UNIVERSITÀ DEGLI STUDI DI SALERNO

"Kinship Recognition"

Progetto di Fondamenti di Visione Artificiale e Biometria

Prof. Michele Nappi

Tutor Carmen Bisogni, Fabio Narducci

Partecipanti Luigi Di Palma, Domenico Nappi, Federica Ungherese

1. Abstract

Il problema della **kinship recognition** consiste nell'individuare una relazione di parentela o non parentela confrontando i volti di due individui. Per risolvere questo problema di tipo binario si utilizzano in ambito informatico le reti siamesi, che lavorano confrontando due immagini per volta.

Nell'ambito della kinship recognition rientra anche il riconoscimento di uno specifico tipo di parentela, quindi un problema non binario.

In questo progetto abbiamo esplorato entrambi gli aspetti della kinship recognition: prima abbiamo utilizzato una rete siamese sul dataset Kaggle per discriminare i parenti dai non parenti, e successivamente, effettuando il transfer learning e utilizzando un diverso dataset, abbiamo sfruttato il modello per classificare parentele del tipo genitore-figlio e distinguerle dalle non parentele.

Di seguito illustriamo nel dettaglio il problema, la rete e i dataset utilizzati, gli esperimenti effettuati sui modelli al fine di ottenere una migliore accuratezza del modello e i risultati ottenuti.

2. Descrizione del problema

Gli umani, come molti animali e anche piante, posseggono dei meccanismi che consentono loro di riconoscere un legame di parentela, o non parentela, tra individui. Tale abilità prende il nome di kinship recognition.

I meccanismi di riconoscimento della parentela tra due individui si fondano su una combinazione di indizi contestuali (età del soggetto, locazione spaziale) e fenotipici (odore, aspetto fisico). Per queste ragioni, uno dei fenotipi più significativi ai fini del riconoscimento della parentela tra umani è il **volto**. [1]

Mentre negli esseri umani si tratta perlopiù un processo inconscio, recenti tecniche di machine learning, poste davanti al problema della kinship recognition, si sono rivelate in grado di ottenere alte percentuali di accuratezza, effettuando un confronto delle features estratte dai volti dei soggetti considerati.

Oltre al riconoscimento della parentela, un altro punto importante della kinship recognition consiste nel riconoscimento dello specifico **tipo di parentela** che intercorre tra gli individui, che consiste quindi in un problema non binario.

3. Reti siamesi

Una **rete siamese** è una classe di architetture di reti neurali contenente due o più identiche sottoreti, che possiedono la stessa configurazione, con gli stessi parametri e pesi. [2]

Tra i principali utilizzi di queste reti rientrano il riconoscimento della scrittura, la detection automatica dei volti e la face recognition.

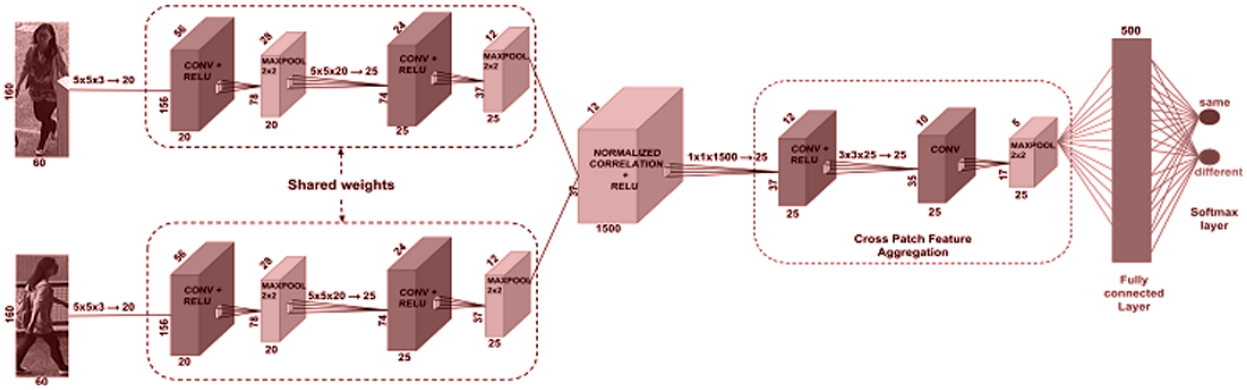


Figura 1: struttura di una rete siamese

Vista la loro struttura, le reti siamesi si prestano bene alla kinship recognition poiché accettano due immagini in input invece che una sola, ragion per cui possono confrontare le caratteristiche dei volti di due individui per determinarne la parentela.

4. "A very simple siamese network"

L'implementazione di partenza della rete siamese utilizzata per questo progetto [3] è stata ideata nell'ambito della competizione "Northeastern SMILE Lab - Recognizing Faces in the Wild", avviata da Kaggle nel 2019 [4] e utilizza le librerie di PyTorch.

4.1 Il dataset: FIW

Per la competizione Kaggle, il dataset è stato fornito da Families In the Wild (FIW), il più grande database per la kinship recognition automatica. Le immagini del dataset provengono da immagini pubblicamente disponibili di personaggi famosi.

Il dataset organizza le immagini in quattro classi di relazioni: **FD** (father-daughter), **FS** (father-son), **MD** (mother-daughter) e **MS** (mother-son).

I dati di training sono organizzati in cartelle, che indicano l'appartenenza a una data famiglia e sono denominate nel formato Fxxxx, e sottocartelle, che indicano le immagini relative a uno specifico individuo e sono denominate nel formato MIDx.

I dati di validation sono organizzati allo stesso modo e provengono unicamente dalla cartella F09xx.

Le coppie di immagini predisposte al training e alla validation sono indicate all'interno di *train_relationship.csv*, mentre le immagini di testing vengono direttamente prelevate dall'apposita cartella *test*.

4.2 Risultati

Insieme ai risultati ottenuti, di seguito sono riportati anche i parametri utilizzati.

epoche	batch size	learning rate	Accuracy training	Accuracy testing
50	16	0.00001	64%	63.3%

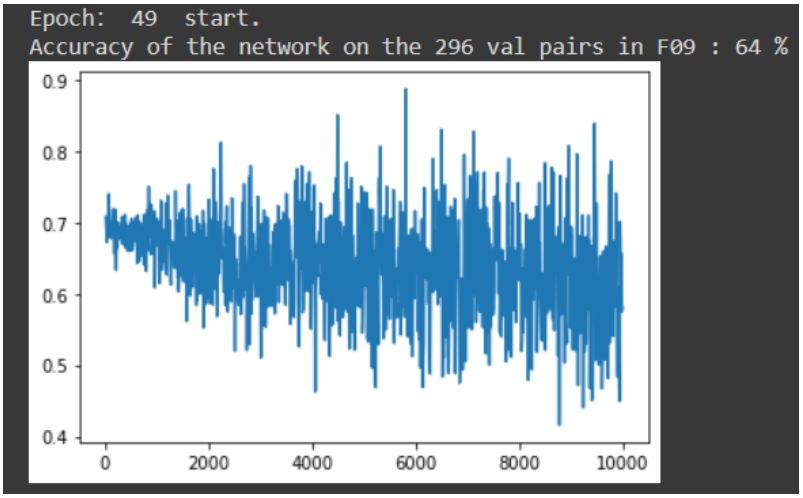


Figura 2: Grafico della loss al termine dell'esecuzione delle epoche

5. Il transfer learning

Il **transfer learning** consiste nello sfruttare una conoscenza, acquisita risolvendo un problema, applicandola a un secondo problema diverso dal primo ma ad esso correlato; in questo modo si adatta un'intelligenza artificiale a un task diverso da quello per il quale era stata addestrata inizialmente.

Uno dei principali **vantaggi** di questo approccio consiste nel fatto di non dover riaddestrare la rete da capo per il nuovo problema, oltre alla possibilità di utilizzare meno dati per addestrare il modello e quindi incorrere in un minor costo computazionale.

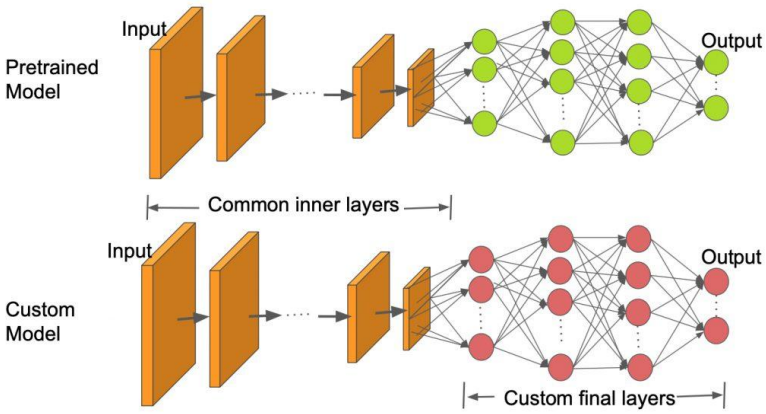


Figura 3: Transfer learning

In ambito image processing, il transfer learning può essere realizzato mediante varie tecniche; tra queste, l'implementazione realizzata per questo progetto sfrutta il modello iniziale come **estrattore** di caratteristiche.

Questa tecnica consiste nell'utilizzare la rete pre-addestrata senza l'ultimo layer pienamente connesso, di modo da estrarre le caratteristiche dell'immagine da esaminare e darle in input alla nuova rete.

6. La nostra soluzione

Facendo uso del modello iniziale, pre-addestrato nel problema binario che consiste nel riconoscimento di una relazione di parentela o non parentela tra due individui, abbiamo realizzato il transfer learning utilizzando il modello di partenza come estrattore di caratteristiche. Abbiamo sostituito l'ultimo layer pienamente connesso così da ottenere un classificatore a cinque classi anziché due.

6.1 Il dataset: KinFace-II

Il dataset si compone di immagini collezionate in internet che includono volti di personaggi pubblici e dei loro genitori o figli. I volti sono stati catturati in ambienti non controllati, senza restrizioni in termini di posa, illuminazione, sfondo, espressione, età, etnia o parziali occlusioni. [5] Di seguito le classi del dataset.

Classe	Nome	Label
Padre-figlia	FD	0
Padre-figlio	FS	1
Madre-figlia	MD	2
Madre-figlio	MS	3
Nessuna relazione	NR	4

Il dataset è **bilanciato** per le quattro classi con 250 coppie di immagini ciascuna, ma presenta un forte sbilanciamento per la classe della non relazione, per questo motivo abbiamo provveduto a un ribilanciamento dell'ultima classe, che in maniera analoga è ora rappresentata da 250 coppie di immagini.

Per ogni classe, il dataset espone un file *.mat* con le seguenti informazioni:

- quantile: valore numerico da 1 a 5;
- parentela/non parentela: valore numerico che indica una non parentela (0) o la presenza di parentela (1);
- prima immagine;
- seconda immagine.

Nella suddivisione dei dati per training, validation e testing, abbiamo utilizzato l'**60%** delle immagini per il training (quantili da 1 a 3), il **20%** per la validation (quantile 4) e il **20%** per il testing (quantile 5). In totale vengono utilizzate 739 coppie di immagini per la cartella di training e 248 sia per validation che per testing.

Per poter lavorare in maniera analoga al precedente dataset della competizione di Kaggle si sono rese necessarie delle modifiche al dataset corrente.

Per ricavare le coppie di immagini per la classe di non parentela, che il dataset non fornisce direttamente, abbiamo separato in una prima fase le coppie di immagini che nella colonna di parentela/non parentela del file .mat presentavano il valore 0 (non parentela) dal valore 1 (parentela). Fatto ciò, abbiamo scartato le coppie in sovrannumero dalla classe di non parentela, avendo cura di mantenere bilanciato il numero di coppie per quantile, poiché i componenti della classe di non parentela erano in numero superiore rispetto agli altri.

Abbiamo assegnato la label indicativa della classe a ciascuna coppia: per il csv composto unicamente da coppie della classe di non parentela abbiamo assegnato label 4, mentre scorrendo i csv delle altre classi abbiamo assegnato progressivamente una label in base al nome delle coppie di immagini (se i primi caratteri del nome dell'immagine contengono "fd", allora si tratta della classe padre-figlia e quindi della label 0).

Una volta completata questa operazione, abbiamo fuso i csv così ottenuti e abbiamo inserito le coppie, in base al quantile di appartenenza, nei file denominati train, val e test. In base al contenuto di questi file, abbiamo smistato le immagini del dataset KinFace-II in tre differenti cartelle, a loro volta denominate train, val e test.

6.2 Transfer learning su 4 classi

Il problema sul quale verte questo progetto riguarda una classificazione a cinque classi, tuttavia può essere utile un confronto con un modello che lavora su quattro classi, e che

quindi non opera sulla classe di non parentela, ma soltanto sulle quattro classi che riguardano una relazione genitore-figlio.

Può essere utile infatti confrontare i due modelli e i risultati che raggiungono, poiché i risultati ottenuti dal modello che lavora su quattro classi possono fornire un benchmark di riferimento per il modello che lavorerà su una classe in più.

6.2.1 Esperimenti e risultati

Nei primi esperimenti abbiamo modificato il modello sovrascrivendo l'ultimo layer pienamente connesso e aggiungendone due, per un totale di cinque layer pienamente connessi. In un ultimo esperimento, invece, abbiamo lasciato intatte le dimensioni dei tre layer originali, modificando solo il numero di classi di output da 2 a 4, notando una uniformità nei risultati, ragion per cui, negli esperimenti sul modello a cinque classi, abbiamo utilizzato solamente tre layer pienamente connessi.

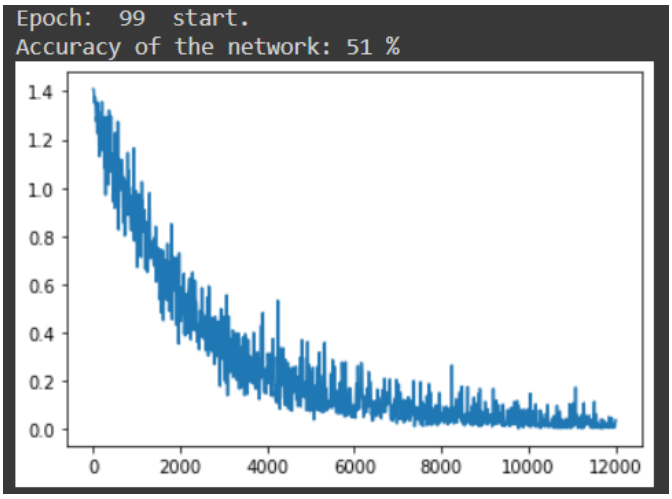
Poiché in alcuni esperimenti il modello raggiungeva l'overfitting, abbiamo diminuito il numero di epoche ed effettuato una data augmentation sui dati di training. In particolare, per la data augmentation PyTorch offre un metodo "transform" che consente di applicare delle trasformazioni alle immagini. In questo modo, abbiamo triplicato i dati di train, utilizzando due diverse trasformazioni: una rotazione e una trasposizione specchiata orizzontalmente delle immagini.

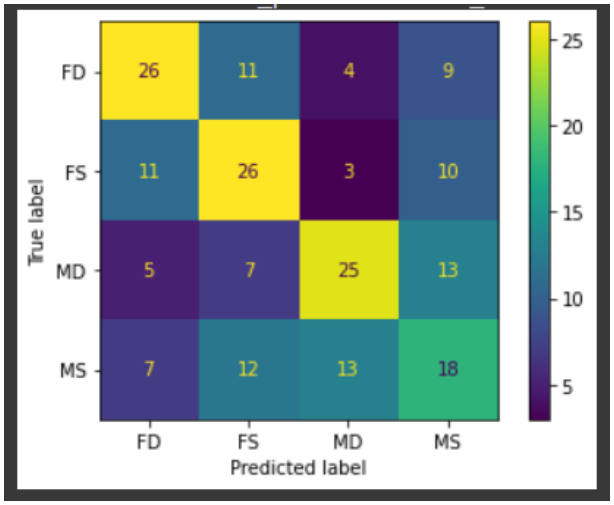
Nella tabella sono presenti maggiori dettagli sugli esperimenti più significativi.

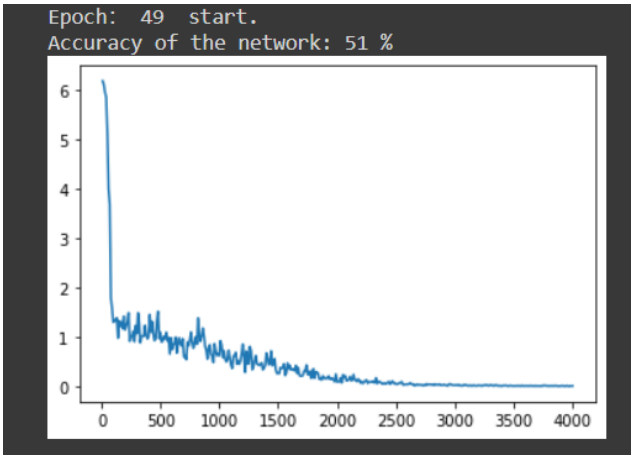
n° esperimento	Layer fully connected	Epoche	Batch size	Learning rate	Accuracy training	Accuracy testing
1	5	50	16	0.00001	51%	49%
2	5	30	16	0.00001	53%	49%
3	5	30	16	0.00001	54%	48%
4	3	100	32	0.00001	51%	51%

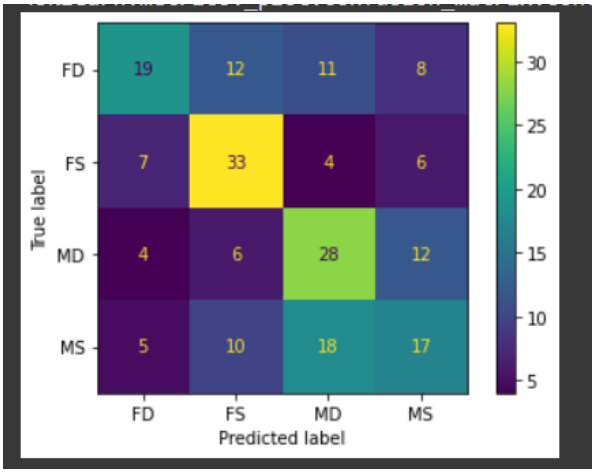
L'esperimento **1** intorno alla 30° epoca raggiunge l'overfitting, perciò nell'esperimento **2** abbiamo diminuito il numero di epoche e nel **3** abbiamo utilizzato la data augmentation.

Nell'esperimento **4**, invece, sono stati utilizzati i parametri che, nella classificazione a 5 classi, hanno dato le migliori percentuali di accuratezza.









Per gli esperimenti **1**, **2** e **3** i risultati relativi alla matrice di confusione sono analoghi alla matrice in *Figura 7:* le classi che vengono riconosciute con maggior precisione sono la classe FS e la classe MD, con una precisione più bassa per la classe FD e serie difficoltà per la classe MS. Ciò può essere dovuto al fatto che il modello è in grado di riconoscere più facilmente una somiglianza tra volti dello stesso sesso, quindi in relazioni padre-figlio e madre-figlia.

Nell'esperimento **4** *(Figura 5)* invece la matrice risulta più equilibrata, pur continuando ad avere qualche difficoltà con la classe MS.

6.3 Transfer learning su 5 classi

Come specificato precedentemente, il transfer learning avviene su cinque classi: quattro di relazione genitore-figlio e una di non relazione. Il modello pre-addestrato sul problema binario di riconoscimento di parentela/non parentela tra due individui viene in questo caso

utilizzato come estrattore di caratteristiche; di quel modello abbiamo sovrascritto l'ultimo layer pienamente connesso.

6.3.1 Esperimenti e risultati

Nei seguenti esperimenti, vista la mancanza di un vero e proprio apporto positivo all'accuratezza del modello utilizzando un maggior numero di layer pienamente connessi, come visto anche a seguito degli esperimenti sul modello a quattro classi, ne abbiamo utilizzati tre.

Gli esperimenti qui riportati fanno uso di data augmentation per i dati di training, svolta in maniera analoga a come illustrato al paragrafo 6.2.1.

n° esperimento	Epoche	Batch size	Learning rate	Momentum	Accuracy training	Accuracy testing
1	30	64	0.00001	0.9	50%	52%
2	75	64	0.00001	0.9	47%	41%
3	200	64	0.000001	1	46%	45%
4	100	64	0.000001	1	45%	43%
5	100	128	0.000001	1	47%	45%
6	100	32	0.00001	0.9	49%	48%

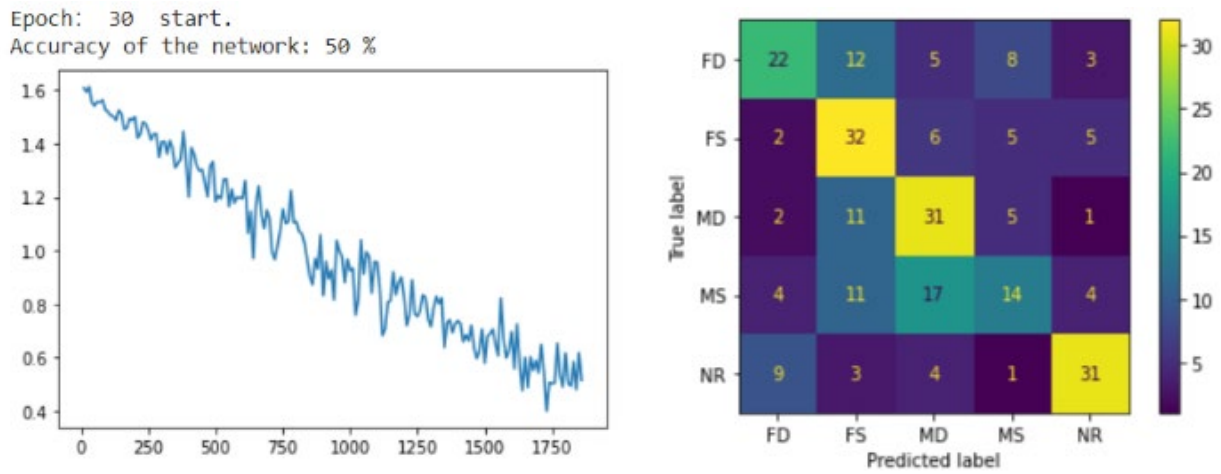


Figura 8: Grafico di loss e matrice di confusione dell'esperimento 1

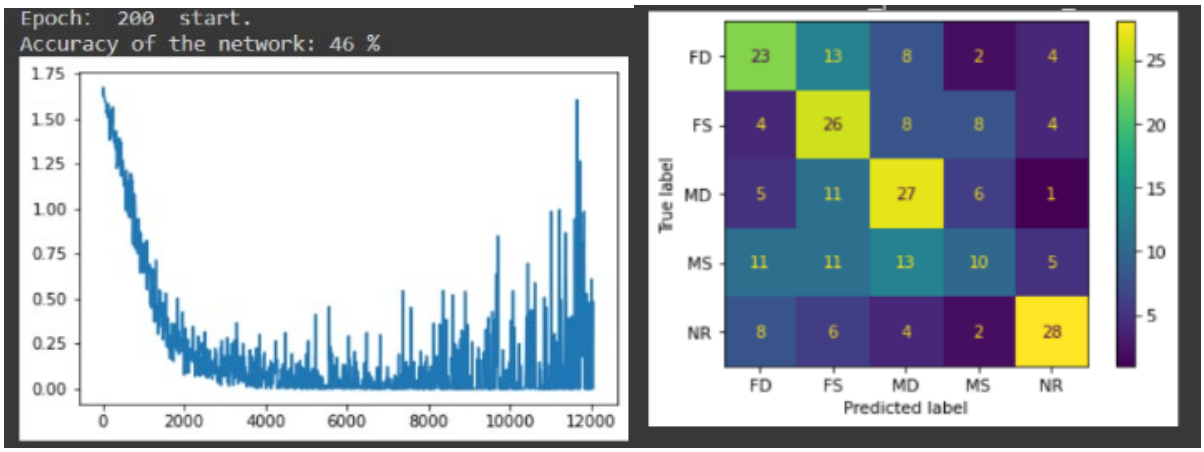


Figura 9: Grafico di loss e matrice di confusione dell'esperimento 3

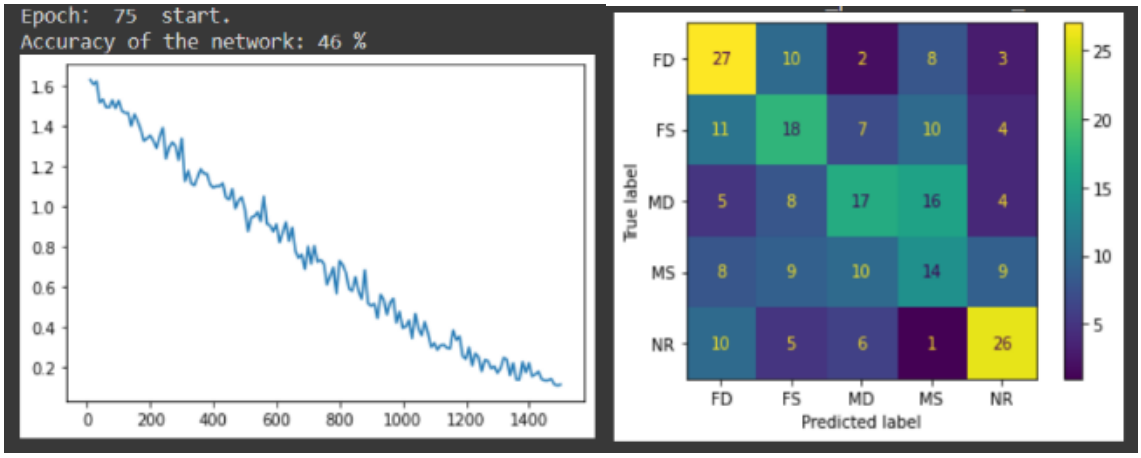


Figura 10: Grafico di loss e matrice di confusione dell'esperimento 2

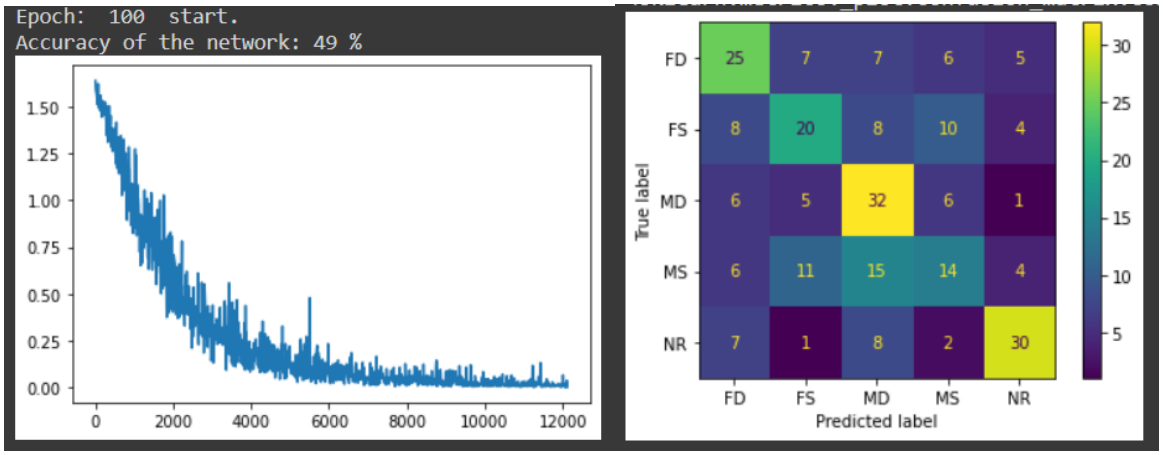


Figura 11: grafico di loss e matrice di confusione dell'esperimento 6 (il modello finale).

A seguito dell'esperimento **1**, vista la maggiore accuratezza nel testing che nel training, abbiamo aumentato il numero di epoche nell'esperimento **2**.

Nell'esperimento **3,** intorno alla 100° epoca la loss subisce un'impennata, per cui nell'esperimento **4** abbiamo diminuito il numero di epoche.

L'esperimento **6** rappresenta la versione finale del modello.

In ogni esperimento il modello ha riconosciuto la classe **NR** con un'ottima precisione, nonostante alcuni esprimenti invece abbiano mostrato delle difficoltà nel riconoscimento delle rimanenti 4 classi. Ciò può essere dovuto al fatto che la rete di partenza era addestrata su un problema binario, dunque è in grado di classificare abbastanza accuratamente una non-parentela rispetto a una parentela.

Come negli esperimenti relativi al transfer learning su 4 classi, anche in tal caso il modello ha classificato con maggior precisione le classi relativi a parenti dello stesso sesso , quindi della classe MD ed FS (esperimenti **1** e **3**), con una controtendenza nell'esperimento **2**, in cui il modello non è stato in grado di fornire buone accuratezze per queste classi ma ha riconosciuto con successo gran parte dei componenti della classe FD.

Per quanto riguarda la classe MS, invece, le difficoltà sono state significative in ogni esperimento svolto: come si può vedere dalle matrici di confusione degli esperimenti **1**, **3** e **6** in *Figura 8, 9* e *11*, il modello ha scambiato la classe MS per MD.

7. "I soliti ignoti"

L'ultima parte del progetto consiste nel **testare** la rete creata ed addestrata precedentemente su un nuovo dataset.

Per creare questo nuovo dataset di immagini sono state prese in considerazione le puntate del programma televisivo "I soliti ignoti", dove l'obbiettivo del gioco, in una prima fase, è quello di abbinare un'identità a un personaggio. Per la seconda, quella che interessa questo progetto, invece entra in gioco un nuovo personaggio, ovvero il "parente misterioso", al quale il concorrente deve abbinare uno dei precedenti ignoti con cui ha un rapporto di parentela.

7.1 Creazione del dataset

Il dataset è stato creato selezionando delle precise puntate, infatti siccome le classi di parentela sono quattro (FD, FS, MD, MS) sono state prese in considerazioni tre puntate per ogni classe.

Sono stati estratti i volti degli otto partecipanti e del parente misterioso utilizzando una funzione di Face Detection [6] per ogni puntata. Per far ciò sono stati individuati i frames del video dove i diversi partecipanti venivano mostrati in primo piano e successivamente passati alla funzione.

Tutte le immagini sono state memorizzate in specifiche cartelle, una per ogni puntata.

Successivamente per ogni puntata è stato creato un csv costituito da tre colonne, dove nella prima colonna viene riportato il nome dell'immagine relativo al parente misterioso ripetuto per tutte le righe, nella seconda invece vengono ripotati i nomi delle immagini di tutti gli altri ignoti e nell'ultima vengono riportate le label che specificano il tipo di parentela tra i diversi confronti.

Per ogni csv nella prima riga sono inserite le coppie per cui vi è effettivamente parentela, mentre nelle successive righe vi sono le coppie che non hanno alcuna parentela.

img1	img2	tipo parentela
image0.JPG	image1.JPG	3
image0.JPG	image2.JPG	4
image0.JPG	image3.JPG	4
image0.JPG	image4.JPG	4
image0.JPG	image5.JPG	4
image0.JPG	image6.JPG	4
image0.JPG	image7.JPG	4
image0.JPG	image8.JPG	4

Figura 12: Struttura del csv per un episodio

7.2 Testing sulle puntate

Una volta superata la fase dedicata alla creazione del dataset "I soliti ignoti" vi è la fase di testing. Infatti, per ogni puntata è stato creato un nuovo csv identico a quello di partenza, ma in cui sono stati sostituiti i valori nella colonna del tipo parentela inserendo il valore che rappresenta la non parentela tra le due foto, che verrà sovrascritto successivamente.

Successivamente le immagini sono state passate al modello e le predizioni fatte per ogni coppia hanno sovrascritto i valori precedenti con le parentele predette dal modello.

Questa operazione è stata iterata per ogni puntata e successivamente è stato calcolata la percentuale di accuratezza per tutte le puntate facendo un semplice calcolo: (NUM_VALORI_CORRETTI / NUM_VALORI_TOTALI) * 100.

7.3 Risultati

I risultati sono molto bassi per via della scarsa accuratezza ottenuta durante la fase di addestramento del modello, infatti l'accuratezza generale del testing per il dataset "I soliti ignoti" è stata del **18%**.

Gli esperimenti mostrano dei buoni risultati solo per la parte relativa alla classe FS, infatti, in tutti e tre gli esperimenti la rete è stata in grado di riconoscere correttamente la parentela fra le due persone realmente imparentate.

Invece per quanto riguarda gli esperimenti relativi alle classi FD ed MD, la rete due volte su tre è stata in grado di riconoscere in maniera corretta la parentela.

Gli esperimenti relativi alla classe MS sono i peggiori: la rete è stata in grado di riconoscere la parentela solo una volta rispetto ai tre esperimenti fatti.

Per quanto riguarda tutte le altre correlazioni per ogni puntata fra il parente misterioso e i restanti concorrenti, la rete avrebbe dovuto dare un risultato di NR, ovvero di non parentela, ma solo in poche occasioni la rete è stata capace di riconoscere questa quinta classe.

Di seguito i risultati sono presentati in tabella.

Classe	Episodio	Parentela riconosciuta	Label assegnata	Non-parentela riconosciuta
FD	14/04/21	No	4 - NR	3/7
FD	16/04/21	Sì	0 – FD	0/7
FD	23/02/21	Sì	0- FD	2/7
FS	05/05/21	Sì	1 – FS	0/7
FS	09/04/21	Sì	1 – FS	0/7
FS	20/04/21	Sì	1 – FS	3/7
MD	10/01/21	Sì	2 - MD	0/7
MD	18/04/21	Sì	2 – MD	0/7
MD	19/04/21	No	0 – FD	0/7
MS	25/04/21	No	0 – FD	1/7
MS	27/04/21	No	0 – FD	0/7
MS	29/03/21	Sì	3 – MS	1/7

In totale, su 96 coppie il modello ne ha riconosciute correttamente 18, contando come successo non solo il riconoscimento del parente misterioso, ma anche il riconoscimento del rapporto di non parentela tra il parente misterioso e gli ignoti.

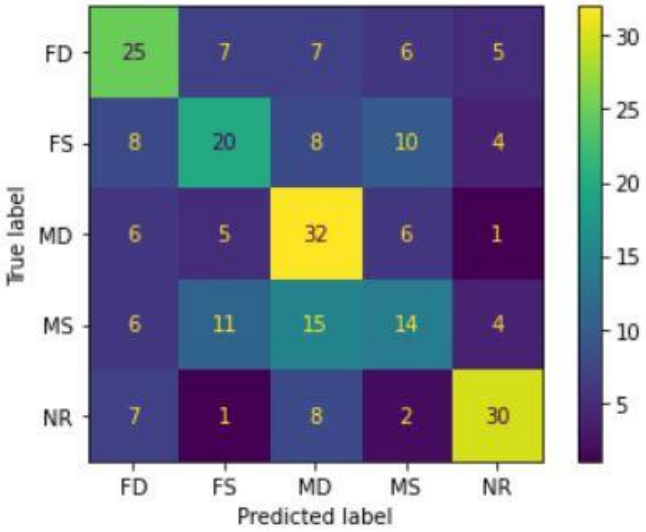
Per quanto riguarda la sola parentela, è stata riconosciuta correttamente in 8 coppie su 12, per un'accuratezza del **66%**. Per le rimanenti coppie a cui è stato assegnato un valore erroneo, solo 1 su 4 è stata scambiata per una non parentela, quindi nel **91%** delle coppie effettivamente imparentate è stato riconosciuto un legame di parentela.

8. Facoltativo

Al fine di comprendere il comportamento della rete neurale, si è deciso di procedere con il metodo della **cascata** che consiste nell'eliminare di volta in volta la classe che ha ottenuto il risultato maggiore in termini di correttezza della predizione fino ad ottenere un problema binario.

8.1 Risultati

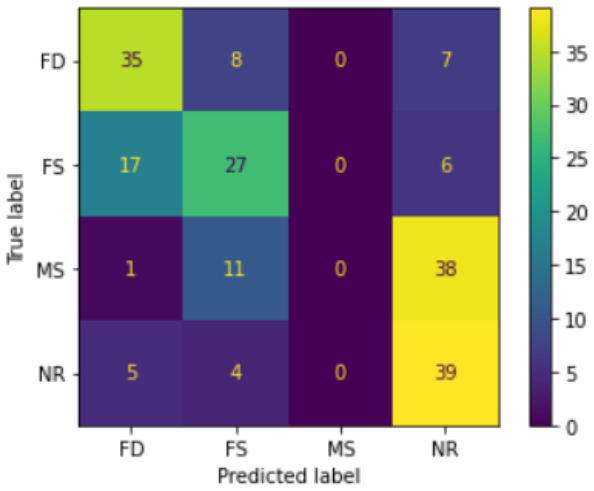
Prendendo come riferimento la **matrice di confusione** ottenuta al termine della fase di testing, la prima classe da eliminare è risultata essere MD (Mother-Daughter).



Primo step:

Eliminando dal dataset di partenza le immagini relative alla classe MD (label 2), abbiamo modificato l'ultimo layer della rete in modo da diminuire l'output finale a 4 classi ed effettuato nuovamente il training avendo cura di modificare le label relative alle classi MS (Mother-Son) e NR (Not-Related) in modo da non incorrere in un errore "**RuntimeError: CUDA error: device-side assert triggered**" [7], che si presentava nel caso in cui le labels relative alle classi non fossero state correttamente istanziate.

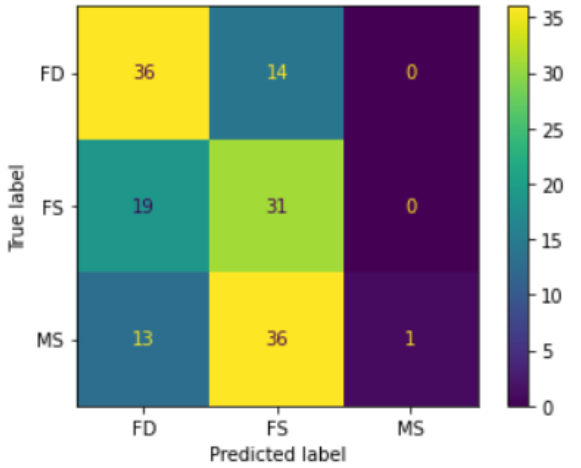
Al termine della fase di testing, abbiamo ottenuto una **matrice di confusione** che ha evidenziato come la rete abbia migliorato la sua capacità nel riconoscimento di tutte le classi fatta eccezione per la classe MS che, al contrario, ha subìto un netto peggioramento probabilmente dovuto all'eliminazione delle immagini all'interno del Dataset della classe MD che ha portato la rete ad avere difficoltà nel riconoscimento di una parentela in cui il genitore analizzato fosse la Madre.



Secondo step:

Successivamente si è proceduto all'eliminazione dal dataset delle immagini relative alla classe NR, abbiamo modificato l'ultimo layer della rete in modo da diminuire l'output finale a 3 classi ed effettuato nuovamente il training. In questo caso non è stato necessario modificare le labels delle classi in quanto NR era associata alla label 3 che è stata automaticamente eliminata quando l'output finale è stato impostato a 3 (labels restanti: 0, 1 e 2).

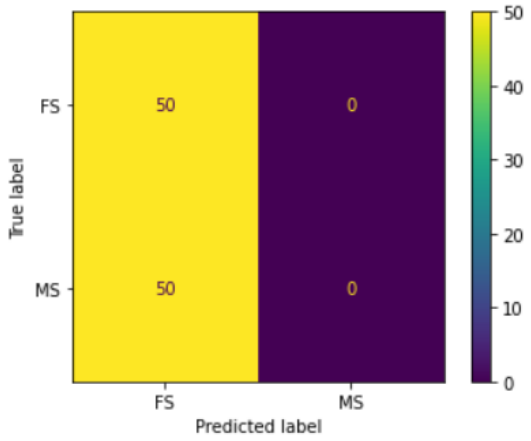
Anche questa volta al termine della fase di testing la **matrice di confusione** ottenuta ha evidenziato come la rete abbia migliorato la sua capacità nel riconoscimento di tutte le classi compresa la classe MS, che è passata da 0 a 1 parentela correttamente riscontrata.



Terzo step:

Infine, abbiamo proceduto ad eliminare dal dataset le immagini relative alla classe FD (Father-Daughter), adattando le labels relative alle restanti classi in quanto la classe FD era associata alla label 0, a modificare il layer finale della rete portando l'output a 2 (binario) e ad effettuare nuovamente il training.

Il **risultato finale** ottenuto a seguito della fase di testing ed evidenziato dalla **matrice di confusione** ottenuta è quello che la rete risulta essere eccellente nel riconoscimento di una parentela appartenente alla classe FD ma del tutto incapace di riconoscere una parentela appartenente alla classe MS.



Accuratezza finale:

Per il calcolo dell'accuratezza finale al termine della cascata, riportiamo di seguito i risultati:

- 1° step: 97 errori;
- 2° step: 82 errori;
- 3° step: 50 errori.

Gli errori totali delle sottoreti sono dunque 229. Il numero totale di immagini fornite alla rete da cui si è partiti per realizzare la cascata, cioè il classificatore a 5 classi, è 248.

La quantità di **predizioni corrette** è quindi 248-229=19 che corrispondono al 7,66%.

9. Conclusioni

In conclusione, la rete ottenuta non ha dato risultati soddisfacenti nella risoluzione del problema di classificazione della parentela.

Di seguito una sintesi dei risultati ottenuti nel corso del progetto.

	Accuratezza training	Accuratezza testing
Problema a 2 classi	64%	63.3%
Problema a 4 classi	54%	48%
Problema a 5 classi	49%	48%
Dataset "I soliti Ignoti"	x	18%

Cascata – eliminazione 1 classe (MD)	51%	51%
Cascata – eliminazione 2 classi (MD, NR)	46%	45%
Cascata – eliminazione 3 classi (MD, NR, FD)	50%	50%

Come possibili sviluppi futuri, viste le basse accuratezze ottenute dal training del modello, un'eventuale modifica futura potrebbe consistere nell'utilizzare un'altra rete per lo stesso problema.

Per quanto riguarda la cascata, di cui noi abbiamo gettato le basi, negli sviluppi futuri può rientrare la fase di fusione finale degli score ottenuti, e quindi andare a mettere insieme tutta la cascata.

Per questo progetto abbiamo utilizzato le risorse di Google Colab, quindi probabilmente con un hardware più prestante, in locale, sarebbe possibile ottenere risultati migliori.

10. Bibliografia

[1] L. DeBruine, «psychological science,» 30 Settembre 2014. [Online]. Available: https://www.psychologicalscience.org/observer/a-sense-of-family.

[2] S. B. J, «towards data science,» 2 Settembre 2020. [Online]. Available: https://towardsdatascience.com/a-friendly-introduction-to-siamese-networks-85ab17522942.

[3] jiangstein, «A very simple siamese network,» 2019. [Online]. Available: https://www.kaggle.com/jiangstein/a-very-simple-siamese-network-in-pytorch.

[4] «Kaggle,» 2019. [Online]. Available: https://www.kaggle.com/c/recognizing-faces-in-the-wild/overview/timeline.

[5] «KinFaceW,» [Online]. Available: http://www.kinfacew.com/datasets.html.

[6] «GeeksforGeeks,» 13 Gennaio 2021. [Online]. Available: https://www.geeksforgeeks.org/cropping-faces-from-images-using-opencv-python/.

[7] «StackOverflow,» 2019. [Online]. Available: https://stackoverflow.com/questions/58242415/how-to-fix-runtimeerror-cuda-error-device-side-assert-triggered-in-pytorch.