Analisi di modelli per la classificazione di immagini su dataset Fashion MNIST

Tommaso Martinelli

***Abstract*—Questo studio presenta una dettagliata comparazione tra una rete neurale convoluzionale (CNN) personalizzata e un modello preaddestrato, nel contesto della classificazione delle immagini, utilizzando il dataset Fashion MNIST. Attraverso un'analisi approfondita, sono stati esplorati diversi approcci di progettazione e addestramento dei modelli di deep learning, includendo la definizione di architetture, l'ottimizzazione dei parametri e l'utilizzo di tecniche di regolarizzazione. Il dataset Fashion MNIST, ampiamente studiato nella letteratura scientifica, è stato sottoposto a una rigorosa analisi esplorativa per comprendere le sue caratteristiche e distribuzioni. La CNN personalizzata è stata sviluppata specificamente per il task di classificazione, mentre il modello preaddestrato, ResNet18, è stato adattato al problema. I risultati indicano una buona performance complessiva, con la CNN personalizzata che mostra risultati superiori al modello preaddestrato. Questo studio fornisce importanti indicazioni sulla progettazione e sull'addestramento di modelli di deep learning per la classificazione delle immagini, evidenziando l'importanza della personalizzazione delle architetture per adattarsi ai contesti specifici dei dataset e della ricerca continua di nuove strategie per migliorare le prestazioni dei modelli.**

I. Introduction

Classificare un'immagine in base al suo contenuto è un compito che l'essere umano svolge con relativa facilità, grazie agli automatismi del cervello che consentono di riconoscere rapidamente oggetti familiari. Tuttavia, delegare tale compito a un calcolatore costituisce una sfida complessa. La classificazione di immagini è da tempo uno dei principali ambiti di studio della computer vision e dell'intelligenza artificiale, con l'obiettivo di insegnare ai sistemi informatici a riconoscere autonomamente il contenuto di immagini digitali e assegnare loro etichette o classi predefinite. Nel corso degli anni, questo campo ha registrato significativi progressi, con lo sviluppo di vari modelli capaci di risolvere task di classificazione sempre più complessi. Tali modelli, grazie all'impiego di approcci innovativi e all'aumento della potenza computazionale, sono diventati sempre più performanti e ampiamente utilizzati in molteplici applicazioni reali. Nonostante i successi ottenuti, la classificazione di immagini presenta ancora numerose sfide che richiedono l'attenzione e l'impegno continuo dei ricercatori. Tra queste sfide vi è la necessità di garantire maggiore robustezza ai modelli di classificazione, specialmente in presenza di dati di input rumorosi o condizioni di illuminazione variabili. Inoltre, è importante migliorare l'interpretabilità dei modelli complessi, al fine di comprendere meglio le decisioni prese dall'algoritmo e favorire la fiducia degli utenti. Parallelamente, è essenziale ottimizzare le performance attuali attraverso lo sviluppo di nuove metodologie. Ciò include l'individuazione di modelli con una complessità computazionale inferiore, che consente una maggiore scalabilità e una migliore gestione delle risorse computazionali.

Questo studio si propone di esplorare alcuni degli approcci principali alla image classification, analizzando in dettaglio diversi modelli e tecniche utilizzati nell'ambito della classificazione di immagini. Attualmente si può affrontare un problema del genere ricorrendo al deep learning, una branca dell'intelligenza artificiale che si basa sull'uso di reti neurali artificiali profonde, composte da molti strati di neuroni interconnessi. Queste reti sono in grado di apprendere automaticamente rappresentazioni dei dati, consentendo loro di riconoscere pattern complessi e svolgere compiti di classificazione con elevate prestazioni.

Nel contesto della classificazione di immagini, le reti neurali convoluzionali (CNN) sono diventate uno dei principali strumenti utilizzati. Le CNN sono specificamente progettate per l'elaborazione di immagini e sfruttano il concetto di convoluzione per estrarre automaticamente features significative dai dati. Queste features vengono quindi passate attraverso uno o più strati completamente connessi per la classificazione finale dell'immagine. Le CNN risultano dunque un'ottima scelta per affrontare problematiche di questo tipo. Esistono diverse opzioni quando si tratta di utilizzare le CNN per la classificazione di immagini. Da un lato, è possibile progettare architetture personalizzate, adattate specificamente al problema in esame, sperimentando con diverse configurazioni di strati convoluzionali, di pooling e completamente connessi. Questo approccio consente una maggiore flessibilità e la possibilità di ottimizzare l'architettura per specifiche esigenze. D'altro canto, esistono anche architetture note e pre-addestrate, come ad esempio VGG [1], ResNet [2] e molti altri, che sono state sviluppate e ottimizzate da ricercatori e professionisti del settore. Utilizzare queste architetture pre-addestrate può offrire diversi vantaggi, tra cui una maggiore facilità di implementazione, prestazioni già validate su benchmark standard e la possibilità di beneficiare del transfer learning per problemi di classificazione di immagini simili. La scelta tra la progettazione di architetture personalizzate e l'utilizzo di architetture pre-addestrate dipende dalle specifiche del problema, dalle risorse disponibili e dagli obiettivi del progetto. Entrambi gli approcci possono essere validi e meritano di essere esplorati e valutati in base al contesto specifico. Il presente studio si pone l'obiettivo di confrontare questi due approcci. Per fare ciò è stato utilizzato Fashion MNIST [3], un noto dataset, molto utilizzato negli studi di computer vision. La sua importanza risiede nella sua utilità come benchmark standard per valutare e confrontare l'efficacia degli algoritmi di classificazione di immagini. Le immagini rappresentano capi di abbigliamento, rendendo il dataset più realistico rispetto a MNIST, consen-

tendo agli sviluppatori di testare e confrontare le prestazioni dei propri algoritmi in un contesto più simile alle applicazioni del mondo reale. Inoltre, la dimensione relativamente piccola delle immagini e il numero limitato di classi rendono Fashion MNIST un dataset ideale per sperimentare differenti approcci.

II. Related Work

Dopo aver delineato il contesto generale della classificazione di immagini e presentato l'importanza delle reti neurali convoluzionali (CNN), verrà posta l'attenzione sui lavori correlati in questo campo. Verranno esaminate le ricerche precedenti che hanno utilizzato il dataset Fashion MNIST come benchmark per valutare e confrontare diverse metodologie di classificazione di immagini.

Il dataset su cui si concentra lo studio presenta immagini di capi di abbigliamento appartenenti a 10 categorie diverse, tra cui magliette, pantaloni, scarpe, vestiti e borse. Ogni immagine è in scala di grigi e ha una risoluzione di 28x28 pixel. Fashion MNIST rappresenta uno dei dataset più noti nel mondo della computer vision e nel corso degli anni sono stati presentati numerosi modelli per risolvere il suo task di classificazione. Il primo è stato presentato nel paper ufficiale del dataset nel 2017, che tramite l'uso di una CNN relativamente semplice, ha ottenuto una Accuracy del 90.3%. Successivamente, sono stati proposti modelli CNN più avanzati che hanno migliorato questo risultato. Nello stesso anno infatti, la pubblicazione di Bhatnagar et al. [4] ha proposto una soluzione migliore. Tramite una semplice CNN con 2 layer convoluzionali, 2 di pooling, 2 fully-connected e l'uso di batch normalization[5], skip connection e dropout[6], si è ottenuto un incremento delle prestazioni, raggiungendo una Accuracy del 92.54%. Il dataset è stato utilizzato anche per condurre analisi comparative tra le architetture di CNN più diffuse. Un esempio è la comparazione fatta da C. Duan, P. Yin, Y. Zhi, e X. Li [7], che confronta una versione custom di VGG con LeNet [8], AlexNet [9], NiN [10], DenseNet [11] e VGG11, dimostrando come nessuno dei modelli classici riuscisse a migliorare i risultati. Un upgrade molto importante è stato quello ottenuto da M. Kayed, A. Anter e H. Mohamed nel 2020, tramite l'uso di una rete LeNet-5 [12]. L'architettura utilizzata è molto semplice, alternando layer convoluzionali con layer di average pooling e, infine, un unico fully-connected. Questa nuova rete è riuscita ad ottenere una Accuracy del 98.8%.

Gli studi descritti sono tra i più noti e importanti nel campo della classificazione di immagini di vestiti, ma esistono molti altri Paper che propongono altre valide soluzioni. Studi successivi infatti hanno ulteriormente incrementato le performance, tramite l'uso di tecniche diverse, che non sono oggetto dello studio in questione.

III. Proposed Approach

In questa sezione, verranno presentati gli approcci trattati in questo studio per affrontare il problema della classificazione delle immagini utilizzando il dataset Fashion MNIST. Prima di entrare nei dettagli dell'architettura del modello e delle tecniche di addestramento, è importante comprendere la struttura e le caratteristiche fondamentali del dataset utilizzato. Per fare ciò, è stato condotto un processo di esplorazione del dataset mediante l'analisi dei dati e la valutazione delle loro distribuzioni e caratteristiche. Questo processo è stato documentato in un notebook dedicato all'esplorazione del dataset, che ha permesso di acquisire una comprensione approfondita della struttura del dataset. Vengono già forniti due inisemi distinti per il training e il test, rispettivamente composti da 60000 e da 10000 campioni. Ogni campione nel dataset Fashion MNIST è rappresentato da un'immagine in scala di grigi di dimensioni 28x28 pixel e viene associato a una delle 10 categorie di capi di abbigliamento. Durante l'esplorazione del dataset, sono state analizzate anche le distribuzioni delle classi per verificare se vi fossero eventuali sbilanciamenti tra le diverse categorie di capi di abbigliamento. È importante sottolineare che le classi sono perfettamente bilanciate, garantendo un equilibrio nella rappresentazione dei diversi tipi di capi di abbigliamento nel training dei modelli di classificazione.

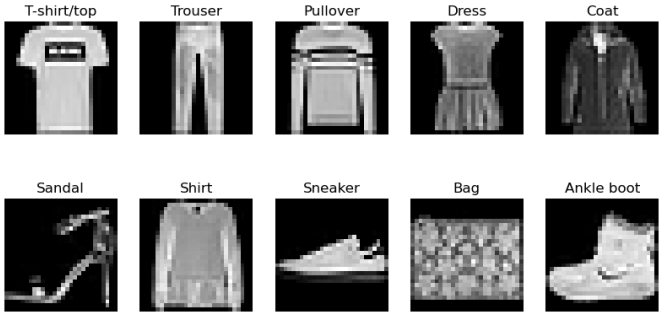

Fig. 1: Esempio di immagini del dataset

Dopo aver esplorato le caratteristiche fondamentali del dataset Fashion MNIST e aver verificato la sua struttura e distribuzione delle classi, si è proceduto a definire due possibili approcci per affrontare il problema della classificazione delle immagini.

A. Custom CNN

La prima soluzione consiste nell'utilizzo di una semplice CNN, sviluppata appositamente per la risoluzione di questo task. La rete è composta da alcuni strati convoluzionali per l'estrazione delle features, seguiti da strati completamente connessi per la classificazione.

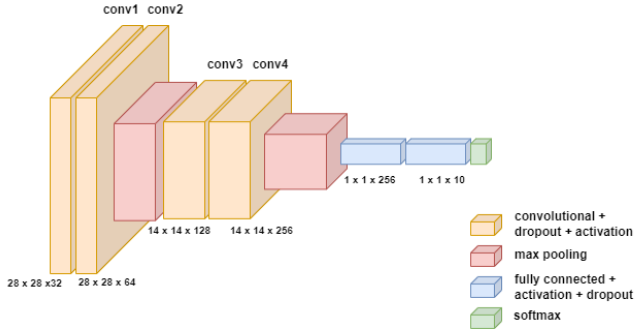

Fig. 2: Custom CNN

L'architettura della rete, mostrata in figura 2, ha una struttura:

INPUT → [[*CONV* → *DROPOUT* → *ATTIVAZIONE*]*2 → *POOLING*]*2 → *FC* → *ATTIVAZIONE* → *DROPOUT* → *FC*

Che può essere vista più in dettaglio andando ad analizzare ogni singolo layer:

- In input riceve delle immagini 28x28, in scala di grigi. Le immagini passeranno per il primo layer convoluzionale, composto da 64 filtri 3x3. Il layer applica un padding 1 e stride 1.
- Prima del secondo layer convoluzionale viene applicato il dropout. Utile per prevenire l'overfitting in fase di addestramento e per migliorare la capacità di generalizzazione del modello. In questa architettura è ripetuto dopo ogni layer convoluzionale.
- Viene applicata una funzione di attivazione non lineare, che introduce non linearità nella rete permettendo al modello di apprendere relazioni complesse nei dati di input.
- Il secondo layer convoluzionale è composto da 128 filtri 3x3. Anche in questo caso, il layer applica un padding 1 e uno stride 1.
- Dopo il secondo layer convoluzionale, viene applicato il max pooling con una dimensione della finestra di pooling pari a 2x2 e uno stride di 2. Ciò significa che l'immagine viene suddivisa in regioni di 2x2 pixel, e per ciascuna di queste regioni viene selezionato il valore massimo. Questo layer dunque da come risultato una riduzione della dimensionalità delle feature map e una maggiore robustezza del modello nei confronti delle variazioni spaziali nelle immagini di input.
- Le feature map risultanti passano attraverso due nuovi strati convoluzionali, rispettivamente con 128 e 256 filtri. Come gli strati precedenti utilizzano padding 1 e stride 1, viene utilizzato il dropout e le funzioni di attivazione. Dopodichè si applica nuovamente il layer di max pooling.
- Infine, le feature map ridotte vengono appiattite e passate attraverso due strati fully connected. Il primo strato fully connected ha 256 unità, seguito da una funzione di attivazione non lineare e l'applicazione del dropout per la regolarizzazione. Il secondo strato fully connected ha 10 unità di output, corrispondenti al numero di classi nel problema di classificazione.

L'architettura appena descritta è stata ottenuta in seguito a una serie di prove sul dataset trattato. Questa rete ha mostrato risultati promettenti ed è stata poi oggetto di studio per ottimizzarne le performance.

B. Modello preaddestrato ResNet18

Il secondo approccio al problema di classificazione consiste nell'utilizzo di un modello preaddestrato. Questo tipo di scelta permette di utilizzare modelli più profondi e complessi senza la necessità di doverli addestrare da 0, ma utilizzando il loro addestramento precedente. Il modello scelto è ResNet18, una delle più note varianti della famiglia delle ResNet. Questo tipo di reti sono state addestrate su dataset ImageNet[13] e sono generalmente molto efficaci per problemi di classificazione di immagini. La caratteristica principale di questo tipo di architetture sono le connessioni residue, che consentono il passaggio delle informazioni non elaborate attraverso gli strati della rete. La struttura di ResNet è costituita da blocchi residui, ognuno composto da più strati convoluzionali seguiti da un'operazione di somma che aggiunge l'input originale al risultato della trasformazione convoluzionale. Ciò rende l'addestramento più efficiente e facilita la propagazione del gradiente. ResNet18 è una delle varianti più leggere della famiglia ResNet, composta da un totale di 18 strati, tra convoluzionali e fully-connected.

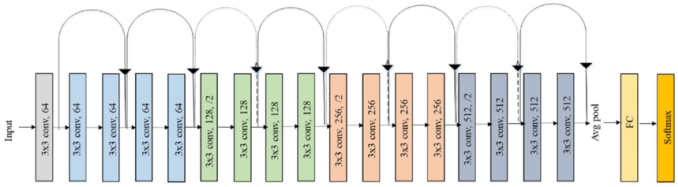

Fig. 3: ResNet18

Per adattare ResNet18 al problema di classificazione delle immagini del dataset Fashion MNIST, sono state apportate alcune modifiche alla struttura originale del modello. Poiché il dataset Fashion MNIST contiene immagini in scala di grigi anziché a colori come in ImageNet, è stata modificata la dimensione del canale di input del primo strato convoluzionale per accettare immagini in scala di grigi anziché a tre canali RGB. Inoltre, poiché il numero di classi nel dataset Fashion MNIST è 10 anziché 1000 come in ImageNet, è stata modificata la dimensione dell'ultimo strato completamente connesso per produrre un output con 10 neuroni anziché 1000, per adattarsi al numero di classi nel nostro dataset di destinazione. Queste modifiche hanno consentito di utilizzare ResNet18 come estrattore di feature preaddestrato per il dataset Fashion MNIST, sfruttando le sue capacità di apprendimento su immagini più generiche e complesse fornite da ImageNet, ma adattandolo al contesto specifico.

IV. Experiments

In questo capitolo verranno descritti gli esperimenti condotti per valutare le performance dei due approcci, sul dataset Fashion MNIST. In questa fase sono stati effettuati anche una serie di test per ottimizzare i modelli tramite varie scelte progettuali. Ciò è stato possibile tramite l'utilizzo di script personalizzati per addestramento, validazione e testing dei modelli, che comprendono al loro interno funzioni per plottare le loss, visualizzare risultati e metriche. La definizione di queste funzioni è stata necessaria per avere una metodologia unica per addestrare e validare i differenti parametri e configurazioni.

A. Ambiente di sviluppo

L'addestramento, validazione e test dei modelli sono stati effettuati tramite l'utilizzo di script Python e Jupyter Notebook, utilizzando la libreria PyTorch. Il tutto è stato eseguito su una macchina con a disposizione una GPU NVIDIA GeForce GTX 1050 con 4 GB di memoria.

B. Dataset

Il dataset, fornito da Pytorch, è già suddiviso in train set e test set, ma è stato necessario suddividere ulteriormente il set di addestramento per avere un set di validazione, a cui è stata assegnata una dimensione pari al 20% del totale. Tutti i dati inoltre hanno subito una trasformazione, per trasformarli in tensori e per normalizzarli.

Al dataset di train inoltre, sono state applicate altre trasformazioni, tramite tecniche di data augmentation per garantire una maggiore variabilità tra i dati. In particolare sono state ruotate casualmente delle immagini in orizzontale e sono state inclinate di 20 gradi in senso orario o antiorario altre immagini. Si è optato dunque solo per semplici operazioni, necessarie però per abbassare il rischio di overfitting.

C. Addestramento CNN custom

La rete personalizzata è stata definita e in seguito addestrata in seguito ad una serie di step, in cui diverse configurazioni di CNN sono state addestrate sul train set e testate sul validation, usando l'accuracy come metrica di riferimento. La rete descritta nel capitolo precedente è stata oggetto dunque di attente analisi. La sua architettura è stata ottenuta seguendo i principi generali della composizione di CNN, ma ha subito numerosi raffinamenti; infatti sono state testate anche architetture più profonde, sono stati modificati i layer convoluzionali, di pooling e fully-connected. L'architettura proposta è stata selezionata in quanto apparentemente è la più promettente dal punto di vista dei risultati.

I filtri utilizzati nei layer convoluzionali sono tutti 3x3 perchè più efficienti e consentono di andare in profondità senza innalzare eccessivamente il numero di parametri. Il parametro batch size può in generale influenzare diversi aspetti delle prestazioni della rete neurale, perciò sono stati fatti addestramenti con dimensioni diverse e si è potuto vedere, dai risultati sul validation, che la dimensione ideale è di 128. Durante gli esperimenti, la rete è stata addestrata utilizzando l'ottimizzatore Adam[14] con diverse learning rate. Sono stati testati valori compresi tra 0.01 e 0.0001, al fine di determinare quale valore fosse ottimale per il modello e il dataset. Questa gamma di valori ha consentito di esplorare l'effetto del learning rate sul processo di addestramento e sulle prestazioni finali del modello. I risultati ottenuti hanno mostrato che un learning rate di 0.001 ha prodotto le migliori prestazioni sul set di validazione, garantendo una convergenza stabile e una riduzione significativa della funzione di perdita durante l'addestramento.

Durante l'addestramento della CNN personalizzata, sono state esplorate diverse tecniche di regolarizzazione al fine di migliorare le prestazioni del modello e prevenire l'overfitting. Le tecniche provate includono dropout, batch normalization e regolarizzazione L2[15]. La scelta della regolarizzazione è stata effettuata facendo una serie di test sul set di validazione, osservando performance migliori in caso di utilizzo del dropout. In particolare si è deciso di utilizzare due differenti soglie di dropout, una per i layer convoluzionali più bassa (0.1) e una più elevata nei fully-connected (0.5). La rete ottenuta in seguito agli esperimenti descritti è stata addestrata sul set di addestramento, testandola sul set di validazione. Si è fatto uso di una funzione di attivazione ReLu[16] e dei valori dei parametri, ottenuti tramite le precedenti validazioni. L'addestramento è stato svolto tramite la funzione personalizzata, precedentemente citata, che implementa un meccanismo di early stopping per la prevenzione dell'overfitting. I parametri scelti per l'addestramento sono 100 epoche e una pazienza di 5. Nei grafici sono mostrati gli andamenti delle loss al passare delle epoche e i valori di accuracy.

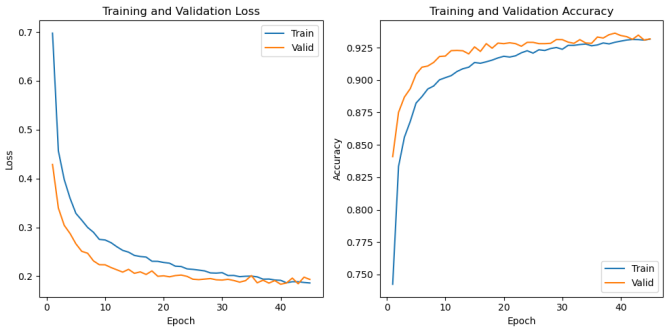

Fig. 4: Loss e accuracy su train e validation della CNN custom

Il modello ottiene una accuracy sul validation set del 93.26%, inoltre viene mostrata la confusion matrix per comprendere al meglio quali errori fa il modello.

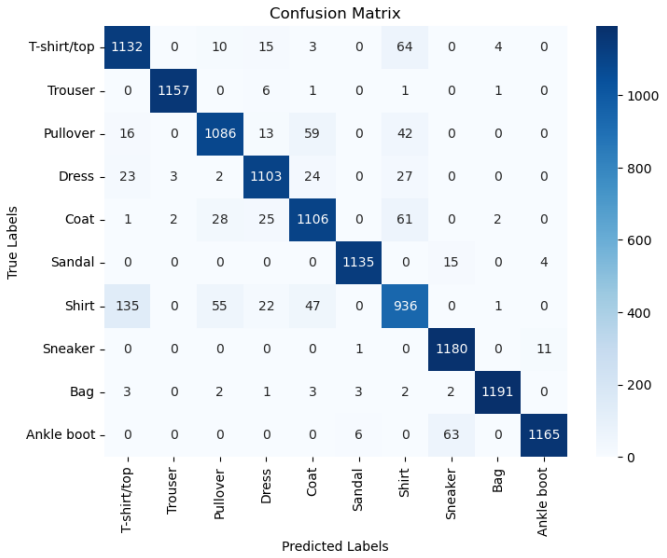

Fig. 5: Confusion Matrix su validation set della CNN custom

Come ultimo step per l'ottimizzazione della rete custom, si è deciso di valutare l'impatto di una funzione di attivazione differente rispetto alla ReLU, che è la funzione più comunemente utilizzata. Sebbene la ReLU abbia dimostrato di essere efficace in molte applicazioni, può comportare alcuni problemi, tra cui il fenomeno noto come "dying ReLU"[17], in cui alcuni neuroni smettono di aggiornare i loro pesi durante l'addestramento, causando una perdita di informazioni importanti. Per affrontare questa questione, si è deciso di testare l'utilizzo della funzione di attivazione Leaky ReLU, che introduce una piccola pendenza per i valori negativi, consentendo ai

neuroni di continuare a trasmettere informazioni anche quando hanno un output negativo. Questo approccio potrebbe aiutare a mitigare il problema e migliorare le prestazioni complessive della rete. Anche questa versione della rete è stata quindi addestrata e testata sul set di validazione.

Fig. 6: Loss e accuracy su train e validation della CNN custom con Leaky ReLu

Il modello ha ottenuto una accuracy del 93.34% ed è inoltre mostrata, anche in questo caso, la confusion matrix. Le performance risultano essere leggermente migliori rispetto alla versione con ReLu.

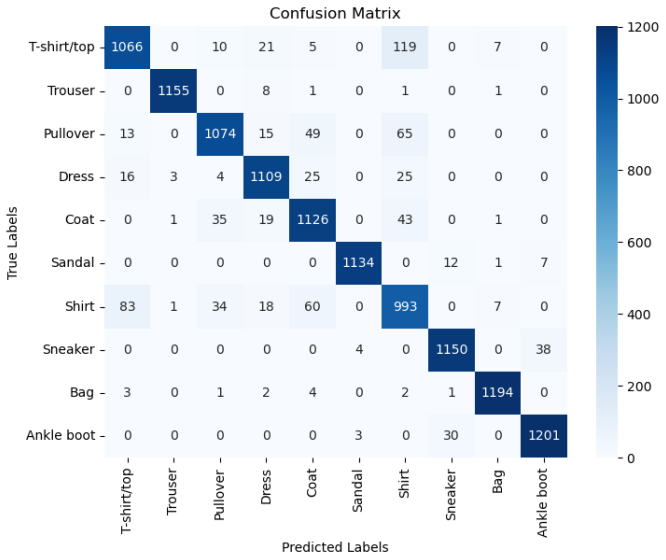

Fig. 7: Confusion Matrix su validation set della CNN custom con Leaky ReLu

D. Addestramento modello preaddestrato

La seconda fase sperimentale ha riguardato l'ottimizzazione del modello preaddestrato ResNet18. Come precedentemente descritto, sono state apportate modifiche per adattare il modello al problema in questione. La scelta del modello ResNet18 è frutto di una serie di test, infatti sono state valutate molteplici versioni di Resnet, come la ResNet34, ResNet50 e ResNet101, che hanno ottenuto performance peggiori sul validation, nonostante la struttura più profonda. Inoltre come per la rete Custom sono stati testati differenti valori per i parametri di batch-size e learning-rate. Per questo tipo di modello inoltre, è stato necessario analizzare la possibilità di usare i parametri del modello preaddestrato oppure effetuare il fine tuning. questa tecnica può essere efficace per adattare un modello preaddestrato a un nuovo problema, consentendo al modello di apprendere dalle caratteristiche specifiche del nuovo dataset, ma il costo computazionale è elevato. Considerando le risorse a disposizione si è deciso di confrontare solo due approcci: l'utilizzo dei parametri del modello preaddestrato senza apportare modifiche e il fine-tuning solo degli ultimi strati del modello. La prima strategia ha performato meglio, probabilmente per via della differenza tra il dataset usato per il fine tuning e quello usato per l'addestramento inziale. Quindi riaddestrare gli ultimi layer non è risultata una soluzione abbastanza efficiente al problema.

L'addestramento e i test sul set di validazione sono stati svolti con le funzioni personalizzate. I risultati del modello senza fine-tuning sono mostrati nei seguenti grafici.

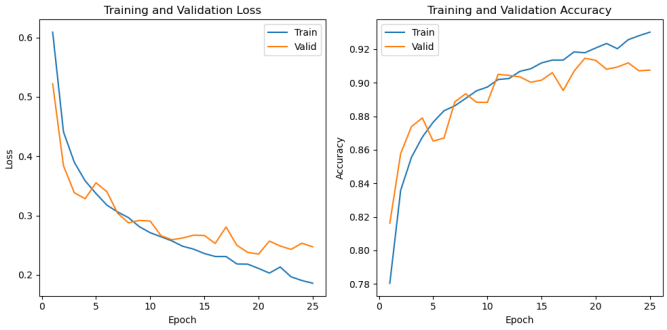

Fig. 8: Loss e accuracy su train e validation del modello preaddestrato

Il modello preaddestrato riesce ad ottenere una accuracy del 90.99% sul validation set.

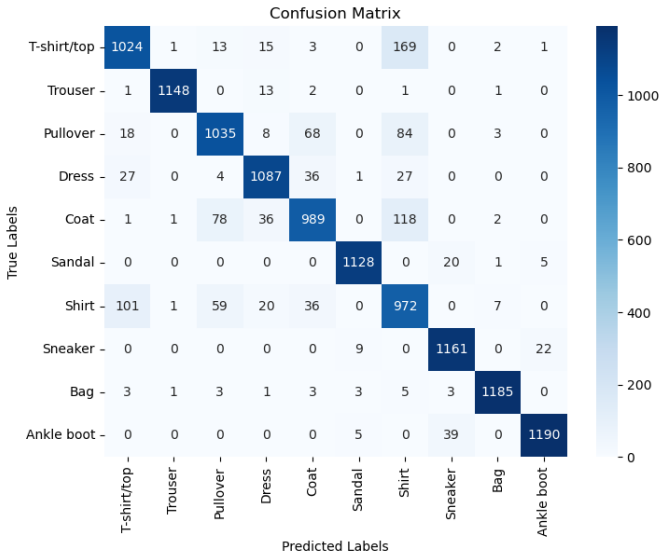

Fig. 9: Confusion Matrix su validation set del modello preaddestrato

E. Test e confronto dei modelli

Come ultimo step della fase sperimentale, sono stati testati i modelli che avevano performato meglio in fase di validazione. Le due reti sono state quindi testate utilizzando il test set, che non era mai stato utilizzato fin'ora. Il confornto tra le prestazioni della CNN custom con Leaky ReLu e del modello preaddestrato senza fine-tuning è mostrato nella tabella sottostante.

Modello	Accuracy (%)
CNN Custom	92.88
ResNet18	90.91

TABLE I: Confronto delle prestazioni tra CNN Custom e ResNet18 sul test set

Sono mostrate anche le confusion matrix dei due modelli sul set di test.

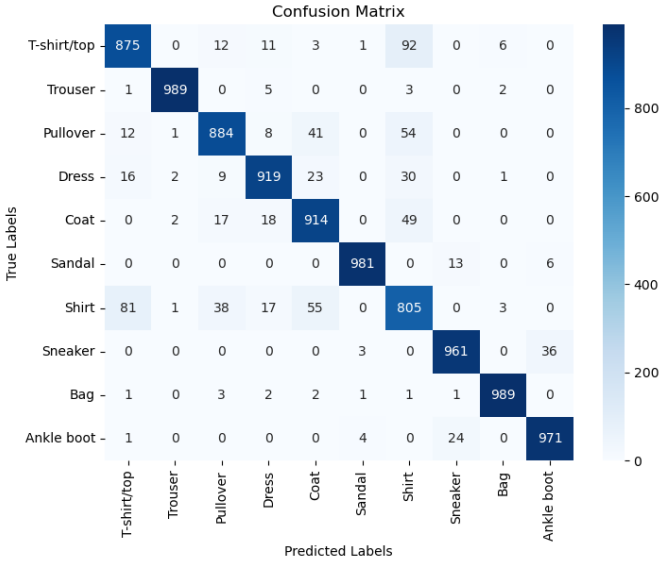

Fig. 10: Confusion Matrix CNN custom

Le metriche mostrano che il modello custom si adatta meglio al problema e ottiene degli score migliori, ma entrambe possono considerarsi soluzioni buone.

V. Conclusion

Lo studio condotto ha fornito una dettagliata comparazione tra una CNN custom e un modello preaddestrato nel contesto della classificazione delle immagini. Nonostante le limitazioni hardware, è stato possibile condurre un'analisi approfondita che ha attraversato diverse fasi della progettazione e dell'addestramento dei modelli di deep learning. Il dataset utilizzato, il Fashion MNIST, è ampiamente documentato nella letteratura scientifica, consentendo un confronto diretto delle prestazioni ottenute con i risultati riportati in numerosi studi precedenti. I risultati ottenuti mostrano una buona performance complessiva, anche se ci sono spazi per ulteriori miglioramenti mediante l'adozione di strategie alternative o l'esplorazione di architetture differenti. È interessante notare che la CNN custom ha mostrato performance superiori al

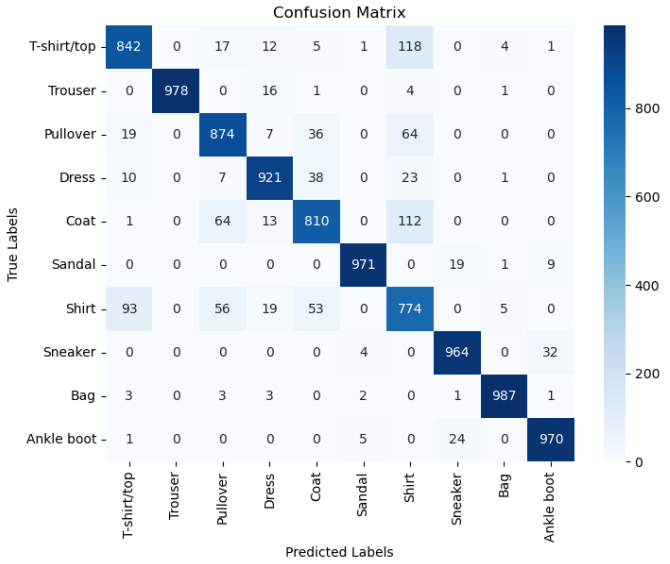

Fig. 11: Confusion Matrix ResNet18

modello preaddestrato, suggerendo che la progettazione di un'architettura specifica per il task può portare a risultati più efficaci. Per quanto riguarda il modello preaddestrato, sarebbe utile considerare l'opportunità di effettuare un fine tuning completo o di esplorare l'uso di altre architetture ben note che potrebbero adattarsi meglio al problema. Questo potrebbe portare a miglioramenti significativi delle prestazioni e alla creazione di modelli ancora più efficaci per la classificazione delle immagini.

In conclusione, lo studio fornisce importanti indicazioni per il futuro sviluppo di modelli di deep learning per la classificazione delle immagini, evidenziando l'importanza della progettazione mirata delle architetture e della ricerca continua di nuove strategie e tecniche per migliorare le prestazioni dei modelli.

References

[1] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[2] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[3] R. V. Han Xiao, Kashif Rasul, "Fashion-mnist: A novel image dataset for benchmarking machine learning algorithms," *arXiv preprint arXiv:1708.07747*, 2017. [Online]. Available: https://arxiv.org/abs/1708.07747

[4] S. Bhatnagar, D. Ghosal, and M. H. Kolekar, "Classification of fashion article images using convolutional neural networks," in *2017 Fourth International Conference on Image Information Processing (ICIIP)*, 2017, pp. 1–6.

[5] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*. pmlr, 2015, pp. 448–456.

[6] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 56, pp. 1929–1958, 2014. [Online]. Available: http://jmlr.org/papers/v15/srivastava14a.html

[7] C. Duan, P. Yin, Y. Zhi, and X. Li, "Image classification of fashion-mnist data set based on vgg network," in *Proceedings of 2019 2nd International Conference on Information Science and Electronic Technology (ISET 2019). International Informatization and Engineering As-*

sociations: Computer Science and Electronic Technology International Society, vol. 19, 2019.

[8] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[9] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, 2012.

[10] M. Lin, Q. Chen, and S. Yan, "Network in network," *arXiv preprint arXiv:1312.4400*, 2013.

[11] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[12] M. Kayed, A. Anter, and H. Mohamed, "Classification of garments from fashion mnist dataset using cnn lenet-5 architecture," in *2020 International Conference on Innovative Trends in Communication and Computer Engineering (ITCE)*, 2020, pp. 238–243.

[13] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[14] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[15] A. N. Tikhonov, "Solution of incorrectly formulated problems and the regularization method." *Sov Dok*, vol. 4, pp. 1035–1038, 1963.

[16] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 807–814.

[17] L. Lu, Y. Shin, Y. Su, and G. E. Karniadakis, "Dying relu and initialization: Theory and numerical examples," *arXiv preprint arXiv:1903.06733*, 2019.