CHALLENGE 3

Riccardo Striano

INTRODUZIONE

Lo scopo della challenge è allenare una serie di reti neurali Fully Connected e Convolutional sul dataset KMNIST per poi commentare i risultati.

Tutti i modelli sono stati allenati su 10 epoche con una batch size di 128.

RESULTS

Dalle Test Accuracy nella tabella sottostante si può osservare come i modelli convoluzionali siano superiori a quelli lineari e che le operazioni di pooling sembrano migliorare ancora di più le prestazioni, in particolare funziona bene un kernel 3x3.

All’interno dei due design notiamo che funzionano meglio i modelli con più hidden layer rispetto a quelli meno complessi. Allo stesso modo un maggior numero di neuroni migliora sia l’accuracy che la Delta Acc. Suppongo che questo comportamento derivi dal fenomeno della regolarizzazione intrinseca.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Rete** | **Train Loss** | **Test Loss** | **Train Acc** | **Test Acc** | **∆Acc** |
| **1-Lineare: 1 hidden layer con 64 neuroni e 32 nell’output layer** | 0.1378 | 0.5193 | 95.9647 | 87.4299 | 8.5348 |
| **2-Lineare: 2 hidden layer con 64 e 32 neuroni e 16 nell’output layer** | 0.1371 | 0.5743 | 95.9864 | 85.6871 | 10.2993 |
| **3-Lineare: 2 hidden layer con 256 e 128 neuroni e 64 nell’output layer** | 0.0447 | 0.4637 | 98.7836 | 90.9856 | 7.7980 |
| **4-Convoluzionale: 1 hidden layer con 32 canali e 64 neuroni nell’output layer** | 0.0062 | 0.7274 | 100.0000 | 90.0040 | 9.9960 |
| **5-Convoluzionale: 2 hidden layer con 32 canali e 16 canali. 64 neuroni nell’output layer** | 0.0106 | 0.4369 | 99.8609 | 94.1006 | 5.7603 |
| **6-Convoluzionale: 2 hidden layer con 32 canali e 16 canali con max-pooling con kernel 2x2. 64 neuroni nell’output layer** | 0.0542 | 0.3023 | 98.4937 | 93.7400 | 4.7537 |
| **7-Convoluzionale: 2 hidden layer con 32 canali e 16 canali con max-pooling con kernel 3x3. 64 neuroni nell’output layer** | 0.0422 | 0.2608 | 98.8760 | 94.3510 | 4.5250 |
| **8-Miglior modello(7) con regolarizzazione lam=0.001** | 0.0432 | 0.2277 | 98.8693 | 95.3826 | 3.4867 |
| **9-Miglior modello (8) con drop-out 0.1** | 0.4080 | 0.2722 | 99.1230 | 95.0421 | 4.0809 |
| **10-Miglior modello (8) allenato per 30 epoche** | 0.2211 | 0.2898 | 100.0000 | 95.9435 | 4.0565 |

Notiamo anche che la differenza tra la Train e la Test Accuracy, riassunta nella colonna ∆Acc, è molto grande per i modelli lineari e i modelli convoluzionali più semplici; ciò suggerisce un overfitting. Nonostante la cosa migliori con l’aggiunta dell’operazione di max-pooling si è visto che, aggiungendo un fattore di regolarizzazione o il drop-out, abbiamo degli ulteriori miglioramenti.

Si noti come il modello migliore sia decisamente il numero 8 caratterizzato sia dalla Test Accuracy più alta che dalla ∆Acc più bassa, tipiche di una buona generalizzazione

*Nota: uso il 5% di differenza come discrimine tra overfitting da smussare o meno.*

ANALISI DEI GRAFI

Andiamo ora a analizzare i grafici di Loss e Accuracy comparandoli tra Train e Test. Quello che notiamo è che, come descritto prima, i modelli più semplici sono tutti simili al grafico 1, dove la distanza tra le curve è relativamente grande suggerendo un overfitting.

I modelli regolarizzati invece sono simili al grafico 2, con una distanza più piccola tra le curve che suggerisce una miglior generalizzazione del modello.

*Immagine che contiene diagramma, Diagramma, linea, testo

Il contenuto generato dall'IA potrebbe non essere corretto.*

*1-Modelli con overfitting. In particolare questo è il modello lineare con 2 hidden layer da 256 e 128 neuroni e 64 nell’output layer*

*Immagine che contiene diagramma, testo, Diagramma, linea

Il contenuto generato dall'IA potrebbe non essere corretto.*

*2-Modelli senza overfitting. In particolare questo è il modello convoluzionale con 2 hidden layer da 32 e 16 canali che usa max-pooling 3x3 e 64 neuroni di output con regolarizzazione lambda = 0.001*