UNIVERSITÀ DEGLI STUDI DI SALERNO

Dipartimento di Informatica

Corso di Laurea Magistrale in Informatica

TESI DI LAUREA

Ecological Validity di Modelli di Machine Learning per Classificazione di Immagini: Uno Studio Empirico

RELATORE

Prof. Fabio Palomba

Dott. Valeria Pontillo

Università degli Studi di Salerno

CANDIDATO

Antonio Cimino

Matricola: 0522500922

Anno Accademico 2021-2022

Questa tesi è stata realizzata nel

"Non c'è nulla di nobile nell'essere superiore a qualcun altro.
La vera nobiltà è essere superiore a chi eravamo ieri."
Ernest Hemingway

Abstract

Oggi è molto diffuso l'utilizzo di sistemi che permettono di analizzare le immagini per trarne informazioni in tantissimi campi. Il presente lavoro di ricerca ha lo scopo di trovare una soluzione per riconoscere i capi di abbigliamento all'interno delle immagini. In letteratura esistono già diversi software che riescono a raggiungere tale scopo e sfruttando queste conoscenze si cercherà di costruire una rete per ottenere dei risultati migliori e successivamente testarne l'affidabilità anche al di fuori di un contesto sperimentale, ovvero capirne l' ecological validity. Quindi, il primo passo consisterà nel strutturare una rete neurale per generare dei modelli che possano soddisfare tale obiettivo che poi successivamente verranno messi alla prova in più modi. I risultati dell'analisi hanno dimostrato che i modelli generati sono soddisfacenti a metà, perché riescono a riconoscere bene nel momento in cui gli elementi da classificare sono ben visibili, meno se le immagini hanno rumore, cosa che nella realtà può accadere, e quindi non utilizzabili al di fuori dello studio.

Indice

Elenco delle figure

Elenco delle tabelle

CAPITOLO 1

Introduzione

1.1 Contesto applicativo

Il progetto svolto è incentrato sul problema del riconoscimento degli elementi nelle immagini, più specificatamente di capi di abbagliamento. Il riconoscimento avviene attraverso la computer vision, una branca dell'intelligenza artificiale (AI) che studia algoritmi e tecniche che consentono ai computer di replicare i processi e le funzioni dell'apparato visivo umano per rilevare e interpretare informazioni importanti in un'immagine digitale, un video o altri input visivi. Per poter funzionare e riuscire a interpretare i contenuti presenti all'interno di un immagine, i sistemi di computer vision devono essere prima "addestrati" attraverso un processo di machine learning utilizzando una grande quantità di immagini catalogate, le quali saranno la base del dataset che permetterà all'algoritmo di riconoscere quelle successive in modo intelligente. La specializzazione del sistema a riconoscere capi d'abbigliamento nasce dai multipli utilizzi che può avere. Parliamo di elementi che vengono utilizzati tutti i giorni e oltre che ai semplici utilizzi commerciali, come ad esempio impedire repliche tra i vari marchi, o suggerire alle persone cose più vicine ai propri gusti, può essere utilizzato anche in altri contesti, come ad esempio nell'ambito ecologico, per suddividere al meglio i prodotti, e cosi via.

1.2 Motivazioni e Obiettivi

L'obiettivo di tesi è quello di costruire una rete che sia in grado di generare un modello per riconoscere i capi di abbigliamento all'interno delle immagini, e successivamente testarne l'efficienza. Prima di tutto verranno effettuate delle ricerche, per capire cosa si trova attualmente in letteratura. Sono tanti gli studi attualmente presenti e molti, come vedremo, hanno ottenuto risultati eccellenti fino ad arrivare addirittura intorno al 99% di accuratezza con LeNet-5. Sulla base di questi lavori verrà costruita una rete per generare dei modelli che possano migliorare o avvicinare quei risultati. Dopo aver generato i modelli si potranno effettuare dei test per capirne effettivamente le potenzialità. Al termine del progetto si vuol portare la ricerca al di fuori del contesto sperimentale, quindi bisognerà capire anche se ciò che è stato prodotto possa funzionare in situazioni reali. Quello di cui stiamo parlando è definibile anche come ecological validity [10], cioè, quanto i risultati sperimentali siano generalizzabili nel mondo reale, come situazioni o contesti tipici della vita quotidiana. La tesi intende verificare l'ipotesi per cui certi tipi di dataset sono molto facili da classificare per le tecniche di classificazione e danno poi la parvenza di aver realizzato degli ottimi classificatori che poi, in casi reali, falliscono o ottengono prestazioni nettamente più basse rispetto a quanto riportato in fase sperimentale.

1.3 Risultati

I risultati all'interno di questa tesi sono frutto di un approccio ingegneristico e, in particolare, un approccio improntato al testing di sistemi di Machine Learning (ML). Dopo la costruzione della rete verranno fatti dei test e i risultati di questi saranno soggetti a delle valutazioni. Per valutare al meglio la qualità del sistema verranno introdotti dei parametri nel capitolo 3 per studiare come il software agisce per ogni classe, così da sottolineare eventuali problemi. La sperimentazione verrà suddivisa in due parti, testando il sistema prima in un contesto puramente sperimentale, quindi artificioso, poi andando a simulare la realtà attraverso l'ausilio di filtri che hanno l'obiettivo di simulare condizione atmosferiche o altre condizioni che influenzano la foto, per capire se il sistema si comporta bene anche al di fuori, e

quindi applicabile in un contesto reale. I test saranno suddivisi per tipologia di filtro applicato sulle immagini del dataset iniziale, e in questi test vedremo come queste immagini vengono classificate. La rete in questa ricerca costituirà un esempio di ciò che oggi possiamo trovare in letteratura, e quindi le valutazioni sono estendibili. I risultati, che si possono consultare nel capitolo 4, verificano l'ipotesi secondo cui i modelli di classificazione che oggi sono presenti in letteratura sono abbastanza scadenti in casi reali, il che implica la necessità di riconsiderare la direzione che hanno preso le ricerche nel campo per il riconoscimento di abbigliamento nelle immagini. Nel caso si volessero effettuare dei test il codice si può trovare su `https://github.com/AntonioCimino/Reconigtion_clothes_in_image`.

1.4 Struttura della tesi

L'elaborato si compone di quattro capitoli: Introduzione, Background e Stato dell'arte, Metodologia di ricerca, Risultati e Conclusioni. Nel primo capitolo verrà esplicato lo scopo del progetto, andando ad introdurre ciò che verrà fatto per raggiungere tale obiettivo. Nel secondo capitolo verrà spiegato di cosa tratta lo studio a livello teorico, cercando di capire più nello specifico le scelte fatte. Verranno approfonditi i lavori già presenti in natura che avevano uno scopo di base molto simile, analizzando pro e contro, e traendo informazioni utili per questo studio. Nel terzo capitolo invece ci sarà scritto quello che effettivamente è stato fatto, quindi come è stata strutturata la rete, come sono stati organizzati gli esperimenti e infine che tipi di metodi di valutazione sono stati utilizzati. Nel quarto capitolo vengono mostrati tutti i risultati ottenuti. I risultati vengono mostrati andando a suddividere le varie fasi di ricerca che comprende prima la costruzione della rete, poi il miglioramento con i test corrispettivi e infine gli esperimenti su quelle che sono le immagini che dovrebbero simulare la realtà. Nel quinto e ultimo capitolo vi è un commento generale sull'analisi condotta. Verrà fatto un resoconto dei risultati più significativi cercando di capire se alla fine è stato ottenuto un sistema che fa fronte al nostro obiettivo iniziale.

CAPITOLO 2

Background e stato dell'arte

2.1 Background

2.1.1 Classificazione delle immagini

Negli ultimi anni si è registrato un costante aumento di nuovi algoritmi e tecniche di classificazione. Quindi, è nata la necessità di identificare la tipologia di classificazione appropriata per uno studio specifico. Partiamo dalla natura del campione che si ha a disposizione. Nel caso in cui si ha una conoscenza preliminare di come debbano essere classificare gli elementi, allora si sta parlando di una classificazione supervisionata. Il vantaggio principale della classificazione supervisionata è la maggiore facilità con cui si possono rilevare gli errori e correggerli. Gli svantaggi di questa tecnica sono i tempi e i costi. Inoltre, i dati di allenamento iniziali potrebbero non evidenziare tutte le caratteristiche per una buona classificazione e quindi si può essere soggetti a errori umani [11]. Mentre se non esiste una conoscenza preliminare allora si può lavorare con i classificatori non supervisionati. I vantaggi in questo caso sono la velocità, l'assenza di errori umani e non vi è alcun obbligo di conoscenza preventiva dettagliata, perché sarà il sistema stesso a decidere come classificare gli elementi in base alle caratteristiche di questi ultimi. Il principale inconveniente di

questa tecnica sono i cluster separabili al massimo, perché la macchina non ha una guida, deve decidere da sé come separare le immagini e potrebbe sbagliare riconoscendo caratteristiche poco significative [11]. Altro elemento di valutazione per la scelta del classificatore è dato del tipo di oggetto su cui si va a lavorare. Tecniche di classificazione delle immagini possono essenzialmente essere suddivisi in due categorie: la classificazione basata sui pixel e classificazione basato su oggetti. I classificatori per pixel sono i classificatori tradizionali, aiutano a combinare i valori di tutti i pixel del set di allenamento con una determinata funzione specificata. La combinazione risultante conterrà i contributi di tutti i pixel del training set e ignorerà i problemi dei pixel misti. Esempio: distanza minima, massima probabilità, macchina vettoriale di supporto, rete neurale artificiale, albero decisionale [11]. Mentre con i classificatori orientati agli oggetti, la segmentazione delle immagini fonde i pixel in oggetti. La classificazione viene effettuata sulla base di oggetti e non su singoli pixel. Esempio: eCognition [11]. La scelta varia anche in base al numero di output generati. In questo caso si può avere una classificazione hard in cui classifichiamo l'immagine, basandoci sul pixel, nelle varie categorie. Questi algoritmi aiutano nella categorizzazione di tutti i pixel in classi. Può causare un gran numero di errori dai dati di risoluzione spaziale grossolana a causa dei pixel misti. Esempio: massima probabilità, macchina vettoriale di supporto, ISODATA (Classificazione non supervisionata), parallelepipedo, centroide (k significa), rete neurale, albero decisionale [11]. C'è anche la classificazione soft che è stata proposta come alternativa alla classificazione hard per trattare con pixel misti. Esempio: classificazione della massima probabilità, classificatori fuzzy-set, classificatore sub pixel, analisi della miscela spettrale [11]. Da qui è facile capire che per la classificazione delle immagini si dovrà optare per una classificazione tramite reti neurali, alberi decisionali, e così via. In particolare questo studio si concentrerà sulle reti neurali che tendono ad essere quelle più utilizzate in questi contesti, in particolare le reti neurali convoluzionali.

2.1.2 Rete neurale e reti neurale profonda

Ci sono un gran numero di pubblicazioni relative all'elaborazione delle immagini con reti neurali. Questo lavoro è legato a questo tipo di ricerche, dove la CNN viene

utilizzata per classificare le immagini. Le reti neurali artificiali (ANN), sono sistemi informatici ideati dall'osservazione del cervello umano. Il cervello, infatti, è formato da neuroni che si scambiano segnali per comunicare tra di loro. Allo stesso modo, una ANN si compone di nodi interconnessi. Ad ogni connessione è associato un valore numerico chiamato peso. Quando un nodo riceve l'input effettua una somma pesata, ovvero moltiplica ogni input per il peso della connessione su cui lo ha ricevuto e somma tutti i risultati. Al risultato di questa somma applica una funzione non lineare e restituisce un output. L'output rappresenta la predizione che viene fatta [Figura 2.1].

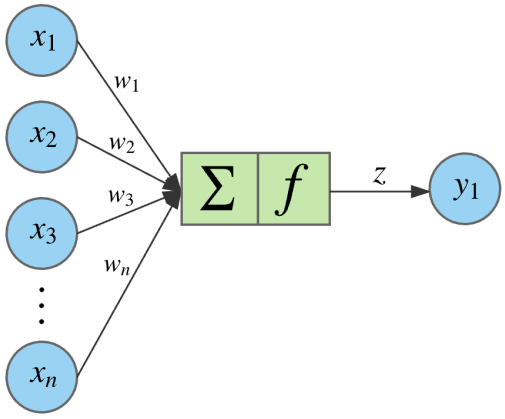

Figura 2.1: Immagine che mostra come i neuroni x generano un risultato y

I neuroni sono disposti su più strati connessi tra loro. Possiamo distinguere tre tipologie di strati [Figura 2.2]: abbiamo uno strato di input e uno di output rispettivamente posti all'inizio e alla fine, poi ci sono quelli che sono definiti strati nascosti, strati in cui viene effettuata la computazione descritta in precedenza.

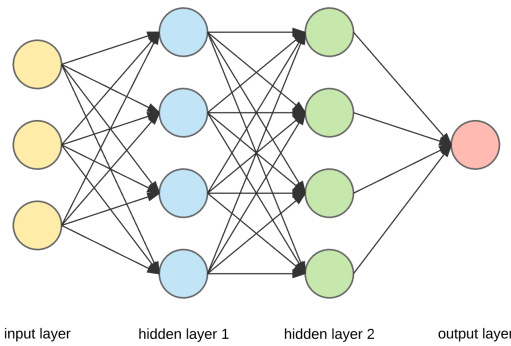

Figura 2.2: Struttura di una rete neurale

Una rete neurale artificiale viene addestrata usando un algoritmo chiamato backpropagation. Questo algoritmo cerca di minimizzare una funzione di perdita in modo iterativo andando a calcolare quali sono i valori ottimali dei pesi delle connessioni, ovvero quali valori producono una predizione migliore. Quindi la logica dall'algoritmo funziona andando in avanti fino alla fine e ottenere una predizione, a questo punto la rete calcola l'errore dell'output utilizzando una funzione di perdita per paragonare l'output giusto rispetto a quello ottenuto e restituisce la misura dell'errore. Data la misura di errore inizia a tornare indietro calcolando per ogni strato il contributo all'errore. L'algoritmo modifica i pesi per minimizzare la funzione di perdita.

Le reti neurali convoluzionali (CNN) sono un'evoluzione dell'architettura delle ANN. Una delle differenze principali è che i neuroni che compongono gli strati all'interno della CNN sono costituiti da neuroni organizzati in tre dimensioni, la dimensione spaziale dell'input (altezza e larghezza) e la profondità [12]. Le CNN sono composte da tre tipi di livelli [Figura 2.3]. Questi sono livelli convoluzionali, livelli di pooling e livelli completamente connessi. Quando questi livelli sono impilati, si è formata un'architettura CNN [12].

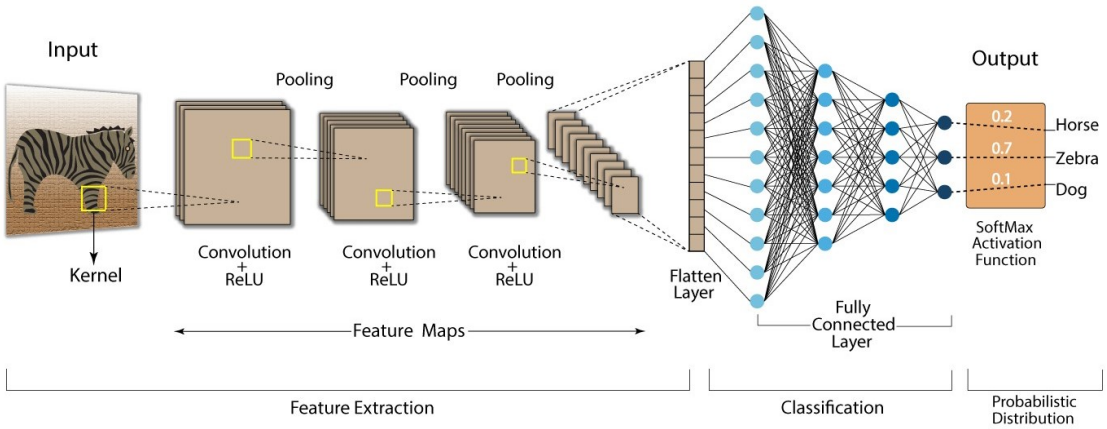

Figura 2.3: Struttura di una rete neurale convoluzionale, immagine di [1]

Gli strati convoluzionali sono la parte fondamentale delle CNN. In questi strati vengono definite delle matrici chiamate kernel o filtri. Questi kernel sono generalmente piccoli, ma si diffondono lungo la totalità dell'input. Quando i dati raggiungono un livello convoluzionali, lo strato fa ruotare ogni filtro sull'input producendo mappa

di attivazione. Da questo la rete imparerà i kernel che si "attivano" quando vedono una caratteristica specifica in una data posizione spaziale dell'input. Questi sono comunemente note come attivazioni [12]. Attraverso quella caratteristica poi cambierà la predizione finale. Si prende in considerazione solo una parte dell'input alla volta grande quanto la grandezza del kernel visto che si tratta di una moltiplicazione tra matrici. I livelli di pooling mirano a ridurre gradualmente la dimensionalità della rappresentazione, riducendo così ulteriormente il numero di parametri e la complessità computazionale del modello. Il livello di pooling opera su ciascuna mappa di attivazione nell'input e ne ridimensiona la dimensionalità utilizzando la funzione "MAX". Nella maggior parte delle CNN, questi si presentano sotto forma di livelli di max pooling con kernel di una dimensionalità di 2×2 applicati con un passo di 2 lungo le dimensioni spaziali dell'input. Questo ridimensiona la mappa di attivazione fino al 25% della dimensione originale, pur mantenendo il volume di profondità alla sua dimensione standard [12]. Lo strato completamente connesso contiene neuroni di cui sono direttamente collegati ai neuroni nei due strati adiacenti, senza essere collegati ad alcuno strato al loro interno. Questo è analogo al modo in cui i neuroni sono disposti nelle forme tradizionali di ANN [12].

2.1.3 Dataset (MMINST e Fashion MMINST)

Il database MNIST [Figura 2.4] fornisce un compito di classificazione statica relativamente semplice per ricercatori e studenti per esplorare l'apprendimento automatico e le tecniche di riconoscimento dei modelli, risparmiando sforzi inutili sulla formattazione dei dati. I classificatori di rete neurali tendono a funzionare significativamente meglio di altri tipi di classificatori sul MNIST. In particolare, le CNN hanno eccellenti prestazioni di classificazione. Le prestazioni migliori con MNIST raggiungono un tasso di errore dello 0,27% circa raggiunto con un insieme di reti convoluzionali [13]. Ma anche una singola rete neurale convoluzione molto grande e profonda dà anche un tasso di errore basso, corrispondente a 0,35% [14]. Aumentare i dati di allenamento è importante per ottenere tassi di errore molto bassi [15]. Anche la profondità delle reti neurali aiuta ad avere bassi tassi di errore. Senza la struttura della convoluzione e le tecniche preprocessing, come l'aumento dei dati,

il tasso di errore più basso in letteratura, 0,83%, è ottenuto utilizzando la rete neurale convessa/profonda [16].

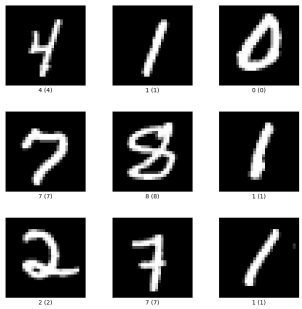

Figura 2.4: Immagini presenti nel dataset di MNIST

La ricerca in questione si concentra sul settore dell'abbigliamento e quindi non verrà utilizzato MNIST ma un altro dataset denominato F-MNIST. L' obiettivo di F-MNIST è quello di avere un buon set di dati di riferimento che ha tutta l'accessibilità di MNIST, vale a dire le sue piccole dimensioni, semplice codifica e licenza permissiva [9]. Hanno adottato l'approccio di le 70, 000 immagini in scala di grigi nella dimensione di 28x28 con 10 classi, come nella MNIST originale [Figura 2.5]. Fashion-MNIST si basa sull'assortimento di vestiti del sito Zalando, dove ogni prodotto di moda ha un set di immagini, che mostrano diversi aspetti del prodotto. Per le etichette di classe, viene utilizzato il codice silhouette del prodotto. L'immagine originale ha uno sfondo grigio chiaro e memorizzato in formato 762x1000 JPEG. Per essere utilizzata in modo efficiente per i componenti del front-end, l'immagine originale viene campionata con risoluzioni multiple [9]. I prodotti provengono da diversi gruppi di genere: uomini, donne, bambini e neutri.

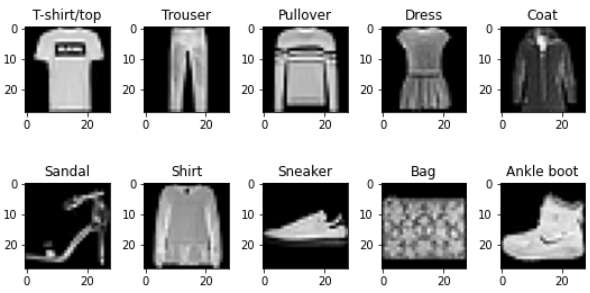

Figura 2.5: Immagini presenti nel dataset F-MNIST

Le miniature vengono convertite, per migliorare il train dei modelli che vogliamo costruire, seguendo il seguendo flusso di operazioni. Tale flusso è stato ripreso dall'articolo [9]:

1. Conversione dell'input in un'immagine PNG.
2. Rifila i bordi vicini al colore dei pixel degli angoli. La "vicinanza" è definita dalla distanza entro il 5% dell'intensità massima possibile nello spazio RGB.
3. Ridimensionando il bordo più lungo dell'immagine a 28 suddividendo i pixel, i.e. alcune righe e colonne vengono saltate.
4. Affilare i pixel usando un operatore gaussiano del raggio e della deviazione standard di 1.0, con effetto crescente vicino ai contorni.
5. Estendere il bordo più corto a 28 e mettere l'immagine al centro della tela.
6. Negando le intensità dell'immagine.
7. Conversione dell'immagine in pixel in scala di grigi a 8 bit.

Sono diversi gli studi effettuati sul dataset presentato con diversi metodi di classificazione. Nel prossimo pragrafo presentiamo dei lavori più nello specifico effettuati sul dataset F-MNIST.

2.2 Stato dell'arte

In questo paragrafo verranno presentati alcuni lavori effettuati nell'ambito del riconoscimento dei capi nelle immagini. L. Bossard, M. Dantone, et al. [2] introducono una pipeline per riconoscere e classificare gli abiti in scene naturali. Hanno creato il proprio set di dati con 80000 immagini etichettate in 15 classi. Vengono utilizzati un classificatore Random Forest (per tipi di abbigliamento) e SVM (per attributi di abbigliamento) [Figura 2.6]. Sul loro set di dati, attraverso le Random Forest hanno prodotto un'accuratezza che si aggira intorno al 40% e addirittura meno con SVM. Come si può ben capire risultati poco soddisfacenti.

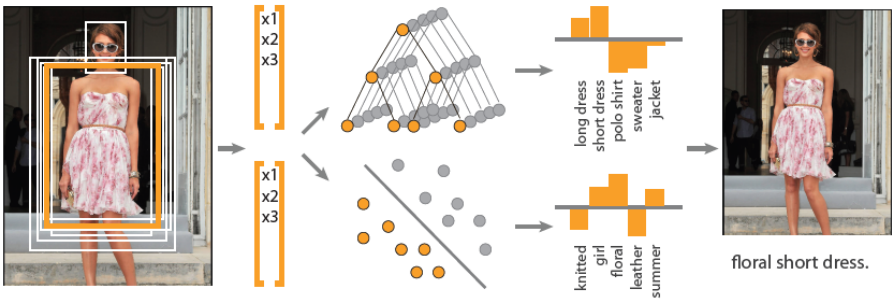

Figura 2.6: Struttura utilizzata nello studio [2]

Il limite di questo studio è dovuto principalmente al dataset che hanno definito loro, attraverso l'utilizzo di immagini prese da ImageNet, un dataset che contiene molto immagini di diverse categorie, anche abbigliamento. Il fatto di avere molte immagini non è per forza una buona cosa sopratutto se tali immagini spesso presentano rumori o contenuti non correlati, e questo non permette alla macchina di comprendere bene le caratteristiche per classificare le immagini. In questo lavoro non dovrebbe essere un problema visto l'utilizzo di una dataset consolidato come F-MNIST.

Greeshma KV e Sreekumar K. al [3] propongono di generare un modollo con un addestramento attraverso un SVM, utilizzando la tecnica HOG per estrare le caratteristiche. HOG è un descrittore di funzionalità veloce ed efficiente rispetto, per esempio, a SIFT. Principalmente viene utilizzato per il rilevamento di oggetti nell'elaborazione di immagini e nella visione artificiale. HOG divide l'immagine in piccole celle e calcola le direzioni dei bordi. Nella [Figura 2.7] è mostrata la visualizzazione della dimensione della cella [2 2], [4 4] e [8 8]. Da ciò si comprende chiaramente che la dimensione della cella [2 2] contiene più informazioni sulla forma rispetto alla dimensione della cella di [8 8] nella loro visualizzazione. La cosa migliore, come viene fatto nel loro studio, è scegliere una via di mezzo, ovvere [4 4], perchè troppe infromazione vuol dire molto calcolo computazionale, poche informazioni invece non riesce a far comprendere alla macchina le caratteristiche per la classificazione. Ottenute le caretteristiche poi è stato addestrato un modello attraverso SVM. Attraverso questa combinazione di tecniche lo studio ci mostra che riescono ad arrivare ad una percentuale di accuratezza poco inferiore al 90%. Lo studio in questo caso è stato effettuato su F-MNIST rispetto all'articolo precedente e si

può notare un grosso miglioramento. In questo caso, invece, il limite sta nell'utilizzo del SVM come algoritmo di addestramento.

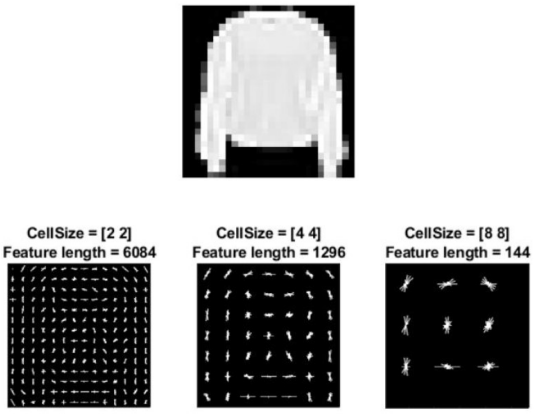

Figura 2.7: Esempio di funzionamento della tecnica HOG preso da [3]

Con SVM e altri algoritmi che non sono le reti neurali non si va oltre il 90%, e questo lo si può notare anche nell'articolo di Han Xiao, et al. [9] dove mostrano i risultati prodotti con diversi algoritmi, così da poterli mettere a confronto tra essi e successivamente anche con risultati prodotti da reti neurali. Tutti gli algoritmi vengono ripetuti cinque volte mescolando i dati di allenamento. Vengono riportata la precisione media sul set di test di alcuni degli algoritmi nella [Tabella 2.1].

Nome classificatore	Parametri	Acc. media
SVC	C: 10, kernel: rbf	0.897
KNeighborsClassifie	weights: distance, n_neighbors: 5, p: 1	0.854
GradientBoostingClassifier	n_estimators: 100, loss: deviance, max_depth: 10	0.880
DecisionTreeClassifier	criterion: entropy, max_depth:10, splitter: best	0.798
RandomForestClassifier	criterion:entropy, max_depth: 100, n_estimators: 100	0.873
LogisticRegression	C: 1, multi_class: ovr, penalty: l1	0.842

Tabella 2.1: Tabella con con i migliori risultati di alcuni degli algoritmi studiati in [9]

Ora verranno mostrati dei studi che utilizzano le reti convoluzionali, per fare il confronto con quelle precedenti. Shobhit Bhatnagar, et al. [4] utilizzano un modello, con 2 livelli di pooling convoluzionale e massimo uno dopo l'altro. Ogni strato convoluzionale ha 32 filtri di dimensioni 3x3 ed è stato eseguito il max-pooling. Successivamente l'output è stato appiattito e inserito in una rete la classificazione finale. Usano anche il dropout, prima dell'ultimo livello denso, settato al 50% come misura di regolarizzazione, per evitare che il modello si adatti molto ai dati di addestramento e generalizzi male per i dati di test.

Precisione del test (%)	Modelli
CNN2	91.17
CNN2 + BatchNorm	92.22
CNN2 + BatchNorm + Skip	92.54

Tabella 2.2: Tabella con i risultati dei test delle reti convoluzionali costruite

A questo modello poi è stato aggiunto in un secondo test la normalizzazione batch eseguita prima di ogni strato convoluzionale per migliorare la velocità di addestramento del modello. Infine in un terzo modello vengono utilizzati anche le Skip connection che, come suggerisce il nome, servono a fornire un percorso alternativo per il gradiente con backpropagation. Nella [Tabella 2.2] si possono osservare i risultati dei test effettuati attraverso l'utilizzo delle reti precedentemente descritte, e sono effettivamente risultati migliori rispetto a quelli degli studi precedenti, cosa che sta a dimostrare che con delle reti neurali convoluzionali si possono ottenere dei risultati migliori. Per questo lavoro di tesi si cercherà di costruire una rete neurale molto simile a quella dell'articolo [4] cercando però di migliorare i risultati, modificando la rete e cercando di lavorare sui parametri di allenamento. In [Figura 2.8] si possono osservare le tabella che evidenziano l'accuratezza e e la loss function. In natura esistono anche reti già consolidate che hanno dato dei risultati molto preformanti. Mohammed Kayed, et al [5] utilizzano l'architettura LeNet-5 per la classificazione delle immagini Fashion- MNIST.

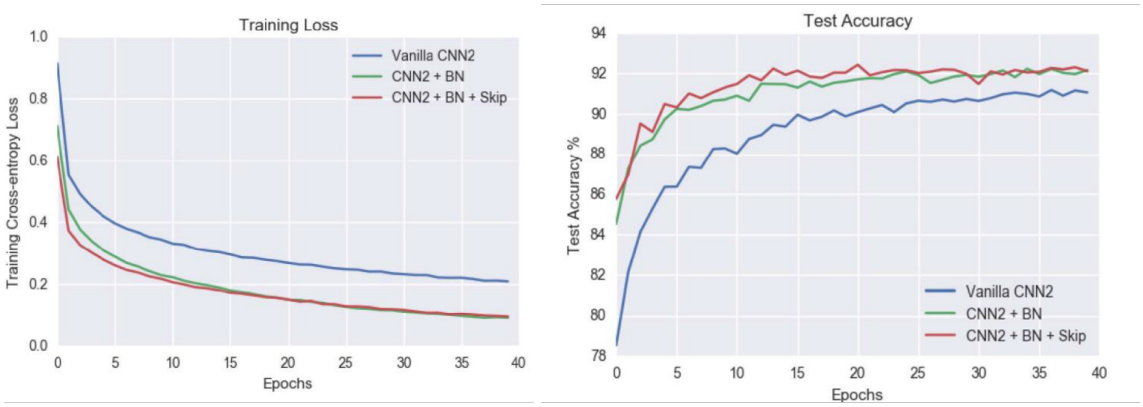

Figura 2.8: Grafici di accuratezza e loss function degli esperimenti nell'articolo [4]

La utilizzano perché è semplice e fornisce risultati ad alte prestazioni. Si basa su campi recettivi locali, pesi condivisi e un sotto-campionamento speciale. Nella [Figura 2.9] viene mostrata la struttura della rete, di cui vengono spiegati bene i vari strati cosa fanno nell'articolo. Per l'addestramento vengono utilizzati i seguenti iper-parametri: a = 0,005, batch size = 32.

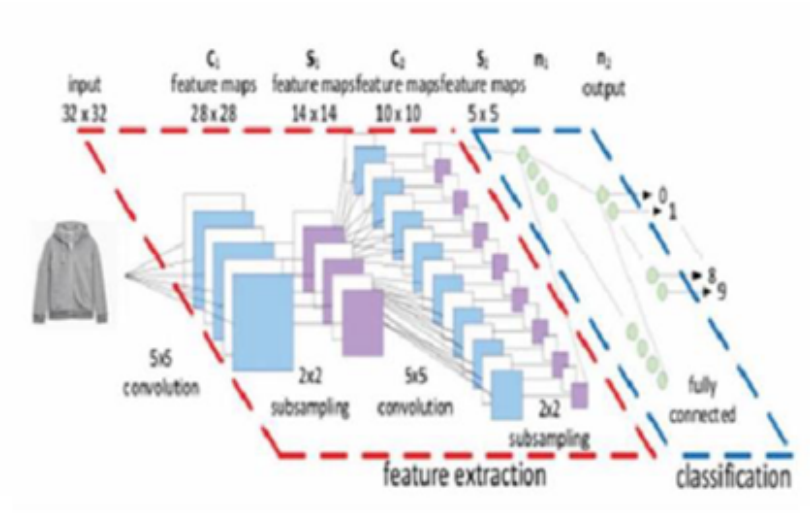

Figura 2.9: Rappresentazione della rete utilizzata in [5]

La [Figura 2.10] illustra l'accuratezza delle diverse suddivisioni di training e testing per le 10 categorie. Confrontando il grafico con quelle precedenti [Figura 2.8] possiamo notare una crescita più costante, il che già ci fa presumere un maggior successo. La media dell'accuratezza su tutte le 10 prove è stata del 98,9%.

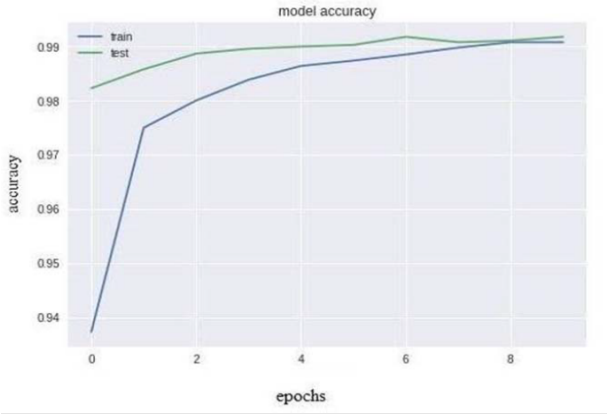

Figura 2.10: Grafici di accuratezza degli esperimenti nell'artcolo [5]

Sull'articolo vengono confrontati il loro modello LeNet-5 con gli altri modelli testati sul dataset Fashion MNIST (SVC, EDEN, e altri) e LeNet-5 raggiunge un'accuratezza superiore rispetto agli classificatori.

Per concludere l'analisi sugli studi attualmente presenti in natura è giusto ribadire in sintesi cosa verrà preso da questi e cosa invece si cercherà di evitare. Innanzi tutto rispetto al primo studio come già detto in precedenza verrà utilizzato un dataset come F-MNIST che è molto diffuso, utilizzato per tutti gli altri studi analizzati, e ha sempre portato ad avere ottimi risultati. Poi un'altra differenza rispetto a molti studi come è stato mostrato è costituita dal fatto che ci focalizzeremo sugli algoritmi di machine learning (reti neurali) che hanno sempre dato i risultati migliori. Infine per testare meglio i modelli generati cercheremo di ampliare i test, con l'obiettivo di analizzare l'ecological validity, ovvero l'efficienza dei sistemi fuori dal contesto sperimentale, visto che molti studi si soffermano solo sui test a livello sperimentale, come anche lo studio di LeNet-5 che risulta essere attualmente uno di quelli con le percentuali maggiori per quanto riguarda le accuratezza dei test.

CAPITOLO 3

Metodologia proposta

Lo scopo di questo progetto di tesi è quello di definire una rete che per generare un modello che possa riconoscere, al meglio possibile, i capi di abbigliamento nelle immagini, per poi riportarlo in un cotesto reale. Per valutare, alla fine, se il progetto ha portato al raggiungimento dell'obiettivo effettueremo due tipi di esperimenti sui modelli generati, uno per generalizzare il modello e un altro per capire se questo possa funzionare in una situazione più realistica. Quindi definiamo due interrogativi, uno per ogni tipologia di esperimento:

1. Il modello ottenuto riesce a classificare bene i capi presenti nelle immagini del dataset di partenza?

2. Il modello come si comporta in una situazione più realistica?

Entrambe le domande fanno riferimento alla valutazione del modello, quindi per entrambe verranno utilizzate le stesse metriche di valutazione, ma con la differenza dei dati con cui i modelli produrranno i valori di queste metriche. Per realizzare l'obiettivo dell'esperimento, è stata costruita una rete neurale convoluzionale con l'ausilio della libreria Keras [1]. L'allenamento del modello è stato effettuato su Colab

[1] Il sito web di keras: https://keras.io/

[2], uno strumento messo a disposizione da Google per utilizzare GPU da remoto. La rete utilizzata per generare i modelli è unica, e nel prossimo capitolo cercheremo di spiegarne la struttura. Sono stati effettuati diversi allenamenti cambiando il valore dei parametri di addestramento e successivamente con l'ausilio di una tecnica per aumentare il volume dei dati, da cui sono state generati diversi modelli.

3.1 Struttura delle rete

La struttura della rete la si può osservare nella [Figura 3.1] e man mano verrà spiegata la sua composizione e il perchè di alcune scelte effettuate.

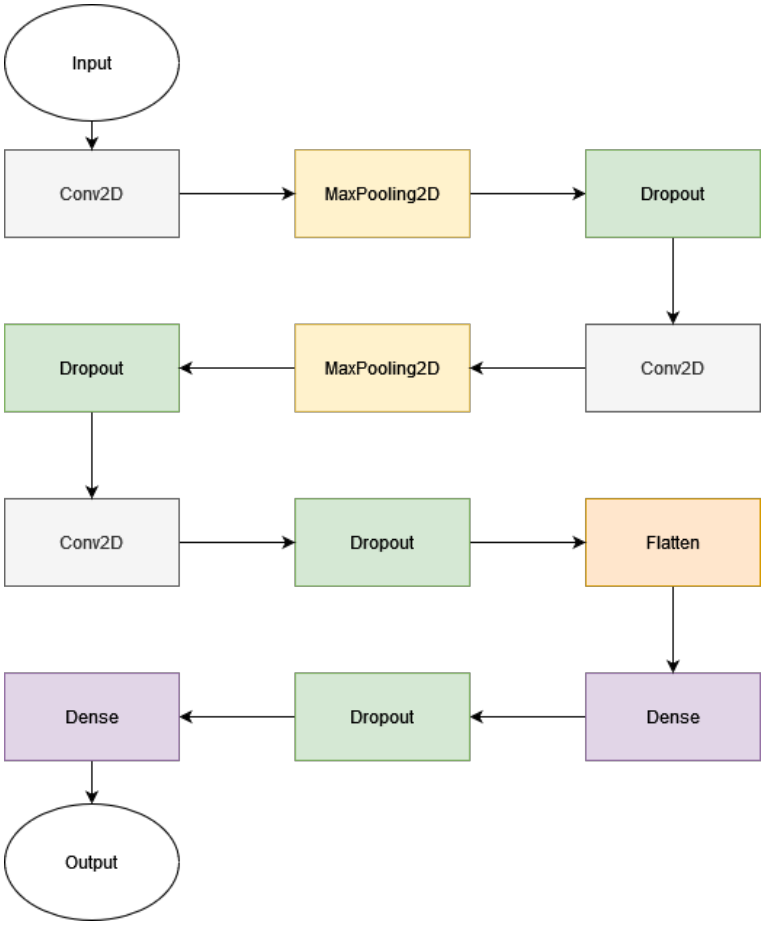

Figura 3.1: Struttura rete costruita

Conv2D è un livello di convoluzione 2D. Questo livello crea un kernel (o filtro) che è una matrice [Figura 3.2] di convoluzione utilizzata per modificare l'immagine in

[2]Il sito web di colab: https://colab.research.google.com/

vari modi (sfocare, aumentare la nitidezza, rilievo, rilevamento dei bordi e altro). Per questo livello abbiamo specificato un numero di kernel uguale a 32 nel primo livello di Conv, poi siamo andati a crescere con 64 e 128 con una dimensione ciascuno di 3x3 come [Figura 3.2]. E' stato poi specificato anche la dimensione dell'input, ovvero quanto deve essere grande l'immagine che andiamo a passare per l'addestramento e successivamente per i test, e la dimensione indicata è di 28x28. Altro paramentro importante per il livello convoluzionale è "activation", consente di fornire una stringa che specifica il nome della funzione che si desidera applicare dopo aver eseguito la convoluzione, in questo caso applica la funzione ReLu per quanto riguarda i livelli interni della reta e la funzione SoftMax per quanto riguarda la classificazione finale [17]. SoftMax e ReLu sono le funzioni più comunemente utilizzate e che garantiscono migliori risultati. Infine è stato specificato il modo con cui vengono impostati i pesi inizialmente. In questo caso verrà utilizzara una funzione normale. MaxPooling è un processo di discretizzazione, ovvero processo di trasformazione dei modelli matematici basati su equazioni continue in modelli matematici basati su equazioni discrete.

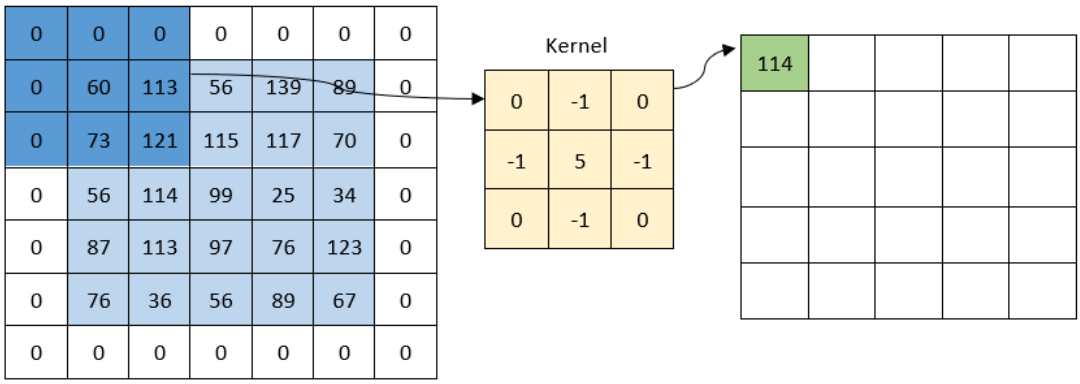

Figura 3.2: Funzionamento del kernel

L'obiettivo è di sotto campionare l'input, riducendone la dimensionalità e consentendo di formulare ipotesi sulle caratteristiche contenute nelle sotto-regioni. Ciò viene fatto in parte per aiutare l'overfitting fornendo una forma astratta della rappresentazione. Inoltre, riduce il costo computazionale riducendo il numero di parametri da apprendere. Il raggruppamento massimo viene eseguito applicando un filtro

massimo a sotto-regioni. In questo caso il filtro che eseguito sull' input è 2x2. Per ogni area 2x2 su cui il filtro passerà verrà preso il massimo di quella regione e verrà creata una nuova matrice di output in cui ogni elemento è il massimo di una regione nell'input originale [Figura 3.3]. Il dropout è una tecnica di regolarizzazione in cui i neuroni selezionati casualmente vengono "abbandonati" [Figura 3.4]. I pesi dei neuroni durante l'allenamento vengono sintonizzati per caratteristiche specifiche, fornendo una certa specializzazione.

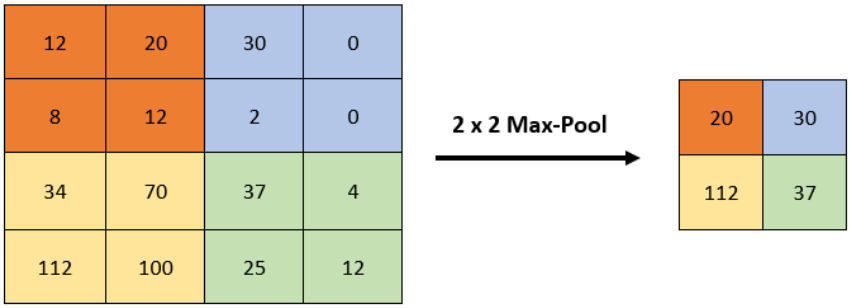

Figura 3.3: Funzionamento del MaxPooling

Se i neuroni vengono eliminati casualmente dalla rete durante l'allenamento, altri neuroni dovranno intervenire e gestire la rappresentazione richiesta per far fronte alla mancanza. Si ritiene che ciò comporti l'apprendimento da parte della rete di più rappresentazioni interne indipendenti. Il dropout aiuta a evitare l'overfitting, come spiegato in [18].

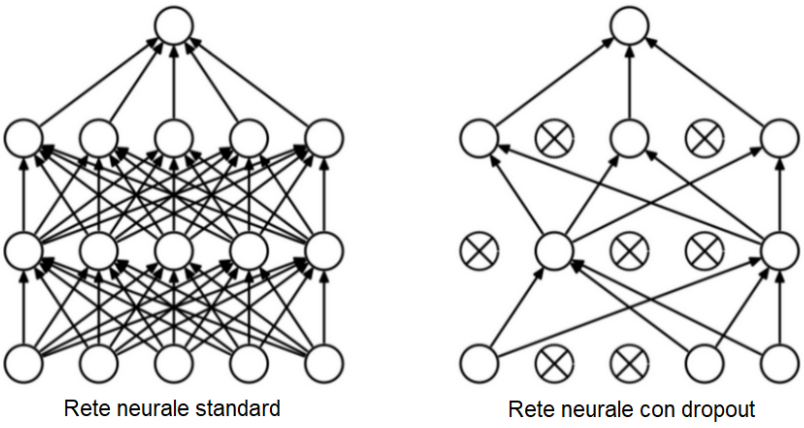

Figura 3.4: Funzionamento del Dropout, immagine di [6]

Il livello di Flatten serve ad appiattire gli input, ovvero serve a trasformare una matrice ad un array da passare al prossimo livello [Figura 3.5]. Si sta creando un modello di classificazione, il che significa che questi dati elaborati dovrebbero essere un buon input per il modello, quindi, deve avere la forma di un vettore lineare unidimensionale e non una forma rettangolare o cubica che non rappresentano degli input diretti. Questo livello appiattisce l'output dei livelli convoluzionali per creare un unico vettore di feature.

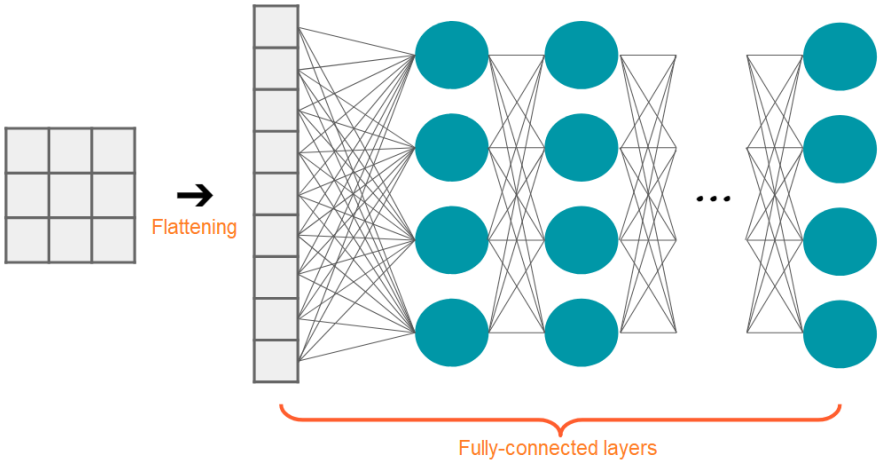

Figura 3.5: Funzionamento del livello di Flatten [7]

Questo livello poi è collegato al modello di classificazione finale, chiamato livello completamente connesso [Figura 3.5]. Infine per addestrare un modello, è necessario specificare una funzione di perdita, un ottimizzatore, ed eventualmente, alcune metriche da monitorare. Per quanto riguarda l'ottimizzatore è stato scelto Adam che è ritenuto uno dei migliori per le CNN [2, 3, 4].Adam è un ottimizzatore di discesa del gradiente stocastico che lavora su stime adattive. La discesa del gradiente è utile per regolare i pesi nei livelli nascosti.

3.2 Dataset

Il dataset utilizzato per effettuare questo lavoro di ricerca è Fashion-MNIST [19]. Questo dataset è stato scelto per via della sua validità e semplicità, questo dovuto al fatto che sia figlio di uno dei dataset più utilizzati, ovvero MNIST. Quest'ultimo è una raccolta di cifre scritte a mano e contiene 70000 immagini in scala di grigi

28x28, associate a 10 etichette. Fashion-MNIST ha la stessa identica struttura, ma le immagini sono prodotti di moda. Un esempio di questo set può essere visto nella [Figura 3.6]

Figura 3.6: Struttura del dataset di MNIST, immagine presa da [8]

Il set di dati può essere descritto come un CSV da 785 colonne. Ogni riga del CSV è un'immagine che ha una colonna con l'etichetta e le colonne rimanenti per descrivere l'immagine 28x28 pixel, con valori da 0 a 255 che rappresentano la luminosità dei pixel.

3.3 Data collection & analysis

Dopo aver specificato la rete e il dataset che verrà utilizzato bisogna distinguere le operazione che effettuiamo per dare risposta a quelle che sono le domande che sono state definite ad inizio capitolo, quindi prima verrà spiegato ciò che verrà fatto per l'esperimento in-vitro, ovvero quello che generalizza i risultati e poi successivamente per l'esperimento in-vivo, ovvero quello che dovrebbe simulare i problemi che possono avere le immagini nella vita reale.

3.3.1 Esperimento in-vitro

Per rispondere alla prima domanda sono stati effettuati dei train con il set di dati con cui si è parlato prima. Sul dataset non è stato fatto molto lavoro perché di per sé

già è sufficientemente consistente a livello qualitativo. Inizialmente la suddivisone dei dati tra set di train e test è 85/15 su 70000 immagini che compongono F-MNIST. Più nello specifico:

- T-shirt/top: Train 6000 or 10.0% and Test 1000 or 10.0%
- Trouser: Train 6000 or 10.0% and Test 1000 or 10.0%
- Pullover: Train 6000 or 10.0% and Test 1000 or 10.0%
- Dress: Train 6000 or 10.0% and Test 1000 or 10.0%
- Coat: Train 6000 or 10.0% and Test 1000 or 10.0%
- Sandal: Train 6000 or 10.0% and Test 1000 or 10.0%
- Shirt: Train 6000 or 10.0% and Test 1000 or 10.0%
- Sneaker: Train 6000 or 10.0% and Test 1000 or 10.0%
- Bag: Train 6000 or 10.0% and Test 1000 or 10.0%
- Ankle boot: Train 6000 or 10.0% and Test 1000 or 10.0%

Successivamente è stata applicato la data augumentation, è una tecnica che ci permette di aumentare il numero dei dati a disposizione andando a modificare le immagini iniziali con dei filtri per modificare il colore, ridimensionandole, spostare, ecc.. e poi aggiungere queste nuove immagini alle altre già utilizzate, per dei nuovi train. In questo caso sono stati mossi gli elementi nelle immagini come possiamo vedere in [Figura 3.7]. Questa tecnica serve ad evitare l'overfitting, ovvero se il modello tende ad avere a disposizione poche informazione per la classificazione e quindi non riesce a distinguere bene gli oggetti. All'inizio già si avevano a disposizione una buona quantità di dati ma sembra che grazie all'utilizzo di questa tecnica c'è stato un miglioramento sul risultato finale. L'idea è nata dal fatto che molti errori erano generati dalla somiglianza che esisteva tra alcuni prodotti (T-shirt e Shirt) che probabilmente venivano confusi dalla macchina e quindi si è provato ad aumentare la quantità di dati per permettere alla macchina di comprendere meglio le differenze, ed effettivamente è migliorata l'accuratezza sul singolo prodotto. Con questa tecnica

applicata sul set di train siamo arrivati a 300000 immagini per l'allenamento di cui poi un 20% abbiamo utilizzato per la validazione.

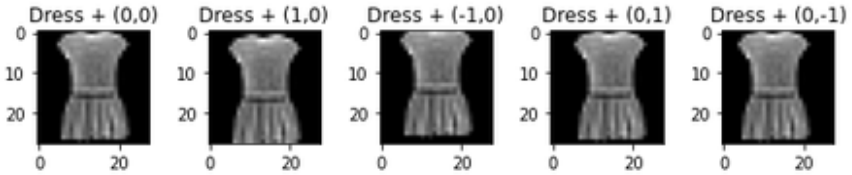

Figura 3.7: L'immagine di un vestito su cui è stata applicata la data augumentation

Sono stati effettuati diversi allenamenti a partire dalla rete descritta precedentemente generando diversi modelli. Gli allenamenti si distinguevano in base alla modifica del batch size, ovvero il numero di campioni che verranno propagati attraverso la rete, e il numero delle epoche, che definisce il numero di volte in cui l'algoritmo di apprendimento funzionerà sull'intero set di dati di addestramento. Il fine della ricerca è di rispondere alle domande che definite all'inizio del capitolo 3 e la prima delle due ci chiedeva se i modelli prodotti fossero efficienti in un contesto generico, e per valutare questa cosa esistono delle metriche. Innanzi tutto per ogni modello sono stati generati alla fine del train due grafici, di accuratezza e di perdita [Figura 3.8]

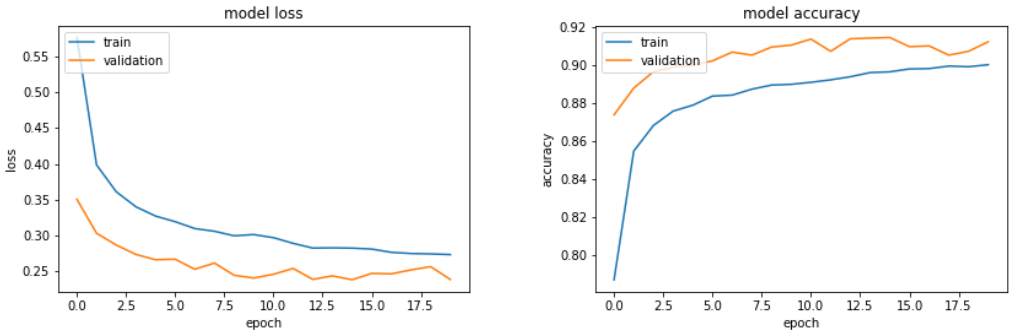

Figura 3.8: Esempio grafici di accuratezza e loss function degli esperimenti

Entrambi questi grafici riportano due dati al loro interno, rispettivamente accuratezza durante l'addestramento e "loss", con cui si intende la correzione effettuata sui valori che permettono la classificazione affinché l'output durante il train risultasse corretta, in pratica rappresenta il feedback della rete per correggere i pesi. L' analisi deve concentrarsi sul fatto che nel grafico di loss la curva deve tendere verso il basso e per l'accuratezza invece verso l'alto. Il modello generato poi deve essere valutato anche

in base a come si comporta sui dati di test che sono stati messi da parte, al fine di poter valutare la qualità del train.

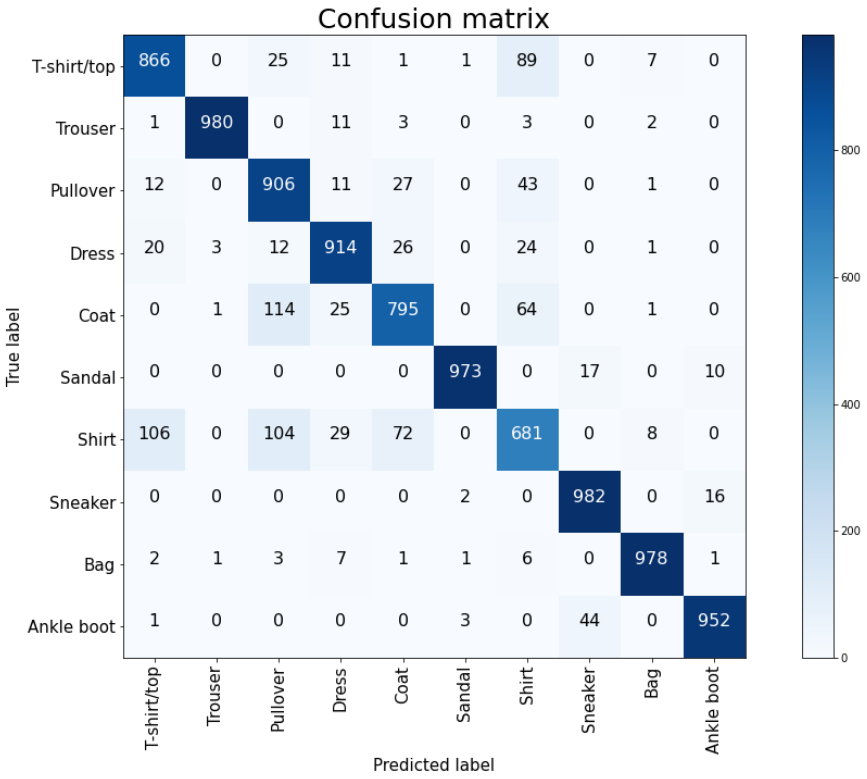

Figura 3.9: Esempio di matrice di confusione

Per ognuno dei modelli quindi viene effettuato un test con gli stessi dati in modo da capire ognuno di essi come si comporta con le stesso tipo di informazioni. Dai risultati dei test vengono calcolati dei parametri con cui poi potremmo fare delle osservazioni. I parametri sono raccolti rispettivamente per ogni singola classe e anche una media e sono:

- Recall che è la capacità di un classificatore di trovare tutte le istanze positive. Per ogni classe è definito come il rapporto tra i veri positivi e la somma dei veri positivi e dei falsi negativi.
- Precision che è l'abilità di un classificatore di non etichettare un'istanza positiva che è in realtà negativa. Per ogni classe è definito come il rapporto tra veri positivi e la somma di veri e falsi positivi.
- F-score è una media armonica ponderata delle metriche Precision e Recall in modo tale che il punteggio migliore sia 1 e il peggiore sia 0.

- Accuracy indica l'accuratezza del modello come dice il nome. Pertanto, la migliore accuratezza è 1, mentre la peggiore è 0.

In più sui risultati vengono prodotte anche le matrici di confusione come in [Figura 3.9] che aiutano a vedere il numero delle previsioni giuste e quelle sbagliate, ed è utile sopratutto per capire in che modo sbaglia la classificazione il software, cioè che tipo di classe individua al posto di quella giusta.

3.3.2 Esperimento in-vivo

Per rispondere invece al secondo quesito, ovvero come si comportano i modelli generati nella vita reali, quindi verificare l'ecological validity, è stato deciso di simulare le situazioni che possono verificarsi nel mondo reale. L'idea è stata quella di lavorare sulle immagini di test in modo che acquisissero un aspetto molto più realistico. Con aspetto più realistico si intende un'immagine non pulita, ovvero dove non si vede perfettamente il capo di abbigliamento. Questo tipo di sperimentazione vine fatto anche in [20], dove cercano di valutare il comportamento del software in presenza di situazioni speciali. L'intento è quello di provare a replicare condizioni atmosferiche reali, come per esempio quando c'è nebbia, con il "blur", oppure quando piove con un filtro "rain", o anche quando l'immagine non è totalmente visibile con un filtro che "occlude" l'immagine, ecc..

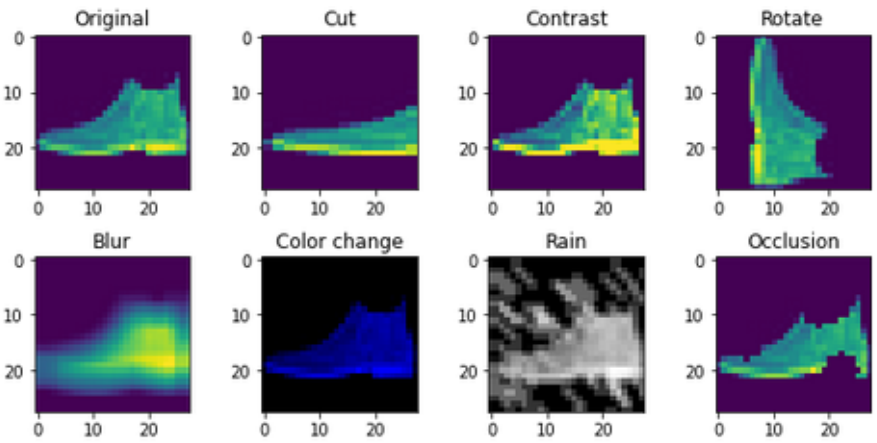

Figura 3.10: L'immagine di una scarpa che è stata modificata con i filtri utilizzati per i test

Da [20] infatti viene ripresa l'idea di quali tipi di filtri si possono implementare, aggiungendone qualcuno in più come il "cut". Quello che è stato fatto è prendere le immagini di F-MNIST, le 10000 di test, e andare ad applicare i diversi tipi di filtri scelti come si può vedere in [Figura 3.10]. Per ogni filtro è stata definita una funzione per modificare l'immagine, e tutte queste si possono trovare sulla repository di git [3], nel caso si volessero ripetere i test. Il codice è suddiviso in diverse sezioni e le funzioni dei filtri si trovano nella sezione "Metodi filtri". Queste funzioni sono servite per definire nuovi set di dati, ognuno formato dalle 10000 immagini del set di test modificate dal filtro. Quindi, per esempio, è stato creato un dataset, che chiameremo "dataset di immagini tagliate", in cui abbiamo inserito le 10000 foto che sono state lavorare con il filtro "cut_filter", che come indica il nome serve a tagliare l'immagine. Ottenuti i 7 nuovi dataset, uno per ogni funzione filtro, vengono valutati nuovamente i modelli, però stavolta con le immagini modificate, con lo scopo di capire come si comportano con delle immagini che simulano situazioni di vita reale, e quindi dare una risposta al secondo quesito. Come per il primo tipo di esperimento anche qui per la valutazione si useranno le stesse metriche specificate in precedenza (Recall, Accuracy, ecc.). Anche se i modelli e i dati di test sono gli stessi, con qualche modifica sui dati di test a seconda del filtro, i risultati, che verranno mostrati nel capitolo 4, varieranno rispetto ai primi e in alcuni casi anche di molto. Per ogni test si cercherà di capire, sopratutto grazie all'ausilio della matrice di confusione, cosa succede sulle singole classi.

[3] Link di github: https://github.com/AntonioCimino/Reconigtion_clothes_in_image

CAPITOLO 4

Risultati

In questo capitolo vengono mostrati i risultati per ogni test effettuato sui vari modelli, andando a visualizzare e commentare i grafici e i parametri che sono stati utilizzati per la valutazione, di cui si è parlato nel capitolo precedentemente.

4.1 Risultati iniziali

Inizialmente i primi modelli generati sono stati allenati con il dataset originale, senza l'ausilio di data augumentation, quindi sulla base delle 60000 immagini per l'allenamento, che poi è stato diviso con 85/15 tra dataset del train e validation. Ogni modello poi generato è stato testato con il dataset di test e nella [Tabella 4.1] vengono mostrati i risultati generali. Come si può notare in [Tabella 4.1] i risultati tendono ad avere tutti un valore superiore al 90% il che indica un ottimo risultato della rete. Il valore medio del 90% della precision indica che i modelli quando prevedono un elemento in una classe mediamente lo fanno bene. Mentre la recall sempre con percentuali medie sul 90% indica che tutte le classi sono utilizzate per classificare in modo proporzionale. Studiare risultati medi non da la possibilità di comprendere al meglio il comportamento della rete nello specifico sulle varie classi, e infatti sono stati raccolti anche i risultati sulle singole classi per ogni test effettuato, di modo

Id	Epoche	Batch size	Precision	Recall	F1-score	Accuracy
0	20	16	0.90	0.90	0.90	0.90
1	20	32	0.91	0.91	0.91	0.91
2	20	64	0.91	0.91	0.91	0.91
3	20	128	0.91	0.91	0.91	0.91
4	25	16	0.91	0.90	0.90	0.90
5	25	32	0.91	0.91	0.91	0.91
6	25	64	0.91	0.91	0.91	0.91
7	25	128	0.91	0.91	0.91	0.91
8	30	16	0.90	0.90	0.90	0.90
9	30	32	0.91	0.91	0.91	0.91
10	30	64	0.91	0.91	0.91	0.91
11	30	128	0.92	0.92	0.92	0.92

Tabella 4.1: Tabella con i risultati dei test delle reti convoluzionali costruite

che si possa capire meglio cosa succede e provare a migliorare l'efficienza se si può. Siccome i valori dei risultati nella [Tabella 4.1] si somigliano è giusto approfondire i singoli risultati sul test che ha restituito i valori maggiori, ovvero il modello indicato con id 11. La prima cosa che si osserva sono i grafici di accuratezza e di loss del modello preso in considerazione [Figura 4.1].

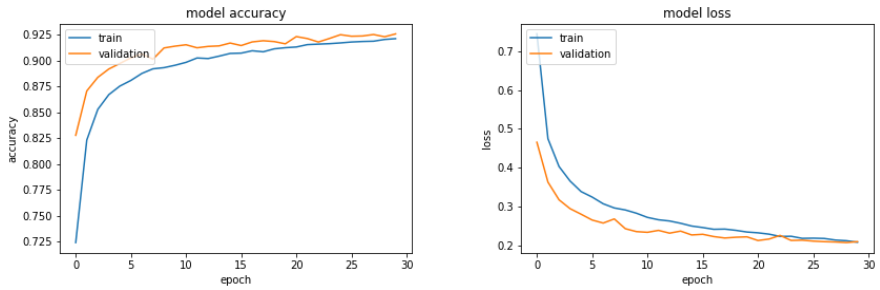

Figura 4.1: Grafici di accuratezza e loss function del modello con id 11

I due grafici mostrano, come visto anche dai risultati in [Tabella 4.1], che il modello funziona bene, anche perché in una analisi dei grafici si vuole sempre che quello

sull'accuratezza continui a crescere, perché comunque vuol dire che man mano riesce sempre più ad essere preciso nell' individuare la classe di appartenenza di un vestito e mentre per il grafico di loss si vuole una discesa sempre costante, perché questo vuol dire che sta riuscendo a trovare dei valori, che identificano le caratteristiche, sempre più efficienti. Quello che però si può notare è che i due grafici non si assestano nel crescere o nella discesa, quindi probabilmente possono fare ancora meglio. Per capire su cosa lavorare basta analizzare i risultati del modello 11 in modo più approfondito. Questi risultati sono visibili attraverso la matrice di confusione in [Figura 4.2].

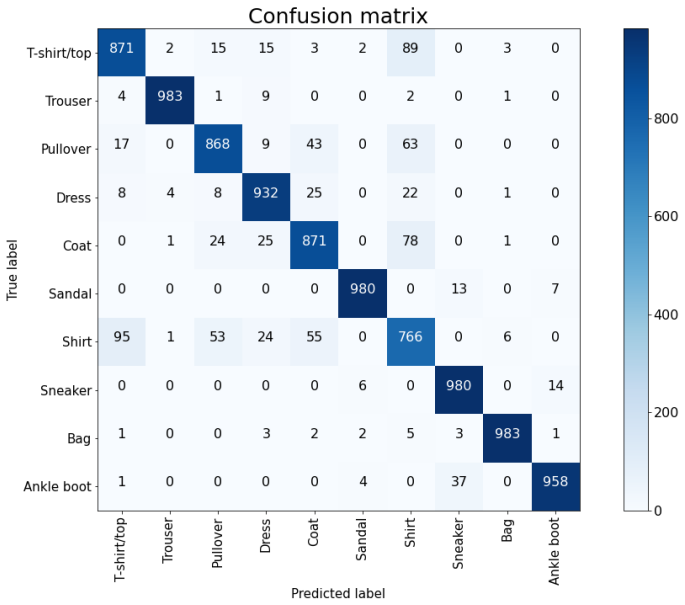

Figura 4.2: Matrice di confusione generata dai test con il modello con id 11

Nella matrice di confusione si può osservare che il problema maggiore è dato dalle previsioni della classe shirt, che ha una recall di 0.77, questo indica che un elemento su quattro viene classificato in una classe che non è quella giusta, e invece viene classificato come t-shirt o come coat, forse perché hanno una silhouette molto simile nelle immagini e quindi non riesce a distinguerle bene [Figura 4.3]. A questo punto l'idea è stata quella di provare ad allenare la rete con più elementi per migliorare anche le percentuali nelle classi su cui sbaglia più previsioni, magari avendo a disposizione più immagini di un elemento la rete riesce a estrapolare più informazioni e quindi riesce a generare un modello più performante, anche su quelle classi.

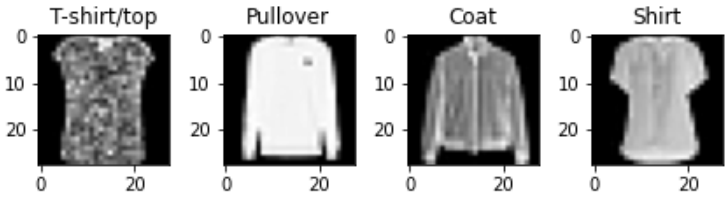

Figura 4.3: Classi che hanno silhouette simili

4.2 Risultati dei modelli con data agumentation

La data agumentation come è già stato spiegato serve a generare più elementi di quelle che si ha inizialmente, sfruttando gli elementi originali e modificandoli in diversi modi, e come già è stato mostrato nel capitolo 3 la scelta per questo lavoro di tesi è ricaduta sul riprodurre nuove immagini andando a spostare l'elemento al centro della foto. A questo punto per il train non ci saranno più 60000 immagini ma 300000 che sono state sempre suddivise tra set di train e validation con 85/15.

Id	Epoche	Batch size	Precision	Recall	F1-score	Accuracy
0	30	128	0.92	0.92	0.92	0.92
1	50	128	0.93	0.93	0.93	0.93
2	60	128	0.93	0.93	0.93	0.93
3	70	128	0.93	0.93	0.93	0.93
4	30	256	0.93	0.93	0.93	0.93
5	50	256	0.93	0.93	0.93	0.93
6	60	256	0.93	0.93	0.93	0.93
7	70	256	0.93	0.93	0.93	0.93

Tabella 4.2: Tabella con i risultati dei test con data augumentation

A questo punto sono stati riaddestrati i modelli partendo da quello che aveva data risultati migliori con il precedente train, ovvero con i parametri impostati su 30 epoche e con 128 di batch size, e da qui sono stati settati i parametri a crescere, visto che prima facendo così i risultati ne hanno beneficiato. Siccome abbiamo modelli

nuovi è giusto rianalizzare tutti gli elementi di cui si è discusso prima per vedere se ci sono stati dei miglioramenti. Tutti i risultati sono visibili nella [Tabella 4.4]. Nella tabella si può notare che i risultati sono migliorati, ovviamente i modelli hanno beneficiato sia del cambio ulteriore dei parametri sia grazie all'aumento di dati a disposizione. Nella matrice di confusione [Figura 4.4] si può notare che però l'intento, ovvero quello di migliorare i risultati su la classe che maggiormente veniva sbagliata (shirt), non è stato raggiunto, perchè la precisione è addirittura diminuita ma i risultati sono migliorati sulla precion raggiungendo un 82%, il che ha influenzato positivamente le altre classi con shiloutte simile.

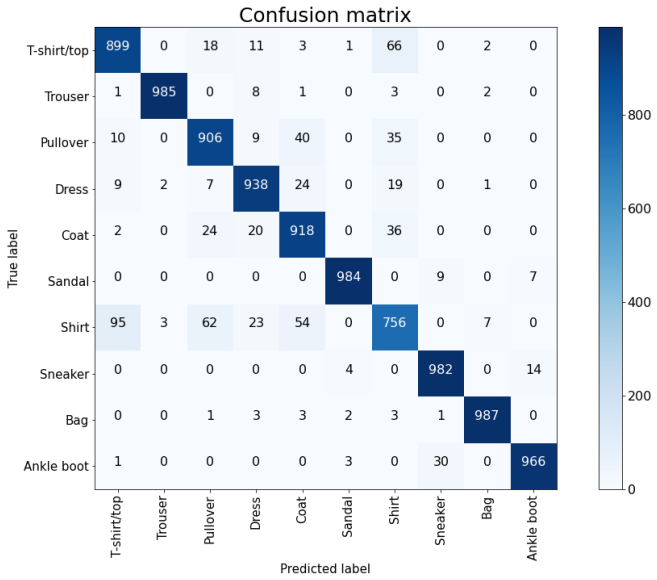

Figura 4.4: Matrice di confusione del con id 5

4.3 Risultati dei modelli con immagini filtrate

In questo paragrafo l'obiettivo sarà quello di capire la potenza reale dei modelli generati, perché comunque è semplice riconoscere immagini simili a quelle con cui sono stati allenati ma il tutto nella realtà diventa nettamente più complesso per via della scarsa qualità che le immagini potrebbero avere. Quindi si andranno ad analizzare i risultati dei modelli testati su immagini costruite attraverso l'utilizzo dei filtri di cui si è parlato nel capitolo 3. Il paragrafo si compone di sotto paragrafi, ognuno per ogni tipo di filtro utilizzato, dove verranno analizzati i risultati e mostrate

le immagini su cui questi test sono stati effettuati, mostrando non esempi come nel capitolo 3, dove l'intento era solo presentare i vari filtri, ma mostrando come per ogni classe agisce il filtro.

4.3.1 Immagini con filtro Blur

Nella [Figura 4.5] si possono visualizzare le immagini originali con il filtro blur applicato, questo filtro va a sfocare l'immagine e simula sostanzialmente la possibilità che lo strumento utilizzato per catturare l'immagine possa non riprendere perfettamente l'immagine, o perché in movimento o perché in condizioni atmosferiche particolari come nebbia. Nella [Tabella 4.3] si può osservare come i modelli che prima avevano percentuali altissime, il che era sinonimo di modelli efficienti, ora hanno dei valori drasticamente più bassi. Vedendo questi valori i modelli possono anche essere definiti poco efficaci, ma comunque bisogna pur sempre considerare la situazione speciale che è stata creata, che ovviamente è sinonimo di ostacolo alla classificazione.

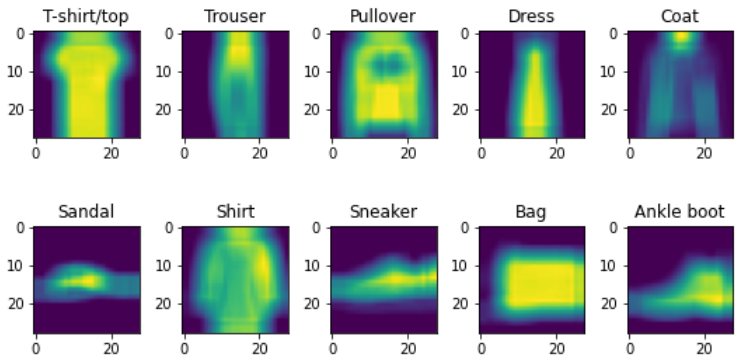

Figura 4.5: Rappresentazioni di tutte le classi con il filtro blur

Il modello migliore in questo caso può esser considerato quello con id 3, che prima comunque risultava essere tra i migliori, quindi per capire eventuali problemi viene presa la sua matrice di confusione. Quello che si può osservare nella matrice in [Figura 4.6] è che l'effetto applicato ha peggiorato tutte le previsione ma in particolare ha quasi azzerato le possibilità di individuare per bene la classe sandal e coat, probabilmente perché sono quelle che come vediamo da [Figura 4.5] hanno risentito di più di questo filtro, fino a diventare quasi irriconoscibili. Queste due classi vengono confuse con altre perché non hanno più quei dettagli che permetteva

Id	Epoche	Batch size	Precision	Recall	F1-score	Accuracy
0	30	128	0.58	0.44	0.38	0.44
1	50	128	0.42	0.40	0.33	0.40
2	60	128	0.56	0.43	0.37	0.43
3	70	128	0.59	0.52	0.46	0.52
4	30	256	0.44	0.47	0.40	0.47
5	50	256	0.59	0.50	0.44	0.50
6	60	256	0.59	0.50	0.45	0.50
7	70	256	0.50	0.49	0.44	0.49

Tabella 4.3: Tabella con i risultati dei test con filtro blur

il modello di classificarli correttamente, e si ripresenta il problema di prima in cui il sistema confondeva le classi solo perché avevano la stessa silhouette. Il sandal non viene più distinto dalle sneakers e dalle boot, infatti nella matrice vediamo che il sistema va proprio a sbagliare individuando i sandali in queste due classi e il coat, invece, viene confuso con il pullover.

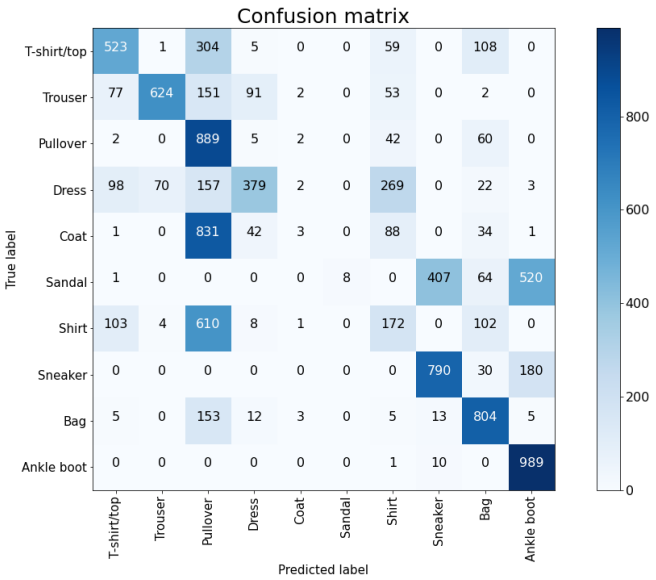

Figura 4.6: Matrice di confusione dei test con il filtro blur

4.3.2 Immagini con filtro Contrast

Nella [Figura 4.7] si possono visualizzare le immagini originali con il filtro contrast applicato, questo filtro come si può vedere serve a rendere i bordi delle figure più nitidi. Questo filtro va a simulare l'effetto che potrebbe esserci nel momento in cui può essere molta luce nell'ambiente circostante.

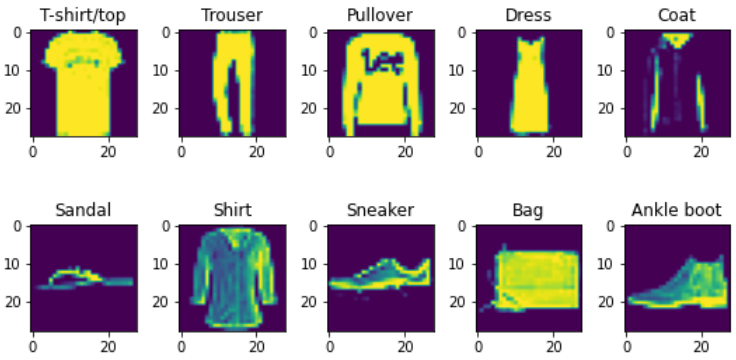

Figura 4.7: Rappresentazioni di tutte le classi con il filtro contrast

Nella [Tabella 4.4] si può osservare come modelli rispetto alla situazione che si verificava con il filtro blur riescono ad avere una risposta migliore con questo altro filtro.

Id	Epoche	Batch size	Precision	Recall	F1-score	Accuracy
0	30	128	0.85	0.84	0.84	0.84
1	50	128	0.84	0.84	0.84	0.84
2	60	128	0.84	0.84	0.84	0.84
3	70	128	0.83	0.82	0.82	0.82
4	30	256	0.85	0.84	0.84	0.84
5	50	256	0.84	0.84	0.84	0.84
6	60	256	0.85	0.84	0.84	0.84
7	70	256	0.85	0.84	0.84	0.84

Tabella 4.4: Tabella con i risultati dei test con filtro contrast

Le percentuali anche se diminuiscono, perché questo filtro comunque genera una perdita di informazioni, riescono comunque a mantenersi su valori molto alti. In questo caso non viene presentata la matrice di confusione perché mantiene una forma molto simile a quella in cui abbiamo testato le immagini originali, con degli errori in più, ma distribuiti sempre alla stessa maniera.

4.3.3 Immagini con filtro Cut

Nella [Figura 4.8] si possono visualizzare le immagini originali tagliate a metà. Il taglio per ogni classe è stato fatto verticalmente praticamente al centro dell'immagine. Quello che si vuol simulare con questo filtro è la situazione in cui in determinato abito all'interno dell'immagine non è completamente ripreso. Nella [Tabella 4.5] si può vedere come i modelli ora hanno dei valori drasticamente più bassi, come nel caso delle immagini su cui era stato applicato il filtro del blur. Questo fa presumere che nel caso in cui il modello non riesca ad inquadrare completamente il capo di abbigliamento può avere dei grossi problemi, dovuti a delle grosse perdite di informazioni, probabilmente più nello specifico la forma, su cui il modello si basava per classificare alcune immagini in specifiche classi.

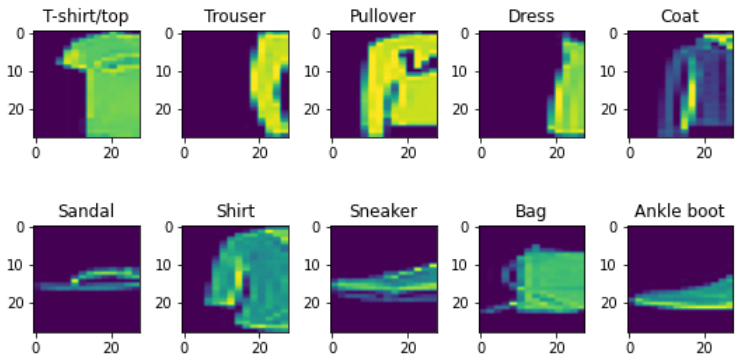

Figura 4.8: Rappresentazioni di tutte le classi con il filtro contrast

Le percentuali sono così basse per via di alcune classi che ora non vengono praticamente più considerate per la classificazione come si può vedere nella matrice di confusione generata dal modello che ha dato dei risultati migliori[Figura 4.9].

Id	Epoche	Batch size	Precision	Recall	F1-score	Accuracy
0	30	128	0.30	0.25	0.19	0.25
1	50	128	0.33	0.31	0.23	0.31
2	60	128	0.31	0.28	0.21	0.28
3	70	128	0.28	0.28	0.21	0.28
4	30	256	0.32	0.28	0.21	0.28
5	50	256	0.33	0.32	0.27	0.32
6	60	256	0.29	0.26	0.19	0.26
7	70	256	0.30	0.26	0.20	0.26

Tabella 4.5: Tabella con i risultati dei test con filtro cut

La classe del trouser e del dress hanno delle percentuali di precision sotto lo il 10%, il che vuol dire che non riesce praticamente più a riconoscere, ed entrambi vengono confusi con la classe del sandal, probabilmente perché così tagliate se viste in orizzontale danno la sensazione che potrebbero essere dei sandal.

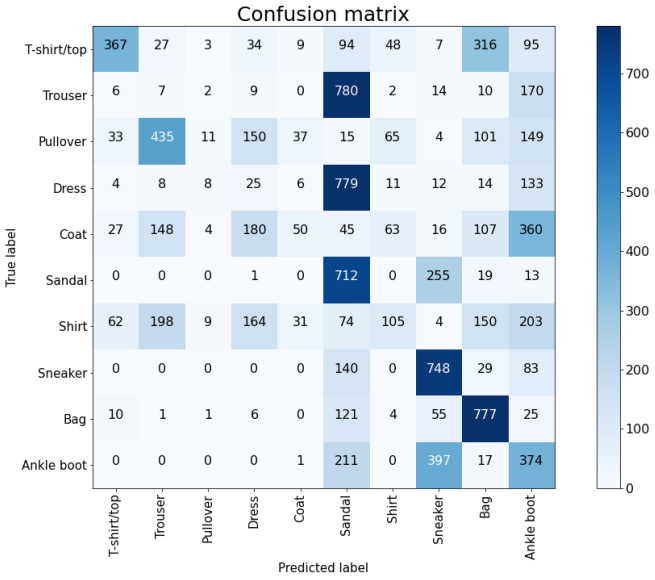

Figura 4.9: Matrice di confusione dei test con il filtro cut

Queste due classi sono nettamente le peggiori, ma comunque su tutte c'è un grosso peggioramento tranne che su tre che sono il sandal, la snekers e la bag. In generale

queste tre classi anche se divise a metà hanno mantenuto molto dei propri elementi che li rendono distinguibili. Si può fare una ulteriore precisazione sul caso del sandal, praticamente la maggior parte degli elementi vengono classificati in quel modo quindi è molto semplice che vengano riconosciuti molti degli elementi che appartengono effettivamente a quella classe, infatti questa classe ha una precision del 24% ma una recall del 70%.

4.3.4 Immagini con filtro Rain

Nella [Figura 4.10] si possono vedere le immagini con il filtro rain. Questo filtro come si può intuire dal nome va a simulare la condizione atmosferica della pioggia ed è stato creato andando a generare delle linee che vengono applicate casualmente sull'immagine in maniera obliqua e successivamente sfogando leggermente l'immagine.

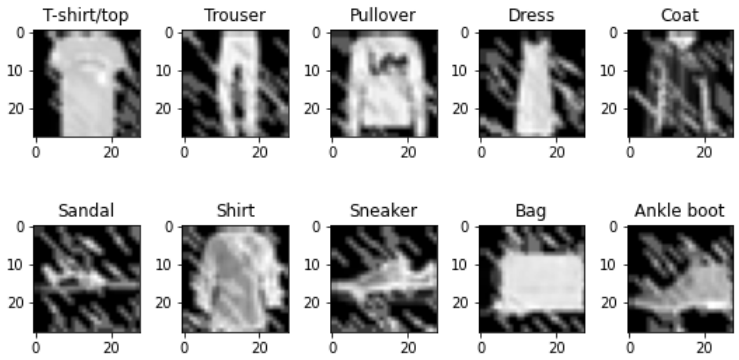

Figura 4.10: Rappresentazioni di tutte le classi con il filtro rain

Nella [Tabella 4.6] i modelli peggiorano, ma nemmeno di troppo visto l'applicazione del filtro. Quello che si può notare è che comunque anche se l'accuratezza praticamente viene dimezzata rispetto ai test con le immagini originali, la precison rimane comunque elevata. Come sempre però per capire il perché si devono studiare i singoli casi, e verrà preso come riferimento il modello con id 4. Nella [Figura 4.11] si può notare una cosa molto interessante per quanto riguarda la classe shirt. Infatti, mentre quando venivano effettuati i test con le immagini originali la recall di questa classe tendeva ad essere la più bassa ora invece è diventata la più alta e anche di molto.

Id	Epoche	Batch size	Precision	Recall	F1-score	Accuracy
0	30	128	0.69	0.54	0.55	0.54
1	50	128	0.67	0.43	0.44	0.43
2	60	128	0.66	0.48	0.49	0.48
3	70	128	0.71	0.58	0.58	0.58
4	30	256	0.72	0.58	0.59	0.58
5	50	256	0.70	0.56	0.57	0.56
6	60	256	0.67	0.50	0.49	0.50
7	70	256	0.70	0.53	0.54	0.53

Tabella 4.6: Tabella con i risultati dei test con filtro rain

Quelle che erano le classi [Figura 4.3] con la stessa silhouette con cui venivano classificate molte delle shirt sbagliando ora vengono classificate esse stesse come shirt, praticamente si sono invertiti i ruoli, questo successo perché per via della perdita di informazione che è avvenuta sugli elementi. Le t-shirt, il pullover e il coat senza elementi distintivi tendono ad essere visti come semplici shirt.

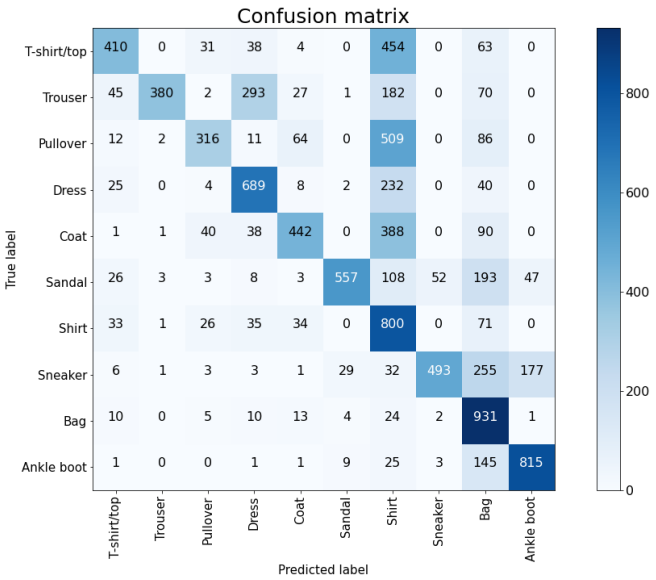

Figura 4.11: Matrice di confusione dei test con il filtro rain

4.3.5 Immagini con filtro Occlusion

Nella [Figura 4.12] si può osservare come agisce il filtro occlusion. Questo filtro va a generare dei buchi sugli abiti attraverso la generazione casuale di pallini neri, e l'effetto che si vuol ricreare è molto simile a quello che abbiamo visto con il cut, ovvero simulare una situazione in cui non tutto il capo di abbigliamento è visibile, però in questo caso l'idea è di simulare la situazione in cui, magari, davanti all'elemento da classificare ci possano essere altri oggetti. Nella [Tabella 4.7] come sempre vengono mostrati i valori dei test e qui questi tendono ad abbassarsi, come si è potuto constatare ormai è una costate in questi test, rispetto ai test effettuati con le immagini originali, ma bisogna sempre specificare il fatto che è una cosa che ci può stare visto la perdita di informazioni, c'è bisogno però di soffermarsi, in particolare, su una cosa che riguarda la classe del sandal.

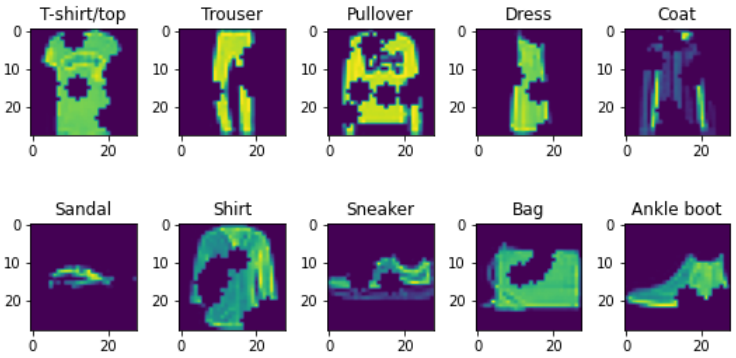

Figura 4.12: Rappresentazioni di tutte le classi con il filtro occlusion

Prendendo in considerazione sempre il modello migliore, in questo caso il modello 0, il valore della recall su questa classe risulta altissimo, tocca il 98% ma con una precision molto bassa intorno al 30%. Questa stessa caratteristica del test già è stata vista quando è stato valutato il test sulle immagini tagliate, che guarda caso è un test molto simile a quello delle occlusion. Questo può significare che è proprio una proprietà del sistema quella di concentrarsi su certe classi e lasciare indietro altre nel momento in cui riesce a vedere meno, o meglio, non ha la silhouette completa del capo di abbigliamento.

Id	Epoche	Batch size	Precision	Recall	F1-score	Accuracy
0	30	128	0.59	0.50	0.49	0.50
1	50	128	0.57	0.42	0.42	0.42
2	60	128	0.57	0.39	0.39	0.39
3	70	128	0.59	0.37	0.37	0.37
4	30	256	0.60	0.39	0.39	0.39
5	50	256	0.60	0.43	0.43	0.43
6	60	256	0.57	0.37	0.37	0.37
7	70	256	0.61	0.45	0.45	0.45

Tabella 4.7: Tabella con i risultati dei test con filtro occlusion

4.3.6 Immagini con filtro Rotate

Nella [Figura 4.13] ci sono le immagini originali ruotate di 90 gradi. Questo filtro è utilizzato per testare la capacità del sistema di individuare il tipo dei capi anche se viste in un'altra prospettava.

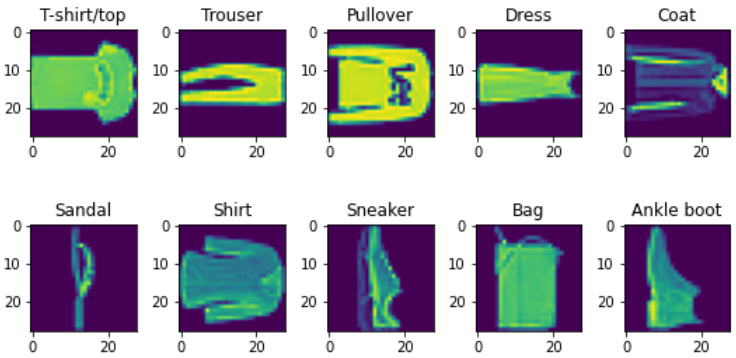

Figura 4.13: Rappresentazioni di tutte le classi con il filtro rotate

Questo tipo di test è stato lasciato come ultimo da analizzare, tra i test fatti con i modelli con data augumentation, per via dei risultati. Nella [Tabella 4.8] sono palesi i risultati disastrosi rispetto alle altre casistiche, tutti i valori sono crollati intorno al 10%. Anche per questi risultati disastrosi però esiste una spiegazione che è intuibile attraverso la matrice di confusione [Figura 4.14].

Id	Epoche	Batch size	Precision	Recall	F1-score	Accuracy
0	30	128	0.10	0.06	0.03	0.06
1	50	128	0.10	0.08	0.06	0.08
2	60	128	0.08	0.08	0.06	0.08
3	70	128	0.14	0.09	0.07	0.09
4	30	256	0.10	0.08	0.06	0.08
5	50	256	0.10	0.09	0.06	0.09
6	60	256	0.11	0.09	0.06	0.09
7	70	256	0.11	0.09	0.07	0.09

Tabella 4.8: Tabella con i risultati dei test con filtro blur

Prendendo in considerazione la matrice del modello con id 7, che è il modello con i risultati migliori, praticamente tutte le predizioni si sono concentrate su una sola classe, ovvero la classe bag. Probabilmente i modelli hanno appreso che quando l'immagine è in un certo verso e ha una certa forma è una bag e andando a ruotare tutte le immagini vengono classificate tutte in quel modo.

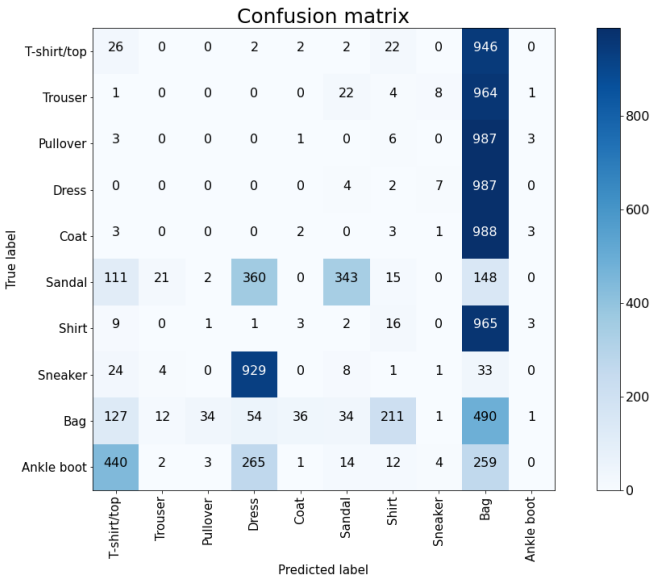

Figura 4.14: Matrice di confusione con il filtro rotate

4.4 Risultati dei modelli con immagini colorate

Per lo studio delle immagini colorate apriamo un nuovo paragrafo, per discostare questi esperimenti dagli esperimenti precedenti, perché come è stato precedentemente scritto per il test con questo tipo di immagini sono stati riaddestrati i modelli per specificare che gli input che riceveva ora non avevano più un unico canale di colori ma 3. Quindi ora andremo a valutare modelli riaddestrati completamente da zero, con una data augumentation minore rispetto ai precedenti visto che il sistema andava in crash con quella quantità di immagini con tre canali, e da 300000 ora il train è stato effettuato con 120000 immagini. Come si può vedere in [Figura 4.15] le immagini sono colorate o di rosso, o di verde o di blu, per il resto l'immagine rimane invariata.

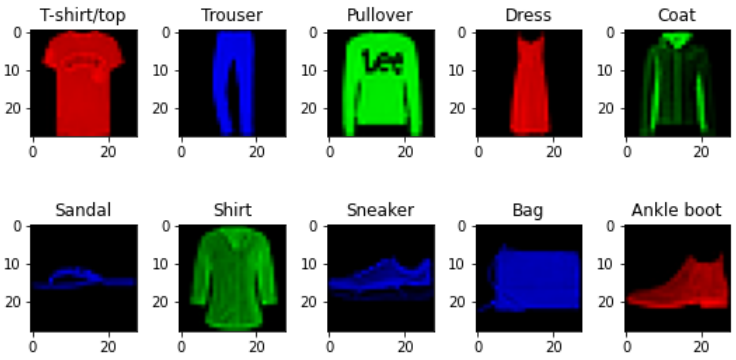

Figura 4.15: Rappresentazioni di tutte le classi con il filtro colored

Nella [Tabella 4.9] si può osservate subito una differenza rispetto agli altri esperimenti, che ci sono due modelli in meno. Il train per modelli con 70 epoche mandavano in crash il sistema, e quindi non si ha a disposizione quei modelli per riuscirli a testare. In questo test comunque sono stati ottenuti risultati molti buoni, quasi quanto l'esperimento con le immagini originali, però ragionandoci con attenzione visto che in questo caso test e train come nel primo esperimento hanno lo stesso formato di immagine, come mai non hanno risultati alti tanto quanto? Alla domanda si può rispondere ipotizzando che il problema principale sta nel fatto che nel train il sistema non ha a disposizione 3 immagini per ogni capo, visto che abbiamo 120000 immagini, ovvero il doppio del dataset iniziale, di conseguenza non si ha una immagine per ogni

Id	Epoche	Batch size	Precision	Recall	F1-score	Accuracy
0	30	128	0.82	0.79	0.80	0.79
1	50	128	0.83	0.80	0.81	0.80
2	60	128	0.75	0.70	0.71	0.70
3	30	256	0.84	0.78	0.78	0.78
4	50	256	0.86	0.83	0.83	0.83
5	60	256	0.82	0.77	0.77	0.77

Tabella 4.9: Tabella con i risultati dei test con filtro color

capo del dataset che viene riprodotta in tutti e 3 i tipi di colori presi in considerazione, questo porta il dataset per il train ad essere squilibrato. Quindi probabilmente per alcuni elementi il sistema riesce ad individuare meglio la classe di appartenenza quando è di un determinato colore.

4.5 Riassunto dei risultati

In questa sezione vengono sintetizzati tutti i risultati di cui si è discusso prima, in un'unica tabella [Tabella 4.10], per farne una analisi più veloce e semplice. La tabella si compone di quattro colonne di cui nella prima vengono specificati che modelli sono stati utilizzati per il test, nella seconda le immagini del test con cui sono stati ottenuti i risultati, nella terza aggiungiamo il riferimento alla tabella con i risultati e infine nell'ultima si può leggere una sintesi delle osservazioni fatte in precedenza. Se si vogliono approfondire determinati risultati consultare i capitoli specifici. Nel prossimo capitolo si cercherà di dare una conclusione alla ricerca in base a tali risultati, è giusto a questo punto capire cosa si può dedurre, in particolare cercheremo di capire se la tesi iniziale, per cui i sistemi odierni non sono efficienti come sembra in un contesto non sperimentale, sia confutata o meno.

Modello	Immagini usate	Risultati	Osservazioni
Modello senza data augumentation	Immagini originali	Tab. 4.1	I risultati sono mediamente ottimali ma con possibilità di migliorare perciò si opta per la data augumentation.
Modello con data augumentatioon	Immagini originali	Tab. 4.4	I risultati sono mediamente ottimi, come ci si aspettava migliori rispetto a modelli senza data augumentation.
Modello con data augumentatioon	Immagini originali con blur	Tab. 4.3	I risultati sono peggiorati, sopratutto per la classe sandal e coat, perché vengono confuse avendo perso dettagli che le distinguevano da altre classi con forma simile, es. sandal con boot.
Modello con data augumentatioon	Immagini originali con contrast	Tab. 4.4	I risultati sono inferiori di poco a quelli con le immagini originali, per via di una piccola perdita di informzioni.
Modello con data augumentatioon	Immagini originali tagliate	Tab. 4.5	I risultati sono bassi per via di alcune classi che ora non vengono praticamente più considerate per la classificazione, come per esempio le classi del dress e del trouser che con una parte mancante vengono scambiate per il sandal dal sistema.

Modello con data augumentatioon	Immagini originali con rain	Tab. 4.6	I risultati sono bassi, il sistema si concentra motlo sulla silhouette e poco su dettagli che distingue i vari capi e quindi confonde molto t-shirt, shirt, ecc..
Modello con data augumentatioon	Immagini originali con occlusion	Tab. 4.7	I risultati sono bassi, per via di una bassa visibilità dell'immagine e presenta la stessa situazione delle immagini tagliate.
Modello con data augumentatioon	Immagini originali ruotate	Tab. 4.14	I risultati sono disastrosi, le percentuali sono tutte intorno al 10%, la motivazione sta nel fatto che ruoatate di 90° le immagini vengono scambiate tutte per la bag che prima era l'unica ad essere in quella posizione.
Modello con data augumentatioon e immagini a 3 canali	Immagini originali colorare di blu, rosso o verde	Tab. 4.9	I risultati non sono altissimi, la possibilità in questo caso è che nell'allenameto non sono state mostrate le immagini con tutti i colori analizzati ognuna e il sistema si trova impreparato.

Tabella 4.10: Tabella riassuntiva dei risultati della ricerca

CAPITOLO 5

Conclusioni

Questo capitolo illustra quanto è emerso dalla ricerca, mostrando i risultati più significativi. Sono, inoltre, riportate le limitazioni incontrate e alcuni possibili sviluppi futuri. Nel caso si volessero effettuare dei test il codice si può trovare su `https://github.com/AntonioCimino/Reconigtion_clothes_in_image`.

5.1 Interpretazione dei risultati

Nel presente lavoro di ricerca lo scopo era quello di costruire un sistema per il riconoscimento dei capi di abbigliamento all'interno delle immagini, e per raggiungere tale scopo è stata costruita una rete neurale. La rete ha generato dei modelli che sono stati valutati per capire quanto fossero soddisfacente per lo scopo iniziale. La valutazione è stata suddivisa in due fasi, la prima per capire come i modelli funzionassero in maniera generale e poi se i modelli potessero funzionare anche nella vita reale, attraverso immagini che potessero simulare determinate situazioni quotidiane, ovvero la così detta ecological validity [10]. Queste due fasi sperimentali quindi rispondono ai due quesiti che abbiamo stabilito all'inizio del capitolo 3. Da tale analisi è stato possibile notare che nei primi esperimenti il sistema sembrava risultare quasi perfetto nelle sue previsioni, infatti vengono raggiunte delle percentuali che vanno oltre il

92%, cosa che non tutti i sistemi presenti oggi riescono ad ottenere, allo stesso tempo esistono sistemi migliori come in [5]. Il problema maggiore è stato di alcune classi che venivano confuse per via della loro somiglianza (T-shirt, coat, ecc..) altrimenti probabilmente saremo riusciti ad arrivare anche oltre. Nei secondi esperimenti invece sono nati sempre più problemi, dovuti alla confusione che veniva effettuate sulla forma degli abiti. I filtri tendevano a far assomigliare delle classi tra di loro anche se inizialmente totalmente diverse. Nei secondi esperimenti si può essere contenti tutto sommato di aver raggiunto delle percentuali in alcuni casi vicino al 60%, anche se molto bassi, visto come sono state modificate le foto, sopratutto perché i modelli sono stati allenati sul riconoscere immagini in perfette condizioni e nel momento in cui queste immagini perdono le loro condizioni ottimali è intuibile che ci può essere un calo di qualità. Sulla base di tali risultati, è possibile rispondere ai quesiti alla base del lavoro di ricerca e affermare che i modelli generati riescono a riconoscere bene i capi nelle immagini in una situazione generica, mentre in situazioni particolari meno, in particolare in alcune di esse, come nell'esperimento con le immagini ruotate, per nulla. All'inizio si parlava del fatto che la rete costruita in questa ricerca è molto simile a ciò che oggi c'è in letteratura e la valutazione sull'ecology validity fatta ci dice che probabilmente in questo campo andrebbero fatte delle riconsiderazioni, in particolare riguardanti la tipologia di dataset utilizzati e alla affidabilità dei test effettuati.

5.2 Limitazioni e sviluppi futuri

Il sistema ha bisogno, probabilmente, di avere dati più diversificati durante l'allenamento perché la semplicità delle immagini utilizzate ha rappresentato un limite. Probabilmente servono immagini con più rumore per il train. Sembra strano sentir dire che per il train servono dati con imperfezioni, ma il ragionamento è che abituare il sistema a vedere i capi di abbigliamento anche in altre situazioni lo potrebbe preparare a non sbagliare la classificazione quando poi viene testato in una situazione più realistica. Il problema è che le risorse a disposizione con cui è stato sviluppato il sistema hanno fatto fatica quando ci si è provato a spingere più in la, il che ha rappresentato un grosso limite, come nel caso del train con le immagini

colorate, quindi servirebbe molta più potenza computazionale, per effettuare degli allenamenti più consistenti. Un'altra cosa che si potrebbe fare è uno step successivo dal punto di vista sperimentale, riportando il sistema in una situazione reale e non in una che simula la realtà come è stato fatto, perché comunque potrebbe aiutare a capire come questo si comporta con altri elementi nelle immagini, visto che nelle immagini utilizzate per questo studio c'era unicamente il vestito.

Bibliografia

[1] "Nozioni di base della cnn nel deep learning," 2022. [Online]. Available: https://zephyrnet.com/it/basics-of-cnn-in-deep-learning/ (Citato alle pagine iii e 7)

[2] L. Bossard, M. Dantone, C. Leistner, C. Wengert, T. Quack, and L. V. Gool, "Apparel classi

cation with style," 2012. (Citato alle pagine iii, 10, 11 e 20)

[3] G. K. V and S. K., "Fashion-mnist classification based on hog feature descriptor using svm," 2019. (Citato alle pagine iii, 11, 12 e 20)

[4] S. Bhatnagar, D. Ghosal, and M. H. Kolekar, "Classification of fashion article images using convolutional neural networks," 2017. (Citato alle pagine iii, 13, 14 e 20)

[5] M. Kayed, A. Anter, and H. Mohamed, "Classification of garments from fashion mnist dataset using cnn lenet-5 architecture," 2020. (Citato alle pagine iii, 13, 14, 15 e 47)

[6] N. Bertagnolli, "Dropout is drop-dead easy to implement," 2022. (Citato alle pagine iii e 19)

[7] J. Jeong, "The most intuitive and easiest guide for convolutional neural network," 2019. (Citato alle pagine iii e 20)

[8] "Mlp for f-mnist classification," 2022. [Online]. Available: https://www.kaggle.com/code/nikolaosjp/mlp-for-f-mnist-classification (Citato alle pagine iii e 21)

[9] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: a novel image dataset for benchmarkingmachine learning algorithms," 2017. (Citato alle pagine v, 9, 10 e 12)

[10] "What is ecological validity?" 2022. [Online]. Available: https://www.scribbr.com/methodology/ecological-validity/ (Citato alle pagine 2 e 46)

[11] N. Dey, G. S. Mishra, S. Chakraborty, and J. Kar, "A survey of image classification methods and techniques," 2014. (Citato alle pagine 4 e 5)

[12] K. O'Shea and R. Nash, "An introduction to convolutional neural networks," 2015. (Citato alle pagine 7 e 8)

[13] U. M. D. Ciresan, L. M. Gambardella, and J. Schmidhuber, ""convolutional neural network committees for handwritten character classification," 2011. (Citato a pagina 8)

[14] D. Ciresan, U. Meier, J. Masci, L. M. Gambardella, and J. Schmidhuber, "Flexible, high performance convolutional neural networks for image classification," 2011. (Citato a pagina 8)

[15] P. Simard, D. Steinkraus, and J. Platt, "Best practices for convolutional neural networks applied to visual document analysis," 2003. (Citato a pagina 8)

[16] L. Deng and D. Yu, "Deep convex network: A scalable architecture for speech pattern classification," 2011. (Citato a pagina 9)

[17] A. F. Agarap, "Deep learning using rectified linear units (relu)," 2019. (Citato a pagina 18)

[18] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," 2014. (Citato a pagina 19)

[19] "Fashion mnist." [Online]. Available: https://www.kaggle.com/datasets/zalando-research/fashionmnist (Citato a pagina 20)

[20] A. Hasnat, M. Rubaiyat, Y. Qin, and H. Alemzadeh, "Experimental resilience assessment of an open-source driving agent," 2018. (Citato alle pagine 25 e 26)

Ringraziamenti

A conclusione di questo elaborato, desidero menzionare tutte le persone, senza le quali questo lavoro di tesi non esisterebbe nemmeno.

Ringrazio il mio relatore Fabio Palomba, che in questi quattro mesi di lavoro, ha saputo guidarmi, con suggerimenti pratici, nelle ricerche e nella stesura dell'elaborato, e insieme a lui anche i dottorandi del SeSa Lab.

Ringrazio di cuore i miei genitori. Grazie per avermi sempre sostenuto e per avermi permesso di portare a termine gli studi universitari. Insieme ai miei genitori vorrei ringraziare anche tutto il resto della mia famiglia, in particolare mia nonna che è venuta a mancare questo anno.

Un ringraziamento particolare va al mio collega Vincenzo De Martino che ha contribuito ad arrivare a questo momento, visto i tanti esami sostenuti insieme, e anche il mio collega Alfonso Golino con cui ho condiviso la totale esperienza universitaria.

Un ringraziamento a tutti i miei amici di Cellole e ai miei vecchi compagni di superiori con cui condivido ancora una grande amicizia. Vorrei ringraziare anche la mia ragazza che ha sopportato, almeno per metà magistrale, le mie ansie continue. Ringrazio anche i ragazzi di lol che hanno condiviso con me gli anni universitari.