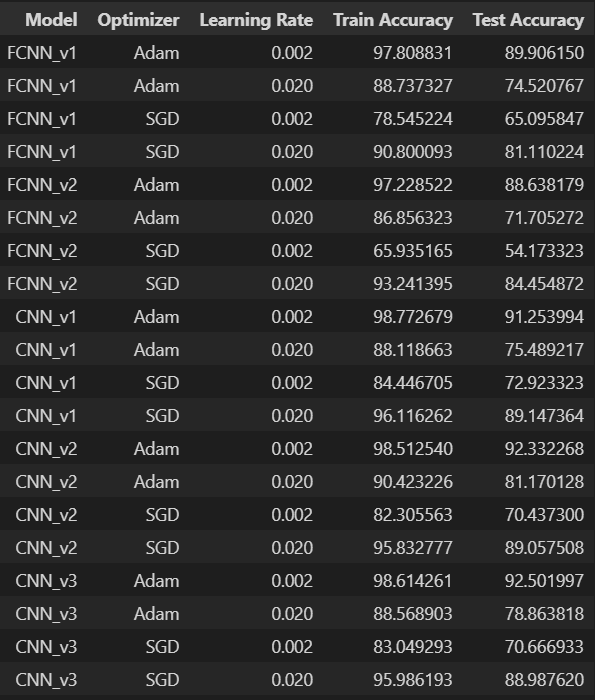
Il modello CNN\_v3 si distingue dal CNN\_v2 in quanto l’operazione di pooling viene effettuata sui dati in input, riducendo da subito la dimensione delle immagini nel tentativo di introdurre immediatamente invarianza rispetto a piccole traslazioni dei caratteri. Senza il vincolo di tre hidden layers massimi, sarebbe interessante osservare anche le peformance di altre combinazioni dei due tipi di layer (pooling e convoluzione).

Vista la bassa riusoluzione dell’immagine, i kernel dei layer convoluzionali e le dimensioni dei pool vengono mantenute ridotte, per evitare di perdere troppi dettagli dell’immagine.

**OSSERVAZIONI SUI RISULTATI**

Per quanto riguarda l’ottimizzatore, per ogni modello si utilizza sia Adam che SGD, utilizzando due learning rate con ordini di grandezza diversi. Il training è stato eseguito su 4 epoche.

Nella tabella seguente sono riportati i risultati dei modelli sopra descritti:

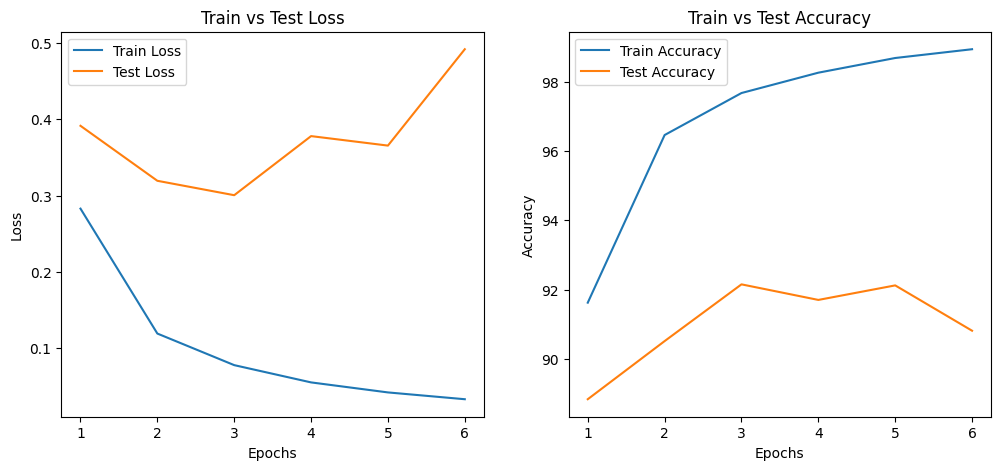


* Dai dati si nota come il learning rate più basso ha favorito l’ottimizzatore Adam, mentre quello più alto ha reso accuracy maggiore tramite SGD. Questo potrebbe essere legato al basso numero di epoche considerate;
* Sorprendentemente quasi ogni combinazione ha reso accuracy piuttosto elevate, anche quelli ritenuti più semplici;
* Il calo di accuracy tra train e test set sembra oscillare attorno al 10% per tutte le combinazioni considerate;
* Non sembrano esserci differenze significative tra i modelli CNN\_v2 e CNN\_v3, al contrario di quanto si pensava in precedenza.

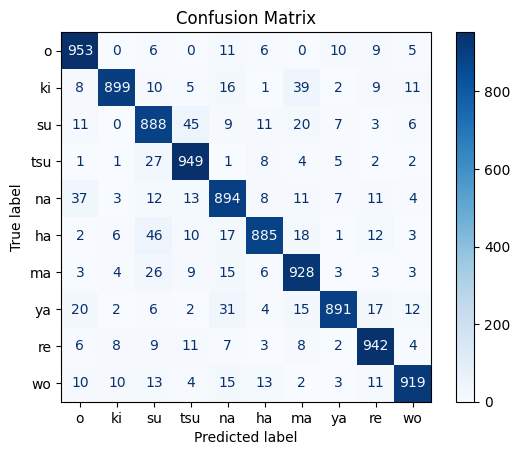
**ANALISI DI LOSS E ACCURACY**

Nel seguito si considera il modello CNN\_v2, ritenuto esemplificativo anche per gli altri modelli convoluzionali osservati in precedenza. In questa sezione il numero di epoche considerate è stato alzato a 6.

Come prima cosa si osserva l’evoluzione di loss e accuracy durante l’allenamento del modello:



Si nota come, superata la terza epoca, mentre la train loss continua a calare, la test loss supera un minimo e comincia a crescere. Un comportamento simile si nota nell’andamento delle accuracy, dove nel test si appiattisce per qualche epoca per poi crescere. Si suppone che questo comportamento segua dall’overfitting che il modello ha verso i dati di train, quindi è ragionevole pensare di fermare l’allenamento del modello alla terza o quarta epoca.

Dalla confusion matrix si nota come la classificazione sembra essere piuttosto buona, visti gli alti numeri sulla diagonale. Si osserva che alcuni gruppi di caratteri vengono classificati erroneamente più di altri, per esempio [ su , tsu ]: questo suggerisce che alcune classi condividono caratteristiche comuni e difficili da distinguere.

Vedendo come le cassi sono equilibrate, si può dire che l’accuracy è una buona misura di partenza per valutare il modello, misura che però non distingue le performance sulle singole classi. Osservando accuracy e F1-score per le singole classi si notano piccole oscillazioni ma tutti i valori rimangono comunque piuttosto elevati:

