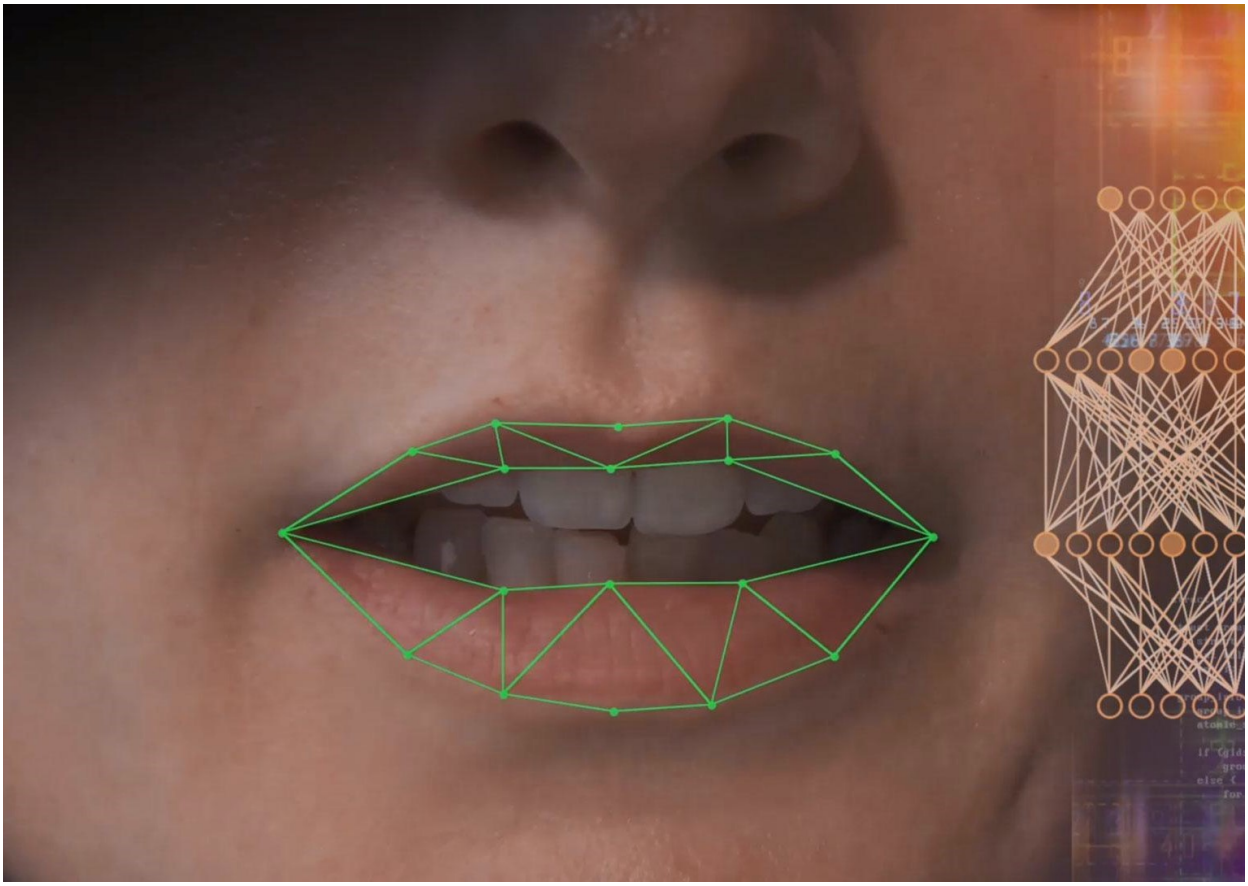
LipRecognize

Fondamenti di Visione Artificiale e Biometria



Componenti del gruppo:
Autore Eduardo - 0522501549
Ianuzziello Francesco Pio - 0522501151
Macaro Alessandro - 0522501459

A.A. 2022/2023

Sommario

Introduzione 2

Requisiti tecnici 2

Preprocessing dei frame 3

Selezione della rete neurale 5

Architettura della rete neurale 6

Test iniziali 8

Dataset Babele 10

Dataset Vid-Timit 12

Considerazioni 15

Sviluppi futuri 18

Introduzione

Le labbra rappresentano un interessante tratto biometrico nel campo del riconoscimento, perché ogni individuo possiede labbra uniche, caratterizzate da una forma geometrica e da un intensità specifica. Le labbra sono oggetto di studio per diverse applicazioni, come il riconoscimento facciale e l'identificazione personale. Analizzandole è possibile individuare feature distintive utili per identificare una persona in modo accurato e sicuro.

L'obiettivo del nostro progetto è stato riconoscere l'identità di un soggetto parlante soltanto mediante la zona labiale, senza disporre della componente audio.

Requisiti tecnici

Per lo sviluppo degli algoritmi impiegati nel nostro lavoro viene utilizzato il linguaggio Python.
Le principali librerie utilizzate, con relativa versione, sono le seguenti:

Nome libreria	Versione	Utilizzo
keras	2.12	Libreria di Deep Learning open source scritta in Python che fornisce un'API di alto livello intuitiva per la costruzione e la formazione di reti neurali profonde e per creare potenti modelli di apprendimento automatico
opencv	4.5.5.62	Libreria open-source per il machine learning e la computer vision che fornisce un'ampia gamma di funzioni per l'elaborazione di immagini e video
matplotlib	3.7.1	Libreria per la creazione di grafici
numpy	1.23.5	Libreria open source che aggiunge supporto a grandi matrici e array multidimensionali insieme a una vasta collezione di funzioni matematiche di alto livello per poter operare efficientemente su queste strutture dati
scikit-learn	1.2.2	Libreria open-source per l'apprendimento automatico che offre una vasta gamma di algoritmi e strumenti per la classificazione, la regressione, il clustering e la riduzione delle dimensioni. La libreria fornisce anche funzionalità per la valutazione dei modelli, la selezione delle caratteristiche e la gestione dei dati
tensorflow	2.12	Libreria open-source di apprendimento automatico sviluppata da Google. È ampiamente utilizzata per la creazione e l'addestramento di modelli di apprendimento automatico, in particolare reti neurali profonde
pandas	2.0.2	Libreria open-source per l'analisi e la manipolazione dei dati che fornisce funzioni per la lettura e la scrittura di dati da e verso una serie di formati di file, tra cui CSV

Preprocessing dei frame

Il Dataset di partenza, chiamato Babele, è costituito da 1024 video della durata di 10 secondi ciascuno (per un totale di 256 soggetti). Ogni video contiene la ROI (Region of Interest) della zona labiale.
Prima di procedere con lo sviluppo della rete neurale, è stata dedicata una particolare attenzione al preprocessing dei frame. Questo lavoro di analisi e miglioramento è stato essenziale, in quanto la qualità è bassa e spesso la zona labiale non è ben identificabile.
Nella directory del progetto è presente un file chiamato *Preprocessing.py* che gestisce la fase di preprocessing dei frame. Per migliorare la scalabilità del codice, si è deciso di utilizzare delle variabili di supporto per memorizzare i path della directory dei video originali e delle directory dei frame preprocessati. Questo approccio consente una maggiore flessibilità nel caso in cui sia necessario modificare le directory di input e di output senza dover modificare il codice sorgente in più punti.
I metodi principali sono:

- *preprocess_video*, prende in input il percorso del video (input_video), l'indice del video (index_video), la directory di output per i frame preprocessati (directoryOutputNormalized) e un parametro booleano (preprocessing_bool) per abilitare o disabilitare il preprocessing.
  La funzione crea una cartella specifica per il video all'interno della directory di output, apre il video e itera su ogni frame del video. Se preprocessing_bool è impostato su True, applica una funzione di preprocessing chiamata preprocessing_meanShift al frame corrente.
  Successivamente, salva l'immagine preprocessata nella cartella creata in precedenza. Infine, rilascia le risorse del video.
- *preprocessing_meanShift*, implementa un algoritmo di preprocessing chiamato Mean Shift.
  Prende in input un frame dell'immagine e svolge le seguenti operazioni:
  - Applica un filtro di sfocatura gaussiano per ridurre il rumore e converte l'immagine sfocata in HSV;
  - Applica l'equalizzazione dell'istogramma al canale di luminanza (Value), utile per migliorare il contrasto;
  - Applica il filtro Mean Shift per la segmentazione dei colori;
  - Converte l'immagine risultante in scala di grigi e applica una soglia adattiva con threshold gaussiana. Dopodichè applica la maschera ottenuta all'immagine originale utilizzando l'operazione bitwise;
  - Restituisce l'immagine preprocessata.
- preprocessing_otsu, applica il metodo di Otsu per ottenere una sogliatura automatica dell'istogramma del singolo frame. Svolge le seguenti operazioni:
  - Converte il frame in scala di grigi e applica l'equalizzazione dell'istogramma per migliorare il contrasto;

- Applica un filtro di sfocatura gaussiano al frame per ridurre il rumore;
- Utilizza l'algoritmo di Otsu per calcolare due soglie di segmentazione. Queste soglie separano le regioni dell'immagine in base ai livelli di intensità dei pixel. Dopodiché combina le soglie ottenute con il frame in scala di grigi.
- Normalizza l'immagine risultante tra i valori 0 e 255 e restituisce l'immagine preprocessata.

Questa procedura ci consente di ottenere un dataset normalizzato e di migliorare la qualità dei frame rispetto al dataset originale. Di seguito, un esempio di frame originale e preprocessato.

Fig.1: Confronto tra un frame originale e lo stesso frame dopo il preprocessing

Selezione della rete neurale

La rete neurale selezionata per lo sviluppo del progetto è una rete neurale convoluzionale (CNN) e una rete neurale ricorrente (RNN) come una long short-term memory (LSTM). La combinazione di queste due reti, CNN-LSTM, si è dimostrata migliore rispetto ad altre reti neurali conosciute. I principali vantaggi nell'utilizzo di una rete CNN-LSTM sono:

- Una rete di tipo CNN è ideale per l'estrazione di feature significative dalle immagini, riconoscendo pattern come bordi, texture e forme. Le CNN possono apprendere autonomamente queste feature mediante l'uso di livelli di convoluzione e aggregazione. Le feature estratte sono quindi passate alla LSTM, che può analizzarle sequenzialmente, in modo da catturare relazioni a lungo termine all'interno delle immagini.
- La combinazione di CNN e LSTM consente di unire sia le feature estratte dalle immagini che il contesto sequenziale, consentendo un'elaborazione più efficace di dati complessi come le immagini.
- Una rete CNN-LSTM apprende in modo end-to-end, il che significa che può essere addestrata direttamente dai dati di input senza la necessità di estrarre manualmente le feature. Ciò semplifica il processo di progettazione del modello e riduce la dipendenza da estrattori di feature esterni.
- La combinazione di CNN-LSTM può aiutare a migliorare la capacità di generalizzazione del modello. La CNN può apprendere feature di base condivise da diverse immagini, consentendo una maggiore capacità di riconoscimento di oggetti e modelli simili.

Architettura della rete neurale

Inizialmente viene suddiviso il video della zona labiale in frame, che verranno normalizzati e successivamente utilizzati come input. Ogni frame passa attraverso l'architettura CNN già addestrata. L'output della CNN passa attraverso il Dense Layer per essere dato come input allo strato LSTM. L'output dello strato LSTM diventa l'input dello strato denso successivo. Infine, riceve l'etichetta in uscita dalla funzione Softmax. La funzione Softmax è utile per ottenere una distribuzione di probabilità tra le diverse classi, consentendo di effettuare una classificazione finale basata sulla classe con la probabilità più alta.

Analizzando i frame ottenuti dai video ci si è resi conto che quando viene estratta la forma delle labbra da un frame si ottengono informazioni che sono molto simili a quelle che si otterrebbero analizzando l'aspetto generale dell'intera immagine. L'aspetto generale dell'intera immagine comprende l'insieme delle feature visive globali presenti nell'immagine nel suo complesso. Queste feature possono includere il colore, i contorni, le forme generali e altri elementi visivi che ci permettono di comprendere l'aspetto complessivo dell'immagine senza focalizzarci su dettagli specifici.
Questa considerazione ha permesso di utilizzare il metodo dell'apprendimento per trasferimento utilizzando un modello precedentemente addestrato su ImageNet. Ci si basa sull'idea che il modello pre-addestrato su un ampio set di dati come ImageNet abbia imparato caratteristiche generali dell'immagine, inclusa la forma delle labbra, che possono essere riutilizzate per il nostro contesto.

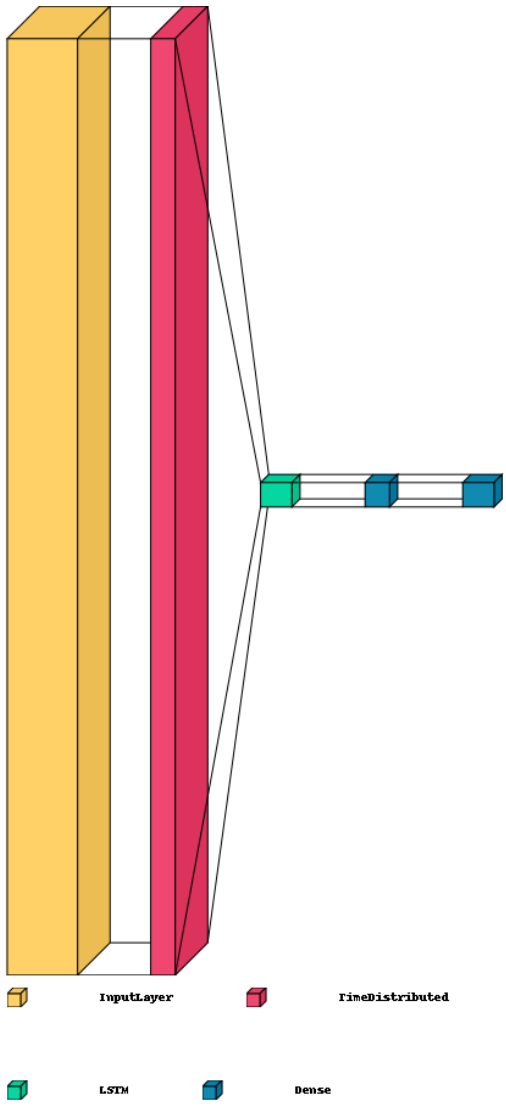
Di conseguenza, è stato deciso di addestrare solo lo strato denso della CNN presente nel modello. Considerando che la scelta si basa sulla somiglianza delle caratteristiche, concentrandoci solo sullo strato denso, che rappresenta l'output finale della rete, si ottiene una rappresentazione ottimizzata della forma delle labbra.

È stata scelta la libreria MobileNet come modello di apprendimento per trasferimento.
Per la preparazione dei dati, sono state eseguite varie operazioni. Innanzitutto, è stato calcolato il numero di istanze per ciascuna etichetta presente nel set di dati. Successivamente, le immagini della forma delle labbra sono state impilate lungo la dimensione del tempo (TimeStep). Utilizzando la libreria Keras, l'elenco delle immagini è stato convertito in un array NumPy. Infine, i dati sono stati mescolati e raggruppati per il processo di addestramento. Il set di dati finale ha una forma di (586, 20, 128, 128, 3), in cui 586 rappresenta il numero di istanze nel set di dati, 20 indica il numero di TimeSteps (frame), 128x128 rappresenta la dimensione delle immagini e 3 indica il numero di canali (immagini a colori).

L'architettura del modello prevede l'utilizzo di MobileNet insieme a uno strato denso. Nello strato denso è stato applicato il dropout, una tecnica efficace per evitare l'overfitting. Inoltre, è stato integrato anche l'early stop, che consente di interrompere l'addestramento del modello se non si osservano miglioramenti significativi nelle metriche di valutazione. È stato inoltre utilizzato un kernel

regolarizzatore L2 per limitare la complessità del modello e ridurre l'overfitting.

L'output di questa parte del modello viene quindi utilizzato come input per uno strato LSTM. L'output dell'LSTM viene poi passato attraverso uno strato denso con un ulteriore dropout. Infine, viene prodotta l'etichetta di classificazione.



Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 20, 192, 192, 3)]	0
time_distributed (TimeDistributed)	(None, 20, 192)	41175424
lstm (LSTM)	(None, 256)	459776
dense_2 (Dense)	(None, 192)	49344
dense_3 (Dense)	(None, 256)	49408

Total params: 41,733,952
Trainable params: 38,505,088
Non-trainable params: 3,228,864

Fig.3: Architettura del modello

Test iniziali

Il primo approccio al problema è stato fornire alla rete neurale scelta l'intero dataset non preprocessato e valutare i risultati. Per configurare il modello di apprendimento, è stato utilizzato l'ottimizzatore Adam, mentre per quanto riguarda la funzione di perdita, è stata utilizzata la categorical_crossentropy, che è una metrica comune per la classificazione multiclasse.

Come previsto, considerando la qualità dei frame estratti dai singoli video, i risultati sono stati molto scarsi. Il principale problema riscontrato è stata la quantità enorme di frame, ciò ha reso impossibile per le nostre macchine gestire il carico computazionale richiesto per addestrare il modello. Ogni video ha una durata di 10 secondi ed è stato registrato a 25 o 30 fotogrammi al secondo, e ciò comporta un numero di 250/300 frame per singolo video.

Successivamente è stato aumentato il resize dei frame da 128 a 192, ma sempre non preprocessati e i risultati sono rimasti comunque molto bassi. Questi primi test sono stati utili a comprendere meglio come gestire le risorse disponibili e per studiare alcuni parametri rilevanti.

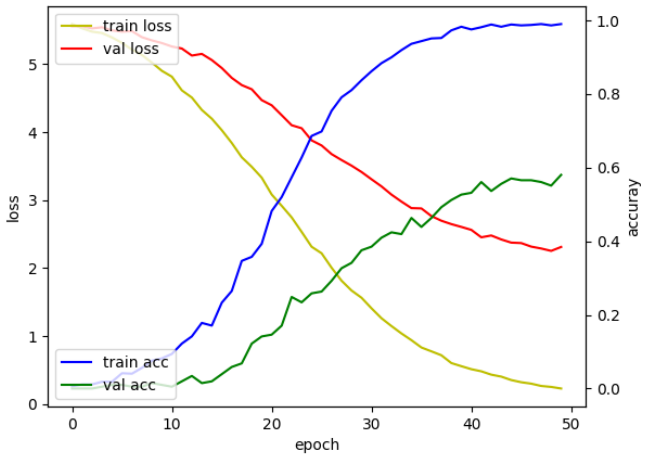


Fig.3: Test con i frame originali e un resize a 128

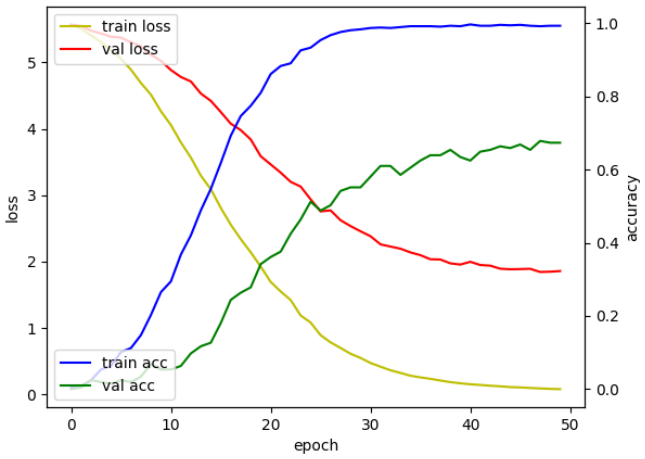


Fig.4: Test con i frame originali e un resize a 192

Dataset Babele non preprocessato		
Parametri	Test 1	Test 2
Algoritmo di ottimizzazione	Adam	Adam
Timesteps	10	10
Learning Rate	0,0001	0,0001
Batch size	32	32
Validation ratio	0,2	0,2
Epochs	50	50
Quantità di frame	Tutti	Tutti
Resize dei frame	128	192
Train Loss	0,3218	0,1294
Validation Loss	2,3785	1,9724
Test Accuracy	57,42 %	77,34 %

È stata dedicata molta attenzione all'operazione di preprocessing dei frame, e per questo motivo è stato deciso di applicarlo sul dataset ed effettuare dei test sul numero di frame da fornire alla rete. I due test principali effettuati sono i seguenti:

- Nel primo test sono stati utilizzati 30 frame per ciascun video, selezionandoli in modo uniforme e non sequenziale. Ciò significa che abbiamo estratto 30 frame distribuiti uniformemente nel corso del video.
- Nel secondo test è stato aumentato il numero di frame a 60 per ciascun video, sempre selezionandoli in modo uniforme e non sequenziale. Anche in questo caso, i 60 frame sono stati distribuiti in modo uniforme lungo l'intera durata del video.

La motivazione principale che ha portato alla scelta uniforme dei frame per ciascun video è che dopo una rapida analisi della maggior parte dei video presenti nel dataset, ci si è resi conto che l'approccio di selezionare i primi 30 o 60 frame in maniera sequenziale non sarebbe stato adeguato a catturare le diverse "situazioni" delle labbra.
Pertanto, è stato deciso di adottare un approccio non sequenziale, ma uniforme per la selezione dei frame, in modo da coprire una varietà di situazioni delle labbra all'interno di ciascun video. Questa scelta consente di catturare una rappresentazione più completa e significativa delle variazioni nella forma e nell'apertura delle labbra.

Dataset Babele

Inizialmente, è stato eseguito un test utilizzando solo 30 frame utilizzando un resize a 192, selezionati in modo uniforme, dopo una fase di preprocessing. Nonostante gli sforzi per migliorare la qualità dei frame estratti da ciascun video, i risultati di questa prova preliminare non sono stati soddisfacenti.
Successivamente, è stata condotta un'altra prova utilizzando 60 frame e applicando la tecnica dell'early stopping. Durante l'addestramento del modello, è stato monitorata la val_accuracy, che rappresenta l'accuratezza sul set di dati di validazione, con un valore di patience impostato a 4.
La patience indica il numero di epoche successive in cui la metrica monitorata non migliora prima che l'addestramento venga interrotto. In questo caso, se la val_accuracy non migliora per quattro epoche consecutive, l'addestramento viene interrotto prematuramente, evitando così l'overfitting e il tempo sprecato su ulteriori iterazioni che non apportano miglioramenti significativi.

Questa volta, i risultati sono stati migliori rispetto alla prova precedente. L'utilizzo di un numero maggiore di frame, combinato con l'early stopping, ha portato a risultati più promettenti.
Complessivamente, l'aumento del numero di frame utilizzati per il test e l'implementazione dell'early stopping hanno contribuito a migliorare i risultati del modello, fornendo risultati più promettenti rispetto alla prova iniziale.

Dataset Babele			
Parametri	Modello 1	Modello 2	Modello 3
Algoritmo di ottimizzazione	Adam	Adam	Adam
Timesteps	20	20	20
Learning Rate	0,0001	0,0001	0,0001
Batch size	32	32	32
Epochs	50	50	50
Frames	30	60	60
Resize dei frame	192	192	192
Train Loss	0,0087	0,0447	1,0434
Validation Loss	1,0742	1,0089	2,1027
Early Stopping	✗	✓	✓
Kernel Regolizer L2	✗	✗	✓
Test Accuracy	82,03 %	83,59 %	83,59 %

Per valutare ulteriormente le prestazioni sul Dataset Babele, è stata condotta un'ulteriore sperimentazione introducendo un kernel regolizer L2. Nell'ambito della sperimentazione sul Dataset Babele, il kernel regolizer L2 è stato introdotto nel processo di addestramento del modello.

L'obiettivo era controllare la complessità del modello e limitare l'overfitting. È importante sottolineare che i risultati ottenuti dalla sperimentazione con il kernel regolizer L2 possono variare a seconda dei parametri scelti per la regolarizzazione e delle caratteristiche specifiche del Dataset Babele. Pertanto, l'efficacia di questa tecnica va valutata in base al contesto specifico e alle esigenze dell'applicazione.

Durante l'analisi delle prestazioni del modello sul dataset Babele, è stata effettuata una valutazione dettagliata utilizzando le metriche di precision, recall e F1-score per ciascuna label.
Durante questa valutazione, è emerso che la precision non è ottimale per tutte le etichette. Questo significa che il modello tende a fare previsioni corrette in modo più accurato per alcune classi rispetto ad altre. Una precision inferiore per una specifica etichetta indica che il modello ha effettuato un numero significativo di previsioni errate per quella classe in particolare.

Dataset Vid-Timit

Ci è stato fornito un secondo dataset per testare la rete precedentemente addestrata e valutarne le prestazioni. Questo secondo dataset è più piccolo rispetto al precedente, poiché invece di 256 soggetti, ne comprende soltanto 43. Tuttavia, il nuovo dataset contiene più video per soggetto, con durata variabile compresa tra 3 e 5 secondi. I video sono stati registrati a una frequenza di 25 o 30 fps.
Nel caso del secondo dataset sono stati forniti alla rete neurale tutti i frame preprocessati dei video.

Al fine di evitare problemi di overfitting, sono state adottate due strategie:

- In primo luogo, è stato utilizzato l'early stopping monitorando la val_accuracy. L'addestramento viene interrotto quando la metrica non migliora più o peggiora per quattro epoche.
- Inoltre, per fare un confronto con il primo dataset è stato utilizzato anche un kernel regolarizer L2 per limitare la complessità del modello e ridurre l'overfitting.

Dopo aver applicato l'early stopping e il kernel regolarizzatore L2, è stato possibile confrontare i risultati ottenuti utilizzando il secondo dataset con quelli ottenuti utilizzando il primo dataset.

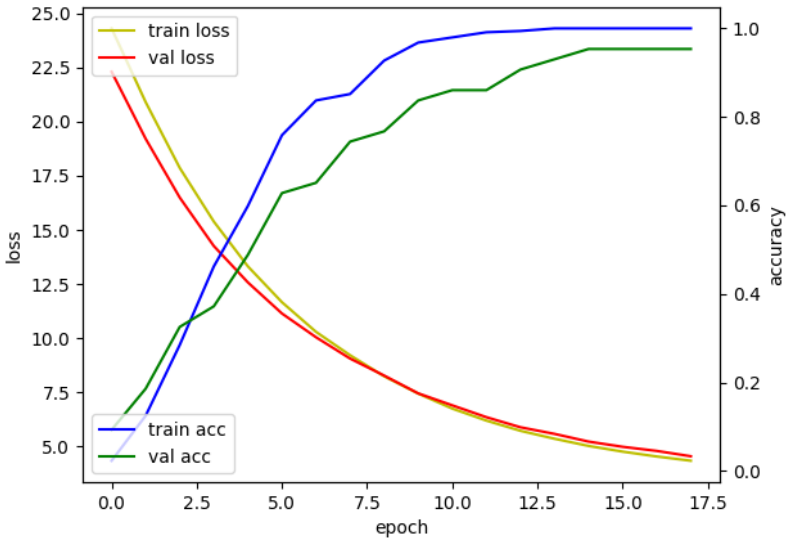


Fig.6: Test con Dataset2 preprocessato early stopping (18 epoch) e Kernel regolizer L2

Dataset VidTimit	
Parametri	Modello Finale
Algoritmo di ottimizzazione	Adam
Timesteps	20
Learning Rate	0,0001
Batch size	32
Validation ratio	0,2
Epochs	50
Resize dei frame	192
Train Loss	4,3272
Validation Loss	4,5342
Early Stopping	✓
Kernel Regolizer L2	✓
Test Accuracy	97,34 %

I risultati ottenuti hanno fornito una valutazione delle prestazioni della rete precedentemente addestrata e si è dimostrato che il secondo dataset ha ottenuto risultati migliori.

Questo è dovuto a diverse caratteristiche specifiche del secondo dataset.
Innanzitutto, il fatto che il secondo dataset contenga più video per singolo soggetto offre una maggiore varietà di informazioni e una visione più completa delle caratteristiche dei soggetti stessi.
Un altro fattore che ha contribuito al miglioramento dei risultati è l'utilizzo di tutti i frame preprocessati nel processo di addestramento. Questo consente alla rete di considerare tutte le informazioni visive disponibili nei video, senza perdere potenziali dettagli utili per la classificazione corretta.

Inoltre, dalla confusion matrix, siamo in grado di fornire una rappresentazione più dettagliata e tecnica dei risultati ottenuti sui dati di test. La confusion matrix mostra la distribuzione delle previsioni corrette ed errate effettuate dal modello per ciascuna label di output. Questo strumento ci consente di valutare l'accuratezza del modello nella classificazione delle diverse label e di identificare eventuali pattern o errori specifici.

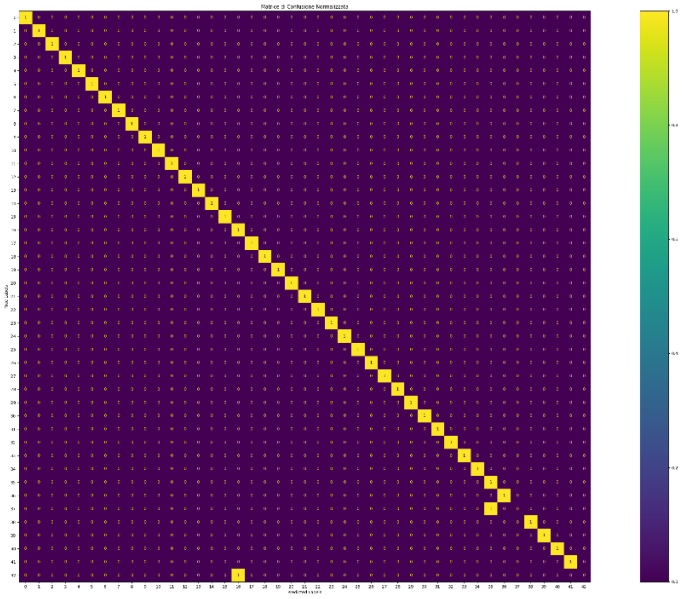


Fig.7: Confusion Matrix con i risultati avuti sui dati di Test

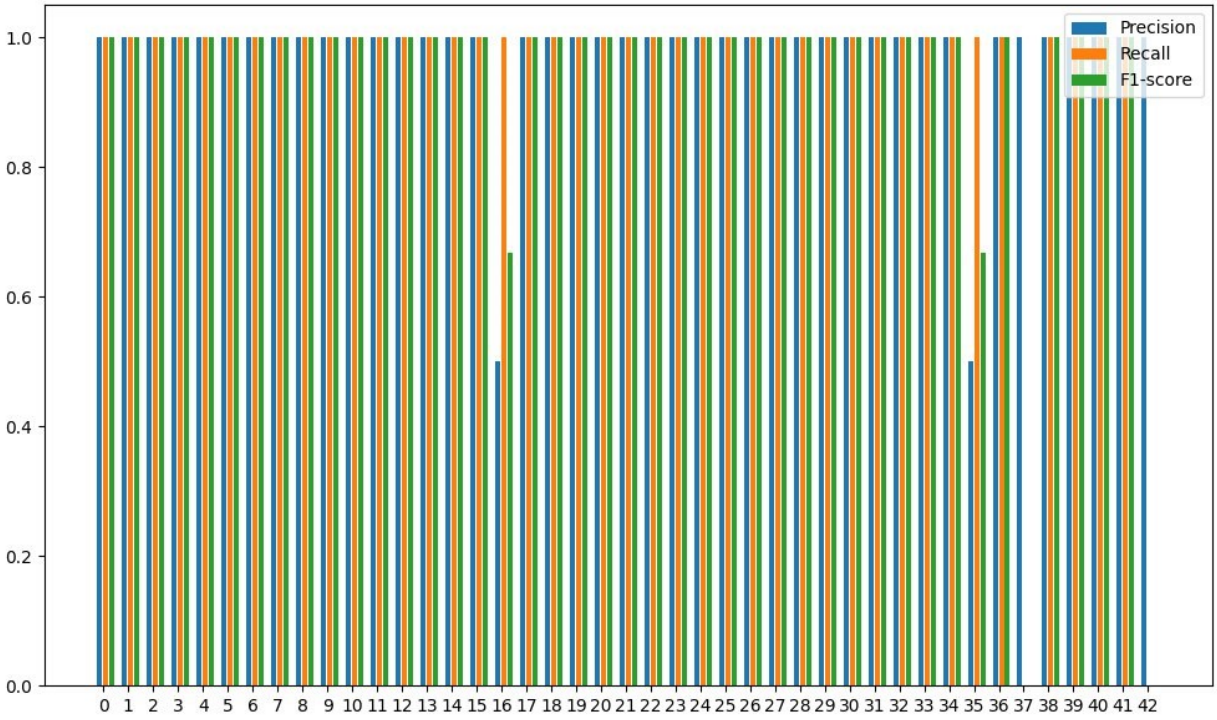


Fig.8: Istogramma con i valori di Precision, Recall e F1-score sul Dataset Vid-Timit

Considerazioni

Dopo aver eseguito l'ottimizzazione del modello per entrambi i dataset, è stata condotta un'approfondita analisi statistica per valutare le prestazioni del modello. Come parte di questo processo, è stata implementata nel codice una funzionalità per il salvataggio dei frame errati associati a ciascuna classe in una cartella dedicata. Ciò permette di analizzare visivamente e studiare in dettaglio gli errori del modello per ogni categoria.
Inoltre, è stato creato un file CSV completo che riassume tutte le informazioni rilevanti. Questo file contiene diverse colonne che registrano dettagli come l'identificatore del frame, la classe corretta e la classe predetta dal modello.

Per realizzare ciò, il sistema prende in input il test set su cui effettuare la previsione, il percorso del modello da utilizzare per la previsione, le dimensioni delle immagini utilizzate durante il training, il numero di etichette presenti nel test set e la directory di output in cui verranno salvati i frame errati. L'outputError viene strutturato in cartelle per ogni dataset, con un file CSV contenente un riepilogo completo. Inoltre, all'interno della directory outputError, vengono creati ulteriori livelli di directory: una per ogni classe reale e una per ogni classe prevista dal modello. All'interno di quest'ultima cartella, vengono salvati i frame errati. La struttura di directory sarà organizzata come segue:

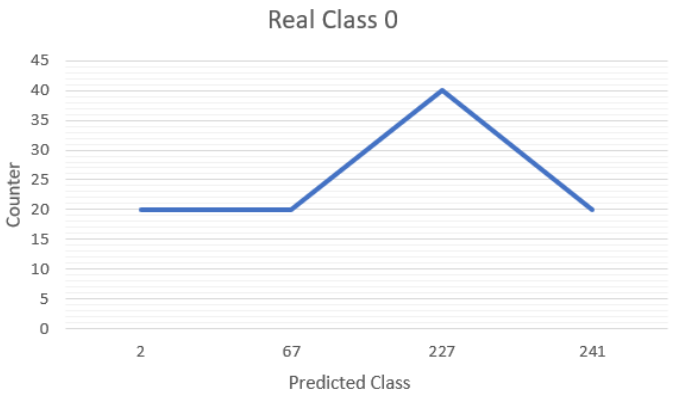
- outputError_dataset1 → CSV
- outputError_dataset1 → failure_frames → real_class_i → predicted_class_j → frame_k

Questo metodo fornisce una visione dettagliata delle prestazioni del modello, consentendo di individuare pattern, tendenze o errori specifici.
Durante la previsione, il modello prende tutti i frame di ogni etichetta del set di test e li suddivide in N gruppi di dimensione 'timesteps', che corrispondono alla dimensione con cui il modello è stato addestrato. Viene quindi controllato se la maggior parte delle previsioni è corretta. Nel caso in cui la maggior parte delle previsioni sia errata $(50\% + 1)$, tutti i frame errati vengono salvati nella lista degli oggetti di tipo 'FrameErrato'.

Durante l'analisi delle directory contenenti i frame predetti in modo errato e la classe corretta che doveva essere assegnata nel primo dataset, caratterizzato da 256 label, sono emerse importanti differenze nella performance predittiva delle varie classi. Si è osservato che alcune classi presentano una maggiore incidenza di errori rispetto ad altre.
Al fine di rendere evidenti le disparità tra le classi, sono stati generati degli istogrammi specifici per le classi più suscettibili agli errori. Questi istogrammi forniscono una rappresentazione visiva chiara della distribuzione delle predizioni errate all'interno di ciascuna classe.

Nell'istogramma, sull'asse delle x sono rappresentate le diverse label errate, mentre sull'asse delle y sono indicate le occorrenze, ossia il numero di frame che sono stati erroneamente classificati come appartenenti a quella specifica label. Questa rappresentazione grafica consente di valutare l'entità degli errori commessi per ciascuna classe e di confrontare la frequenza degli errori tra le diverse categorie.

L'istogramma facilita l'individuazione delle classi che presentano un numero significativo di predizioni errate, fornendo una chiara visualizzazione delle discrepanze nella distribuzione degli errori. Di seguito alcuni esempi.

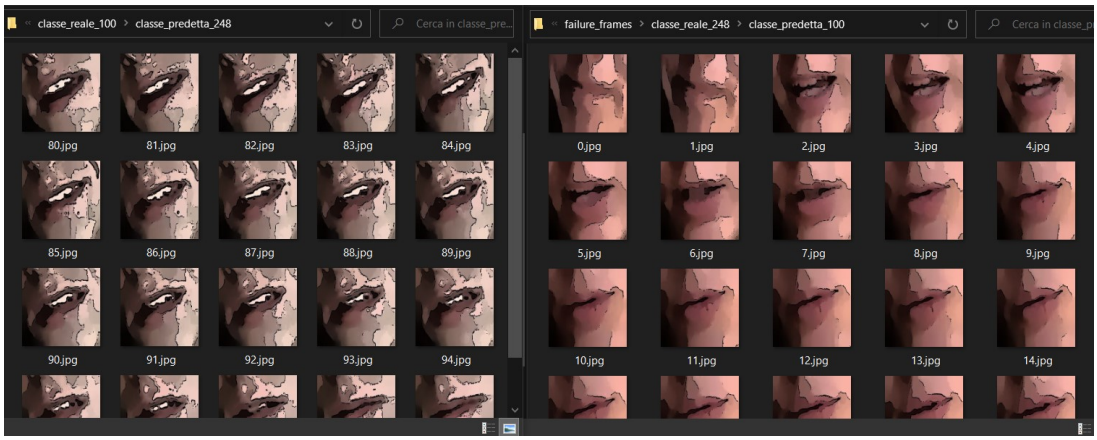


Tra le varie classi, è emerso che la classe reale 0, insieme ad altre poche classi, risulta essere particolarmente suscettibile agli errori del modello. In queste classi ci sono soggetti di genere maschile che condividono somiglianze notevoli con altri soggetti maschili che sono stati erroneamente predetti dal modello. Analizzando i risultati ottenuti dal modello, emerge che l'errore nella previsione riguarda principalmente la confusione tra soggetti maschili anziché coinvolgere errori tra soggetti di genere diverso. Questa tendenza è comprensibile, considerando che i soggetti maschili sono spesso caratterizzati da barba e/o baffi, mentre i soggetti femminili sono spesso caratterizzati dalla presenza del rossetto. Di conseguenza, il modello commette errori principalmente nel distinguere tra soggetti maschili simili, così come tra soggetti femminili simili.

Durante l'analisi, è emerso un aspetto rilevante riguardante il comportamento speculare degli errori di predizione del modello. Ciò significa che, quando il modello commette un errore nel classificare una classe specifica, si riscontra un errore simmetrico in cui viene commessa la stessa confusione tra le due classi coinvolte.
In particolare, utilizzando questa implementazione nel codice per la predizione delle label del Dataset Babele è stato rilevato che il modello ha predetto correttamente 243 label su un totale di 256 label. Tuttavia, su 13 label erroneamente predette, è interessante notare che 8 di queste sono errori speculari. Ciò implica che il modello ha commesso confusione tra una classe X e la classe Y, e allo stesso tempo ha commesso confusione tra la classe Y e la classe X.

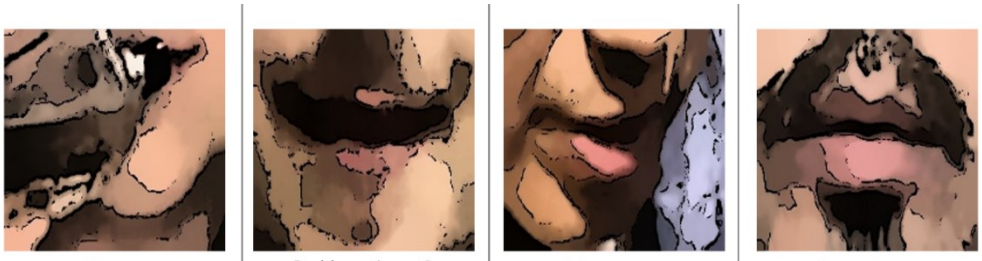
Questo comportamento speculare evidenzia la presenza di una simmetria o un'ambiguità nelle caratteristiche distintive delle classi coinvolte.

Per illustrare questo fenomeno, prendiamo ad esempio le classi 100 e 248 nel nostro dataset. È stato osservato che il modello tende a confondere la classe 100 con la classe 248, e allo stesso tempo, la classe 248 viene confusa con la classe 100. Questo errore speculare si verifica per un totale di 20 frame all'interno del dataset. In questo caso si può notare come l'orientamento del volto combinato con una situazione di labbra più o meno chiuse incide sulla prediction fatta dal modello.



Ci sono diverse motivazioni principali che influiscono negativamente sulla corretta classificazione delle labbra nei frame:

- Decontestualizzazione dei frame[1]: alcuni frame sono privi di contesto, il che significa che non contengono informazioni sufficienti per riconoscere e classificare correttamente le labbra. In questi frame è assente la presenza delle labbra o sono coperte.
- Labbra molto chiuse[2]: la presenza di labbra molto chiuse, come nel caso di un sorriso stretto o una chiusura completa delle labbra.
- Rotazione del volto[3]: se il volto è ruotato in modo significativo rispetto all'orientamento standard, la forma delle labbra può apparire distorta o non allineata rispetto alla posizione attesa.
- Presenza di barba[4]: la presenza di barba può aggiungere ulteriore complessità alla classificazione delle labbra. La peluria o i peli della barba possono nascondere parzialmente o completamente le labbra. La presenza di barba richiede al modello di adattarsi a variazioni nell'aspetto delle labbra e di considerare ulteriori fattori per una classificazione accurata.


Problematica n. 1 | Problematica n. 2 | Problematica n. 3 | Problematica n. 4

Sviluppi futuri

Dalle sperimentazioni effettuate e dai test condotti su entrambi i dataset forniti, si è arrivati alla conclusione che preprocessare un numero maggiore di frame per ciascun video (ad esempio, 120 frame per video) potrebbe portare a risultati di accuracy più elevati.
Tuttavia, l'applicazione di questo approccio è limitata dalla nostra potenza computazionale, poiché richiede maggiori risorse di calcolo per la fase di preprocessing (poiché l'algoritmo di Mean Shift utilizzato ha un costo computazionale di $O(kN^2d)$ dove k è il numero di iterazioni, N è la finestra e d il numero di feature) e soprattutto per la fase di addestramento del modello.

Nonostante queste limitazioni, è possibile cercare di migliorare ulteriormente il processo di preprocessing dei frame esplorando l'utilizzo di algoritmi diversi rispetto a quelli precedentemente citati e utilizzati nello sviluppo del modello.
Inoltre, è possibile definire in modo più dettagliato degli iperparametri del modello. Gli iperparametri, come il learning rate, il numero di strati del modello, la dimensione dei filtri convoluzionali e altri parametri di configurazione, possono essere regolati in modo da massimizzare le prestazioni del modello durante l'addestramento.

In sintesi, nonostante il limite computazionale che impedisce di preprocessare un numero maggiore di frame per video, ci sono ancora diverse possibilità per migliorare ulteriormente il preprocessing dei frame e l'addestramento del modello.