Cross Modal Analysis per Speaker Detection attraverso Canonical Correlation Analysis

Carmine Romaniello

Abstract—In questo paper viene illustrata l'applicazione della Canonical Correlation Analysis (CCA) implementata dalla libreria Pyrcca. Con essa viene eseguita la Cross Modal Analysis per affrontare il problema di Speaker Detection (riconoscimento di chi sta parlando) in contenuti multimediali audio-video. Un esempio di input/output è rappresentato in figura 1. I video utilizzati nella fase di training sono stati reperiti su Youtube, mentre per la fase di test sono stati selezionati alcuni già etichettati reperiti dal dataset del lavoro [1]. Dal video in input vengono estratte le features audio e le features video da ogni volto che verranno utilizzate per determinare quello parlante. Per le feature audio vengono estratti 12 MFCC (Mel Frequency Coepstral Coefficent). Per le features video vengono affrontati due approcci: il primo in cui si estraggono le Action Units, il secondo in cui invece si estraggono le distanze tra i Landmark relativi alla bocca.
Nella sezione 5 vengono infine analizzati i risultati ottenuti con entrambi gli approcci in cui il secondo si è dimostrato decisamente migliore.

Fig. 1: Esempio di input e output in cui viene marcato il volto di chi sta parlando

C. Romaniello 921326, Progetto d'esame di Interazione Naturale e Modelli di Computazione Affettiva, A/A 2018-2019, Universitá di Milano, via Celoria 18, Milano, Italy.
E-mail: carmine.romaniello@studenti.unimi.it

1 Introduzione

Speaker Detection è un problema fondamentale nell'ambito di settori applicativi, come le videoconferenze, caratterizzati da file multimediali aventi stream audio-video. Esso consiste nell'identificare chi parla e quando all'interno di un video in cui sono presenti più persone. I classici approcci a tale problema prevedono l'analisi e l'elaborazione separata dei due stream audio e video le cui informazioni vengono poi eventualmente fuse per fornire l'output finale. Questo tipo di approccio però ha lo svantaggio di perdere molte informazioni derivanti dalle relazioni semantico-temporali tra i due stream sincronizzati. Infatti nel momento in cui una persona sta parlando è ovvio che a dati movimenti del corpo, in particolare quelli delle labbra, corrispondano dati suoni. Analizzando ed elaborando le due modalità separatamente queste informazioni vengono di conseguenza perse.
Questa limitazione viene superata adottando la `Cross Modal Analysis`. Essa permette di effettuare un'analisi incrociata per ottenere un maggior contenuto informativo dalle `cross modal associations` tra immagini e suoni negli stessi istanti di tempo.
Lo scopo di questo progetto è quello di eseguire la Cross Modal Analysis, per affrontare il problema di Speaker Detection, applicando la CCA. Essa consiste nel suddividere il video in frame di lunghezza fissa ed estrarre per ogni frame un insieme di feature video e un insieme di feature audio. CCA analizza le `cross modal associations` tra i due insiemi per definire un spazio in cui proiettarli che massimizzi il loro valore di correlazione. L'obbiettivo è quello di ottenere, per ogni frame, un valore di correlazione maggiore per la persona che sta parlando rispetto a quelle in silenzio. Per la fase di training sono stati selezionati un insieme di video dove compare una sola persona

che sta parlando, mentre per la fase di test sono stati utilizzati video dove sono presenti più persone. Verranno mostrati due approcci in cui dallo stream video sono stati scelti due diversi tipi di feature video, nel primo corrispondono alle `action unit` mentre nel secondo alle `distanze tra landmarks`. Queste verranno poi esposte in modo più approfondito nella sezione 4.5.
I risultati ottenuti sono stati abbastanza soddisfacenti, in particolare quelli ottenuti con il secondo approccio sono stati di gran lunga migliori rispetto al primo. Non mancano però alcuni problemi di imprecisione in quei casi in cui la persona che non parla tende a muovere o a tenere aperta la bocca.

2 Stato dell'arte

Esistono diversi lavori in letteratura in cui viene applicato CCA per l'analisi cross modale tra datasets contenenti diverse features. Un particolare campo di interesse riguarda lo studio di neuroimmagini per analizzare la relazione tra l'attività di determinate aree cerebrali e specifiche funzioni cerebrali affrontato in [2]. Gli stessi autori hanno sviluppato la libreria PyrCCA utilizzata in questo progetto. Per quanto riguarda invece il problema di Speaker Detection sono presenti vari lavori in cui vengono confrontati i risultati ottenuti con diversi approcci oltre a CCA. Il più interessante lo si può trovare in [3] dove i risultati degli esperimenti mostrano un accuracy tra il 70% e 80%. Questo è stato usato come principale riferimento in questo progetto.

3 Modello teorico

CCA è un metodo per la ricerca di relazioni lineari tra due o più datasets multidimensionali. Esso cerca uno spazio di coordinate canoniche che massimizza la correlazione tra le proiezioni dei dataset in quello spazio. CCA non richiede che i datasets abbiano la stessa dimensione e non presuppone che uno sia dipendente dall'altro, ma solamente che non siano tra loro indipendenti [4].

3.1 Descrizione formale

Dati due insiemi di variabili X e Y, $X = (X_1, X_2, ..., X_p) \in \mathbb{R}^p$ e $Y = (Y_1, Y_2, ..., Y_q) \in \mathbb{R}^q$, si definiscono due insiemi di combinazioni lineari U e V. U corrisponde all'insieme di combinazioni lineari ottenute da X mentre V a quelle da Y, dove gli elementi U_i e Vi vengono detti **componenti canoniche**. Il numero di componenti canoniche è limitato dalla dimensione di X e Y e quindi:

$$numCC \leq min\{p, q\} \tag{1}$$

Assumendo $p \leq q$:

$$\begin{aligned} U_1 &= a_{11}X_1 + a_{12}X_2 + \cdots + a_{1p}X_p \\ U_2 &= a_{21}X_1 + a_{22}X_2 + \cdots + a_{2p}X_p \\ &\vdots \\ U_p &= a_{p1}X_1 + a_{p2}X_2 + \cdots + a_{pp}X_p \end{aligned}$$

$$\begin{aligned} V_1 &= b_{11}Y_1 + b_{12}Y_2 + \cdots + b_{1q}Y_q \\ V_2 &= b_{21}Y_1 + b_{22}Y_2 + \cdots + b_{2q}Y_q \\ &\vdots \\ V_p &= b_{p1}Y_1 + b_{p2}Y_2 + \cdots + b_{pq}Y_q \end{aligned}$$

Ogni componente canonica U_i viene accoppiata con la componente V_i a formare l'i-esima coppia canonica (U_i, V_i). Successivamente viene calcolata la varianza delle componenti U_i e delle componenti V_i:

$$var(U_i) = \sum_{k=1}^{p}\sum_{l=1}^{p} a_{ik}a_{il}cov(X_k, X_l)$$

$$var(V_i) = \sum_{k=1}^{p}\sum_{l=1}^{q} b_{ik}b_{il}cov(Y_k, Y_l)$$

poi la covarianza tra U_i e V_i:

$$cov(U_i, V_i) = \sum_{k=1}^{p}\sum_{l=1}^{q} a_{ik}b_{il}cov(X_k, Y_l)$$

La correlazione tra U_i e V_i calcolata con il **coefficente di correlazione di Pearson**:

$$\rho_i = \frac{cov(U_i, V_i)}{\sqrt{var(U_i)var(V_i)}} \tag{2}$$

è la quantità che si vuole massimizzare attraverso la giusta scelta dei vettori di coefficenti a_i e b_i per ogni componente canonica i.
Nel caso specifico del problema Speaker Detection il metodo CCA viene applicato a due datasets multidimensionali. Dato un contenuto multimediale composto da uno stream video e uno stream audio, vengono estratte p features video da ogni frame. Per ognuno di questi frame vengono estratte q features audio nella stessa finestra temporale del frame video. Assumendo ci siano per ogni stream n frames, si avranno due matrici:

$$\begin{aligned} \mathbf{X} &= (x_1, x_2, ..., x_p) \in \mathbb{R}^{n\times p} \\ \mathbf{Y} &= (y_1, y_2, ..., y_q) \in \mathbb{R}^{n\times q} \end{aligned} \tag{3}$$

dove x_i e y_i sono vettori n-dimensionali.
La soluzione al problema è data dalle due matrici:

$$\begin{aligned} \mathbf{A} &= (a_1, a_2, ..., a_d) \in \mathbb{R}^{p\times d} \\ \mathbf{B} &= (b_1, b_2, ..., b_d) \in \mathbb{R}^{q\times d} \end{aligned} \tag{4}$$

dove $d = numCC$, a_i sono vettori p-dimensionali e b_i sono vettori q-dimensionali.
La funzione obbiettivo sarà quindi:

$$\rho = \arg\max_{A,B} \frac{cov(X \cdot A, Y \cdot B)}{\sqrt{var(X \cdot A)var(Y \cdot B)}} \tag{5}$$

che corrsiponde alla massima correlazione tra le proiezioni di ogni frame video e il relativo frame audio.

3.2 Kernel CCA

CCA cerca delle relazioni lineari tra i due datasets assumendo che non siano tra loro indipendenti, ma che ci sia una relazione non causale che li lega. Alle volte tali relazioni potrebbero essere non lineari quindi, prima di applicare CCA, è necessario proiettare i dati in uno spazio dimensionale maggiore. Per poter catturare le eventuali relazioni non lineari la funzione di proiezione deve essere non lineare come ad esempio una funzione polinomiale o gaussiana ([5] per approfondimenti). Questo procedimento è conosciuto con il nome di **kernel trick** o **kernelizzazione** e la funzione di proiezione viene chiamata **funzione kernel**.

3.3 CCA regolarizzato

Se i dataset X e Y hanno una dimensione $n \leq min\{p, q\}$ oppure viene applicata la kernelizzazione è possibile incorrere nell'overfitting. Per mitigare l'overfitting si può regolarizzare CCA vincolando i vettori dei pesi per mantenere la convessità del problema. Il valore λ usato per regolarizzare viene detto **coefficente di regolarizzazione** e la funzione obbiettivo in 5 diventa:

$$\rho = \arg\max_{A,B} \frac{cov(X \cdot A, Y \cdot B)}{\sqrt{(var(X \cdot A) + \lambda A^2)\ (var(Y \cdot B) + \lambda B^2)}} \tag{6}$$

3.4 Training

La fase di training consiste nel costruire uno spazio di coordinate canoniche di dimensione $numCC$ definita in 1 in cui proiettare i due insiemi di variabili. Quindi per ogni dimensione i dello spazio canonico è necessario trovare una coppia di vettori di proiezione $\boldsymbol{a_i} = (a_1, a_2, ..., a_p)$ e $\boldsymbol{b_i} = (b_1, b_2, ..., b_q)$, chiamati **pesi canonici**, che massimizzino la correlazione in 2. Per ottenere lo spazio canonico migliore che fornisca la massima correlazione audio/video, i datasets di training devono essere composti solamente da features di volti che parlano. Inoltre bisogna impostare in modo opportuno 3 parametri:

- Se applicare o meno la kernelizzazione e di che tipo: lineare, polinomiale o gaussiana;
- il valore del coefficente di regolarizzazione;
- il numero di componenti canoniche in 1 da trovare tra i due datasets;

Terminato il training si ottengono le due matrici dei pesi canonici formalizzate in 4 che verrano utilizzate per la proiezione dei vettori di features nello spazio canonico.

3.5 Classificazione dei volti

Dato un frame f in cui sono presenti n volti diversi sia $X = \{X_1, X_2, ..., X_n\}$ l'insieme di vettori di features video estratti dagli n volti e sia Y il vettore unico di features estratto dall'audio. Si calcolano le proiezioni dei vettori nello spazio canonico utilizzando le matrici A e B formalizzate in 4:

$$X' = \{X_1 \cdot A, X_2 \cdot A, ..., X_n \cdot A\}$$

$$Y' = Y \cdot B$$

Al frame f viene quindi associato l'indice i del volto che viene classificato come parlante determinando quale tra le proiezioni dei vettori X_i ha la maggior correlazione con la proiezione del vettore Y:

$$C_f \leftarrow i \mid corr(X'_i, Y') = \max\{corr(X'_i, Y'), \forall i \in [1, n]\} \tag{7}$$

dove la correlazione $corr(V_1, V_2)$ tra due vettori V_1, V_2 viene calcolata come in 2 e riportata qui per completezza:

$$corr(V_1, V_2) = \frac{cov(V_1, V_2)}{\sqrt{var(V_1)var(V_2)}}$$

4 Simulazione e esperimenti

CCA prende in input due datasets (uno relativo allo stream video e l'altro allo stream audio) di pari dimensione corrispondente al numero di frame. Considerando che il problema consiste nell'identificare il volto parlante, si localizzano prima di tutto le singole regioni in cui è presente un volto e per ognuna di queste si estraggono le **face features**. In parallelo vengono estratte le features audio che, a differenza delle precedenti, saranno uniche per ogni frame. E' necessario effettuare dapprima una scelta su quali sono le face features e quali le features audio che forniscono la migliore informazione possibile. Per l'audio si estraggono 12 Mel Frequency Coepstral Coefficents (MFCC) come in altri lavori che affrontano il problema Speaker Detection in cui viene applicato CCA [3] [6]. MFCC sono perfetti per questo tipo di applicazione in quanto si basano sulla percezione dell'udito umano misurata attraverso la scala di frequenza Mel. Essa consiste in una scala in cui è presente qualunque intonazione della voce permettendo di catturare importanti caratteristiche della fonetica nel parlato [7]. Per le face feature le uniche informazioni rilevanti sono quelle relative ai movimenti delle labbra, in quanto il susseguirsi della loro apertura e chiusura è sufficiente per dedurre che un dato volto è un potenziale candidato ad essere quello parlante. Openface fornisce due diversi tipi di features per modellare i movimenti delle labbra, quindi gli esperimenti sono stati eseguiti seguendo due approcci.

4.1 Approccio Action Units

A seconda dei movimenti dei singoli muscoli facciali è possibile ricavare l'espressione assunta dal volto. Queste espressioni prendono il nome di **action units** (AU) e sono codificate dal **FACS** (Facial Action Coding system). In totale vengono identificate 42 AU [8], ma per questo approccio si utlizzano unicamente le 9 relative alla bocca perchè, considerando il problema trattato, le altre rappresenterebbero solamente rumore. Ogni AU viene misurata da un valore reale compreso tra 0 e 5 (Openface standard) che determina quanto è marcata una data espressione del volto.

4.2 Approccio Landmarks Distances

Su ogni volto presente nel frame vengono fissati un totale di 68 marcatori (**landmarks**) come rappresentato in figura 2. Ogni landmark è identificato da una coppia di coordinate (x,y) sul piano dell'intero frame e ogni coordinata indica il numero di pixel dal margine superiore sinistro. Tra tutti i 68 vengono considerati solamente i 20 relativi alla bocca perchè le variazioni delle loro posizioni rappresentano movimenti delle labbra (figura 3). Calcolando la distanza tra diversi

landmark della bocca è possibile rilevare l'apertura e la chiusura di quest'ultima, se le distanze sono piccole allora la bocca sarà chiusa viceversa sarà aperta. In questo approccio si utilizzano le 10 distanze euclidee calcolate tra i landmarks delle coppie: {(50, 58), (61, 67), (51, 57), (62, 66), (52, 56), (63, 65), (49, 59), (53, 55) (48, 54), (60, 64)}. Queste distanze vengono poi normalizzate per evitare che una stessa apertura comporti dei valori delle distanze maggiori per un volto in primo piano rispetto ad uno in secondo piano. L'operazione di normalizzazione è fondamentale in quanto nei video utilizzati sono presenti più persone riprese in piani diversi.

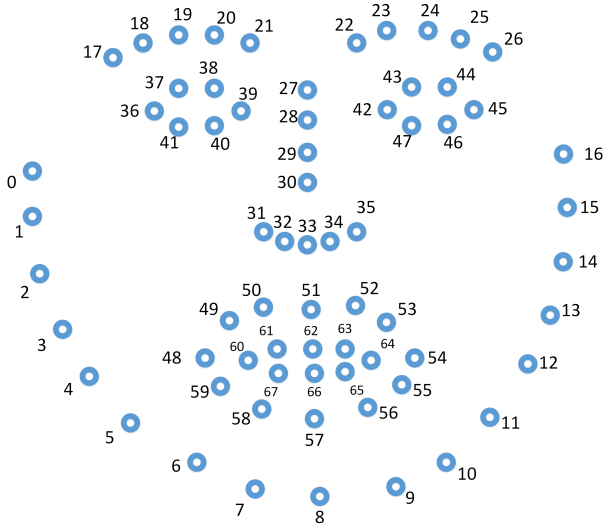

Fig. 2: Rappresentazione dei 68 landmarks con i relativi indici

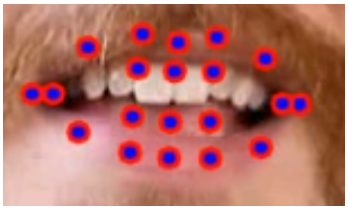

Fig. 3: Identificazione dei 20 landmarks relativi alla bocca in un frame video

4.3 Dataset

4.3.1 Training Set

Nei datasets di training le features di volti in silenzio rappresentano rumore, pertanto bisogna individuare la presenza di eventuali pause durante il parlato ed eliminare i relativi frames. Per avere il miglior dataset possibile sono stati scelti e tagliati 5 video reperiti su `Youtube` in modo tale che:

- gli individui fossero eterogenei e quindi sia uomini che donne;
- le pause durante il parlato fossero quasi nulle e quindi con il minimo numero di frame in silenzio (per questo i migliori sono stati video di giornalisti e bloggers);
- l'audio non fosse sporcato da rumore di sottofondo;
- i movimenti della testa fossero lievi così da poter rilevare il volto con la maggior precisione possibile;

Sono stati utilizzati in tutto 5 video per un totale di circa 7500 frames.

4.3.2 Test Set

Per la fase di test sono stati utilizzati alcuni video presenti nella raccolta del lavoro in [1] (scaricabili all'indirizzo https://github.com/yufanLIU/find/tree/master/Our_database/raw_videos) con allegati i corrispettivi vettori di etichette. Questi vettori hanno lunghezza pari al numero di frames e gli elementi corrispondono all'indice del volto che sta parlando.

4.4 Architettura del sistema

Nella figura 4 è rappresentato il diagramma dell'archiettura del sistema. Il sistema è formato da una pipeline per ogni stream composte da alcune fasi in comune con funzioni diverse a seconda dello stream:

- **Streams Extraction**: Prende in input il video e da esso estrae i due stream video e audio che parallelamente seguiranno diversi processi di elaborazione;
- **Face Detection**: Elabora solamente lo stream video da cui rileva ed estrae i volti presenti in ogni frame;
- **Features Extraction**: Estrae da ogni frame le features relative allo stream che sta processando. Per lo stream video estrae le AU (4.1) o le LD (4.2) da ogni volto, per lo stream audio estrae 12 MFCC;
- **Silent frames Removal**: Rimuove da entrambi gli stream i frame in cui non viene rilevata alcuna voce;
- **Audio Visual Cross Modal Association**: E' il cuore dell'intero sistema in cui nella fase di training viene eseguito CCA per definire le matrici dei pesi. Nella fase di test le matrici costruite precedentemente vengono utilizzate per proiettare la features e determinare il volto che parla in ogni frame;

4.5 Dettagli implementativi

Il progetto è stato interamente sviluppato in Python. I video in input sono file nel formato mp4. Lo stream video viene estratto e processato con `Openface` nelle prime tre fasi viste nella sottosezione precedente. Lo stream audio viene estratto con il tool `ffmpeg` e le features vengono estratte attraverso la libreria `scipy` in combinazione con `python_speech_features`. Lo stage che esegue la Cross Modal Analysis è implementato dalla libreria `PyrCCA`.

4.5.1 Problemi di implementazione

Nella fase di Face Detection Openface presenta alcune limitazioni: non riesce a rilevare più di quattro volti contemporaneamente ed è impreciso se queste non sono rivolte verso la telecamera o fanno movimenti molto rapidi. Per la rimozione dei frame in silenzio non è stato trovato alcun tool in grado di rilevarli in modo abbastanza accurato e anche i diversi tentativi di implementazione non hanno fornito soluzioni soddisfacenti. Perciò, per ovviare al problema ed evitare imprecisioni, in fase di training sono stati rimossi manualmente tagliando i video. In fase di test vengono rilevati direttamente dai vettori delle etichette quindi non vengono nemmeno testati.

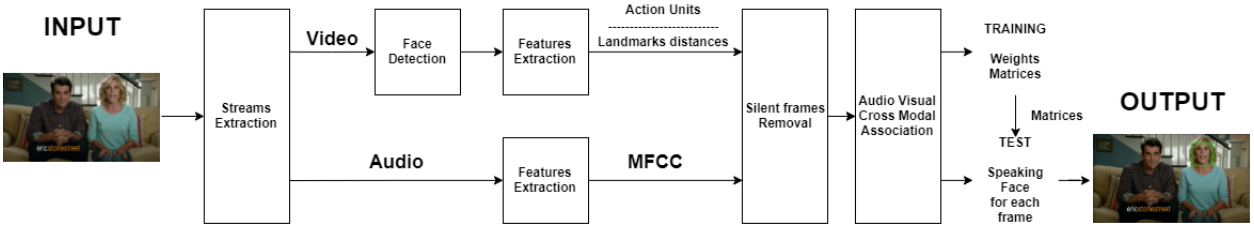

Fig. 4: Architettura del sistema

4.5.2 *logica applicativa - Pseudocodice*

Algorithm 1 Generazione dei datasets ed esecuzione training e test

```
//Training
v_matrix, a_matrix ← [], []
for i = 1 to number_of_trainings do
    v_matrix.insert(video_features(i))
    a_matrix.insert(audio_features(i))
w_v, w_a = cca_train(v_matrix, a_matrix)
//Test
for t = 1 to number_of_tests do
    for frame = 1 to number_of_frames do
        a_feats ← audio_features(t) · w_a
        max_corr ← 0
        for face = 1 to number_of_faces do
            v_feats ← video_features(t, face) · w_v
            corr ← corr(v_feats[frame], a_feats[frame])
            if corr > max_corr then
                max_corr ← corr
                speakers[frame] ← face
    print(speakers)
```

4.6 Parametri di training

Per eseguire il training CCA necessita di quattro parametri:

- **kernelcca**: specifica se applicare o meno la kernelizzazione mostrata nella sottosezione 3.2;
- **ktype**: nel caso il precedente sia True bisogna specificare che tipo di funzione kernel utilizzare;
- **reg**: specifica il valore del coefficente di regolarizzazione esposto nella sottosezione 3.3;
- **numCC**: specifica il numero di componenti canoniche definite in (1) che CCA dovrà trovare tra i due datasets;

Il training è stato eseguito con tutte le diverse configurazioni di parametri:

- **kernelcca**: True e False
- **ktype**: lineare, polinomiale e gaussiana
- **reg**: {100, 10, 1, 0, 0.1, 0.01, 0.001, 0.0001, 0.00001}
- **numCC**: tutti i valori nell' intervallo [3,10] per l'approccio in 4.2 e [3,9] per 4.1.

Per l'approccio 4.2 la configurazione che ha dato i risultati migliori è stata:

- **kernelcca**: True
- **ktype**: Polinomiale
- **reg**: 0.1
- **numCC**: 8

Per l'approccio 4.1:

- **kernelcca**: True
- **ktype**: Polinomiale
- **reg**: 0.1
- **numCC**: 4

5 Risultati ottenuti

Le prestazioni raggiunte nei singoli test sono state calcolate tenendo conto dei problemi di implementazione esposti nella sottosez. 4.5.1. A causa di essi, in alcuni frames, il calcolo della correlazione può comportare degli errori non imputabili a CCA, ma alla qualità delle features che vengono estratte e fornite. Perciò la percentuale di accuratezza è stata calcolata come rapporto tra il numero di frame correttamente classificati e il numero di frame "classificabili" ovvero il numero di frames non in silenzio e in cui Openface riesce a rilevare correttamente almeno due volti:

$$Accuracy = \frac{Correct\_classified\_frames}{Classifiable\_frames} \tag{8}$$

In questo modo si ottiene il valore di accuratezza di CCA in un sistema "ideale", dove le eventuali imprecisioni dei tool dedicati all'estrazione delle features non influiscono nel calcolo della correlazione.
Nella tabella 1 vengono mostrati i risultati raggiunti in 16 diversi video con entrambi gli approcci `Landmarks Distances (LD)` e `Action Units (AU)` esposti in 4.2 e 4.1. Da questi è immediato evincere che con il primo approccio si raggiungono prestazioni di gran lunga migliori in quasi tutti i video con una media totale del 71% rispetto al 45% ottenuta con il secondo. Dopo aver constatato che l'approccio `LD` è quello migliore, sono stati analizzati i risultati delle classificazioni. Da questi è emerso che in alcuni casi vi sono piccoli segmenti di pochi frame scorretti contornati da segmenti corretti (Figura 5).

0	0	0	1	0	1	1	0	1	0	0	0	0

Fig. 5: Esempio di frame scorrettamente classificati (volto 1) in un video con due volti (0 e 1)

Se si considera che la durata di un frame è inferiore al decimo di secondo, un caso del genere denota una situazione inverosimile. Infatti ciò indicherebbe che potenzialmente un individuo possa parlare per un tempo quasi nullo. Questo è dovuto al fatto che la posizione assunta dalla bocca di una persona che parla potrebbe essere simile se non identica a quella di una persona in silenzio con la bocca chiusa. Ad esempio, quando vengono pronunciate le consonanti "p" o "m" la bocca è ovviamente chiusa, di conseguenza il valore di correlazione del volto che sta parlando potrebbe essere, per qualche frame, uguale o addirittura inferiore ad uno in silenzio. Per mitigare questo problema è stato scelto di classificare un frame con l'indice del volto che in media ha la correlazione più alta in un intorno del frame. Per determinare il numero di frames precedenti e successivi da considerare sono stati testati tutti valori nell'intervallo [0,30]. Il valore 30 è stato scelto come massimo in quanto equivale ad un intorno di dimensione 60 che nella maggior parte dei video (con 30 fps) corrisponde a 2 secondi di tempo. Per ogni valore sono stati testati di nuovo tutti i video e viene calcolata la media delle accuracy ottenute (fig. 6).
Da quest'analisi è emerso che classificando ogni frame in base alla media dei valori di correlazione in un intorno di raggio 10, l'accuracy media aumenta sensibilmente. La tabella 2 rappresenta i risultati finali ottenuti nei singoli test applicando questo diverso metodo di classificazione.
Nonostante il notevole miglioramento ci sono due casi di test: `044` e `056`, la cui accuracy risulta al di sotto del 50%. La particolarità di questi due test è dovuta ai volti in silenzio che sono inclini a rimanere con la bocca aperta o semiaperta e vengono classificati scorrettamente come parlanti in molti frames.

TABLE 1: Accuracy ottenuta nei video i cui nomi corrispondono a quelli presenti nel test set

Video	LD	AU
002	74 %	56 %
006	79 %	36 %
008	76 %	75 %
014	80 %	24 %
016	98 %	87 %
018	73 %	25 %
025	78 %	38 %
035	65 %	41 %
044	53 %	56 %
053	75 %	14 %
056	36 %	11 %
057	67 %	80 %
061	66 %	37 %
068	82 %	77 %
073	61 %	42 %
075	75 %	30 %
Media	71 %	45 %

6 Commenti conclusivi

In conclusione, l'applicazione della CCA per affrontare il problema Speaker Detection risponde in modo abbastanza soddisfacente, ma in alcuni casi presenta un limite non trascurabile. Se i volti in silenzio muovono le labbra e/o non chiudono la bocca posssono essere identificati come parlanti e quindi classificati in modo errato. Questo problema è

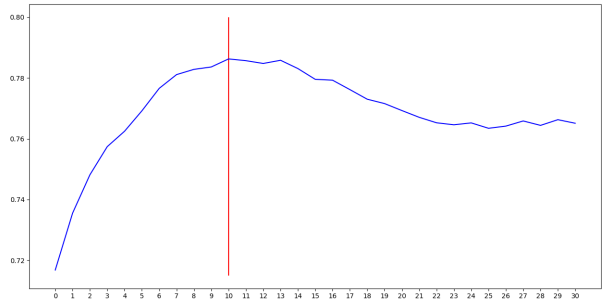

Fig. 6: Rappresentazione dell'accuracy media al variare del raggio dell'intorno

TABLE 2: Risultati finali ottenuti classificando i frames in base alla correlazione media in un intorno di raggio 10

Video	Accuracy
002	84 %
006	88 %
008	77 %
014	90 %
016	100 %
018	89 %
025	98 %
035	79 %
044	47 %
053	80 %
056	41 %
057	71 %
061	71 %
068	83 %
073	70 %
075	85 %
Media	78.6 %

dovuto al fatto che lo spazio canonico costruito in fase di training dovrebbe massimizzare la correlazione tra tutte le possibili posizioni della bocca e i relativi fonemi di qualsiasi lingua. Ciò significa che servirebbero un numero di esempi molto elevato e una macchina che possa eseguire il training in tempi ragionevoli. Infatti, per lo stesso tipo di problema, nel lavoro in [3] sono stati utilizzati circa 50000 esempi e il training è stato eseguito su cluster distribuito. Per questo progetto purtroppo i limiti di un comune computer desktop non hanno permesso di superare i 7500 esempi.
Altri problemi aperti che possono essere superati e approfonditi in futuro sono quelli della sottosez. 4.5.1:

- Utilizzo di un tool per l'estrazione delle face features che superi i limiti di Openface
- Implementazione di un tool per la rimozione di frames in silenzio

References

[1] M. X. Yufan Liu. (2016) Find-who-to-look-at. [Online]. Available: https://github.com/yufanLiu/find
[2] N. Y. Bilenko and J. L. Gallant, "Pyrcca: regularized kernel canonical correlation analysis in python and its applications to neuroimaging," *Frontiers in neuroinformatics*, vol. 10, p. 49, 2016.
[3] D. Li, N. Dimitrova, M. Li, and I. K. Sethi, "Multimedia content processing through cross-modal association," in *Proceedings of the eleventh ACM international conference on Multimedia*. ACM, 2003, pp. 604–611.

[4] P. E. C. of Science. Canonical correlation analysis. [Online]. Available: https://newonlinecourses.science.psu.edu/stat505/lesson/13
[5] N. Cesa-Bianchi, "Funzioni kernel," *Metodi statistici per l'apprendimento*, 2017.
[6] M. E. Sargin, Y. Yemez, E. Erzin, and A. M. Tekalp, "Audiovisual synchronization and fusion using canonical correlation analysis," *IEEE Transactions on Multimedia*, vol. 9, no. 7, pp. 1396–1403, 2007.
[7] L. Muda, M. Begam, and I. Elamvazuthi, "Voice recognition algorithms using mel frequency cepstral coefficient (mfcc) and dynamic time warping (dtw) techniques," *arXiv preprint arXiv:1003.4083*, 2010.
[8] F. Ekman. (2002) Facs - facial action coding system. [Online]. Available: https://www.cs.cmu.edu/~face/facs.htm