Software per a llegir els llavis (Lip Reading)

Javier Alejandro Camacho Machaca

Resum—Aquest treball presenta el desenvolupament d'un programari capaç de llegir els llavis mitjançant el reconeixement facial. Amb aquest programari es busca facilitar la comunicació per a persones amb dificultats visuals o auditives, així com per a aquelles que vulguin saber què diu algú en una multitud o a una distància considerable. Per assolir aquest objectiu, primerament s'ha desenvolupat un model que prediu quina vocal està pronunciant una persona utilitzant la funció 'FaceMesh()' de la llibreria Mediapipe. Aquest pas inicial ha permès obtenir un programa que ajuda a entendre millor les característiques més importants de la captura de característiques facials. A continuació, s'ha creat el model que prediu quina paraula, dins d'un grup de paraules en anglès, està pronunciant una persona en un vídeo. Aquest model s'ha entrenat amb el dataset 'The Oxford-BBC Lip Reading in the Wild (LRW)' [10] utilitzant Convolutional Neural Network (CNN), per ser més precisos, capes de convolució 3D de la llibreria de xarxes neuronals Keras (tf.keras.layers.Conv3D) i una versió d'una xarxa neuronal recurrent (RNN) Long Short-Term Memory (LSTM). La convolució 3D ha demostrat oferir els millors resultats, gràcies també al preprocessament d'imatges realitzat.

Paraules clau—Lip Reading, Mediapipe, Face Mesh, detecció del rostre, detecció de la boca, CNN, CONV3D, Keras, RNN, LSTM.

Abstract—This work presents the development of software capable of lip reading through facial recognition. This software goals to facilitate communication for people with visual or hearing impairments, as well as for those who want to understand what someone is saying in a crowd or at a considerable distance. To achieve this goal, a model has been developed that predicts which vowel a person is pronounced using the 'FaceMesh()' function from the Mediapipe library. This initial step has allowed for the creation of a program that helps better understand the most important features of facial feature capture. Besides, a model has been created that predicts which word, within a group of words in English, a person is pronounced in a video. This model was trained with the 'The Oxford-BBC Lip Reading in the Wild (LRW)' dataset [10] using Convolutional Neural Network (CNN), to be more precise, 3D convolutional layers from the Keras neural network library (tf.keras.layers.Conv3D) and a version of a Long Short-Term Memory (LSTM) recurrent neural network (RNN). The 3D convolution has shown to provide the best results, also thanks to the image preprocessing performed.

Index Terms—Lip Reading, Mediapipe, Face Mesh, face detection, mouth detection, CONV3D, Keras, RNN, LSTM.

1 Introducció - Context del treball

És ben sabut que les persones amb dificultats o discapacitats visuals o auditives tenen més dificultats per interactuar amb altres persones. Això pot ser degut al fet que la resta de la gent pot no conèixer el llenguatge de signes, o perquè la persona amb la qual volen comunicar-se té dificultats per veure o bé no té capacitat visual. També cal considerar aquelles persones que han perdut la capacitat de comunicar-se o entendre als altres, com ara les persones que pateixen càncer de gola i perden la capacitat de parlar, com ha estat el cas de Val Kilmer (Figura 1). Un altre exemple és el cas de Bruce Willis, que pateix d'afàsia, una malaltia que provoca la pèrdua de la capacitat d'expressar-se o comprendre el llenguatge, com a resultat del dany a les àrees del cervell que controlen el llenguatge. Aquestes situacions poden ser causades per diverses malalties o accidents que canvien la vida de les persones per sempre.

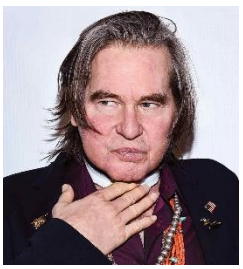

Figura 1. Val Kilmer, en el 2015 es va sotmetre a una traqueotomia i avui porta sempre mocador al coll.

Encara que ja hi ha diverses empreses i centres que han abordat aquest problema, nosaltres volem contribuir a aquesta causa amb el nostre esforç.

Per aquesta raó, hem desenvolupat un software que serà capaç de llegir els llavis mitjançant un reconeixement facial que se centrarà en les expressions dels llavis. Utilitzant xarxes neuronals RNN (Recurrent Neural Network) com el LSTM (Long Short-Term Memory) o capes de convolució 3D, processarem les dades d'un vídeo per convertir-les en una seqüència d'imatges, que serà el nostre veritable input i que serà utilitzat per predir el que la persona objectiu del vídeo està dient.

- *E-mail de contacte: 1566088@uab.cat*
- *Menció realitzada: Enginyeria de Computació*
- *Treball tutoritzat per: Coen Antens (Ciències de la Computació i Intel·ligència Artificial)*
- *Curs 2023/24*

Un cop el vídeo ha estat processat pel nostre model, tenim llibertat per ampliar aquest resultat de diverses maneres. Podríem utilitzar el nostre propi software per a nosaltres mateixos, en cas que no tinguem la capacitat de parlar, però sí de moure els llavis, podem transcriure la predicció del model a text i, gràcies a una API de "text-to-speech", expressar-ho a través d'un altaveu. Donant-li veu a les persones que no ho tenen.

També podríem aplicar aquesta última idea de "text-to-speech", però de manera inversa. És a dir, podríem utilitzar el nostre programari per transcriure el que s'està dient a text mitjançant una API "lips-to-text" (per al nostre cas). Això permetria que una persona es pugui comunicar amb algú que no pot escoltar, fent que pugui llegir el que estem dient. L'opció més viable a futur seria utilitzar les APIs de l'empresa OpenAI, les quals fan ús de la intel·ligència artificial.

Altres casos importants a tenir en compte, més enllà de qualsevol mena de discapacitat, són aquells en què hi ha molta gent i és difícil escoltar el que el teu company vol dir. En aquests casos, el nostre programa podria llegir els seus llavis. Un altre escenari seria quan vols comunicar-te amb una persona que està molt lluny, i per tant no la pots escoltar ni veure clarament. En aquest cas, si apliquem el nostre programari a una càmera amb una resolució suficient, podríem saber què vol dir aquella persona. També hi ha situacions que van més enllà de la necessitat personal, com ara els comentaristes esportius. Gràcies a la nostra aplicació, tant les persones que no entenen el que diuen per què parlen massa ràpid com les persones sordes, podran gaudir de qualsevol partit.

Per comprendre el problema que planteja la barrera de l'idioma a l'hora de parlar, ens centrarem en les expressions dels llavis per interpretar els seus gestos, ja que aquesta interpretació ens permetrà saber quin so fonètic està produint-se, i la fonètica és similar en els idiomes llatí i germànic. Per aconseguir-ho, entrenarem el nostre model amb un conjunt de dades de vídeos proporcionat per la mateixa BBC (British Broadcasting Corporation) [10]. Aquest conjunt de dades consisteix principalment en frases o paraules en anglès extretes de les transmissions de la BBC News. Va ser necessari sol·licitar permís abans de poder-ne fer ús.

2 Objectius

Fer un software inicial que classifiqui quina vocal estàs dient: Per començar a familiaritzar-nos amb les característiques més rellevants de les expressions dels llavis. Farem ús de la llibreria 'Mediapipe' per identificar els rostres de les persones i poder agafar punts de coordenades que seran els punts de referència que conformen la forma dels llavis. Aquesta informació l'utilitzarem per entrenar un model de 'Support Vector Machine' i un altre de 'Random Forest'.

Fer un software que detecti, dins d'un rang, que paraules estàs dient de manera aïllada: Aprofitant l'experiència i les dades recollides durant el desenvolupament del programa per a la detecció de vocals, iniciarem la creació d'un programa destinat a predir quines paraules està pronunciant una persona en un vídeo. Com aquesta tasca difereix considerablement de la simple detecció de vocals, ens orientarem cap a models de Deep Learning, especialment les RNN (Xarxes Neuronals Recurrents) i xarxes convolucionals, farem ús de la convolució 3D de la llibreria 'keras'. Aquests últims mètodes mencionats seran utilitzats per entrenar el nostre model objectiu, i processaran els frames del vídeo comentat anteriorment per a predir la paraula que està pronunciant la persona objectiva.

Desenvolupar el software per a la detecció de paraules amb capacitat per classificar un major nombre de paraules de manera més precisa i fluida: Aquesta millora implica enriquir el programa de detecció de paraules aïllades mitjançant un entrenament amb una base de dades més àmplia. Així, serà capaç de predir una gamma més gran de paraules i nosaltres desenvoluparem una forma de què ho faci d'una manera més continuada, permetent la formació de frases com a resultat. Aquest objectiu representa la fita ideal per a aquest treball, ja que abraça de manera integral els nostres objectius en aquest TFG.

3 Metodologia

L'organització i les eines utilitzades en la realització d'un projecte són crucials per aconseguir-ne el màxim rendiment en el temps disponible i per mantenir sempre una referència clara del que s'havia planejat en comparació amb la realitat. En aquest sentit, explicaré les opcions que he decidit utilitzar i les raons darrere d'aquesta elecció.

3.1 Mètode Àgil

La metodologia que hem triat és el Kanban, ja que en tractar-se d'un projecte realitzat per una sola persona, no cal seguir pautes tan estrictes com les que requeriria el mètode Scrum. El sistema Kanban permet gestionar les tasques en tres categories: "Per fer", "En curs" i "Fet". Aquesta metodologia és ideal per gestionar les tasques de manera senzilla i eficaç.

Per tant, farem servir aquest mètode àgil per organitzar les tasques més generals i rellevants d'una manera que sempre sigui clara quant al temps necessari per dur-les a terme i quines subtasques les componen, assegurant-nos que cada tasca es compleixi de la manera més exhaustiva possible.

3.2 Eina de seguiment

Per al seguiment i la gestió de les tasques necessàries per desenvolupar aquest software, hem optat per utilitzar Jira Software. Aquesta eina, a més de ser una de les principals en el seu àmbit, compleix tots els requisits necessaris per al nostre projecte. Ens permet disposar d'una interfície on

puc aplicar el mètode Kanban, gestionant les meves tasques en subtasques i assignant-les estats com "En curs" o "Finalitzat", al marge del sistema principal de Kanban. A més, em possibilita crear un calendari per establir terminis. Les funcionalitats que hem utilitzat de Jira Software són el "Tauler", que em permet visualitzar les meves tasques com a pòsits i moure'ls pels estats de Kanban (Per fer, en curs i fet). Les "Incidències", on puc veure totes les meves tasques i terminis per entendre-les millor, i on puc marcar-les com a "Finalitzat" o "Per fer". Finalment, el "Cronograma", que em mostra el temps dedicat a cada tasca i el temps restant per a completar-les. Però aquesta última funcionalitat no el vaig fer servir perquè tenia una estructura que no m'agradava molt, així que vaig utilitzar l'eina Excel per fer el cronograma del projecte (Fig. A. 3).

4 Estat de l'art

Aquest projecte comença amb una investigació i recerca de altres projectes o empreses que hagin realitzat alguna cosa similar a la lectura dels llavis. Els treballs que he pogut trobar que tenen com a objectiu final el Lip Reading són:

LipNet (End-to-End Sentence-level Lipreading) [1]: És un model que assigna una seqüència variable de fotogrames de vídeo a text, fent ús de convolucions espaciotemporals i una xarxa recurrent, entrenat totalment de manera integrada. Segons diuen ells, LipNet és el primer model de lectura de llavis a nivell de frase, integrat de principi a fi, que aprèn simultàniament de característiques visuals espaciotemporals i de models de seqüència.

LipType (A Silent Speech Recognizer Augmented with an Independent Repair Model) [2]: Aquest model es presenta com una versió millorada de LipNet, amb una major precisió i velocitat. Es destaca per un programa de preprocessament d'imatges que s'utilitzen com a entrada, incloent-hi la capacitat de modificar la il·luminació si és necessari, així com corregir els possibles errors en els resultats obtinguts. A més, el model s'implementa utilitzant la llibreria 'Keras' de 'TensorFlow'.

AV-HuBERT [3]: És un model autosupervisat centrat en l'AVS (Audio Visual Speech), és a dir, que té com a entrada tant l'àudio com la imatge. Tot i que l'objectiu és el mateix, la manera de fer-ho és diferent. Tanmateix, això no ha de ser una raó per no tenir-ho en compte a nivell de processament de dades.

Auto-AVSR [4]: És un model de codi obert dissenyat per a l'entrenament end-to-end. Similar al model anterior, aquest se centra en el ASR (Audio Speech Recognition) però també en el VSR (Visual Speech Recognition), sigui de manera individual o combinada. Utilitza la llibreria 'PyTorch' per a models d'entrenament.

Lipreading using Temporal Convolutional Networks [5]: És un repositori centrat en la lectura dels llavis que utilitza tant l'àudio com les imatges. El que diferencia aquest model de la resta és que ha estat entrenat per una xarxa basada en la convolució 3D i un preprocessament de les imatges del vídeo centrat en els llavis abans de l'entrenament.
Visual Speech Recognition for Multiple Languages [6]: En aquest repositori, com la majoria, s'utilitza de manera individual i combinada l'àudio i el vídeo per al seu entrenament. No obstant això, en aquest cas, es fa servir un model basat en ResNet-18 i un transformador de convolució augmentada (Conformer). Aquest enfocament representa un canvi dràstic en la manera d'afrontar el repte de la lectura dels llavis, ja que, per exemple, fa servir el Conformer en lloc d'una RNN (Recurrent Neural Network).

Tots aquests exemples són bons punts de partida per a iniciar un projecte centrat en el LipReading. Personalment, no seguiré o milloraré alguns d'aquests treballs, però m'han proporcionat idees sobre les eines, llibreries, models o mètodes de processament que puc desenvolupar per fer el meu propi projecte per ser un altre punt de referència en l'àmbit del LipReading.

En el meu cas, a més de ser un punt de referència més, vull ser-ho especialment per a totes aquelles persones interessades en el 'Lip Reading' i temes similars. Vull mostrar que es poden crear models de predicció d'aquest nivell amb les eines que tenim a l'abast i aprofitar-les al màxim.

5 Planificació

El primer gran repte o objectiu que es va proposar per a aquest projecte va ser desenvolupar un codi capaç de detectar els llavis d'una persona per determinar quina vocal està pronunciant. En primer lloc, vaig utilitzar la llibreria 'Mediapipe', específicament la funció 'face_mesh', per obtenir una malla del rostre. Un cop teníem tots els punts del rostre, ens vam centrar en els punts que formen els llavis per obtenir un mapa de punts que utilitzarem per entrenar un model d'Aprenentatge Automàtic. Posteriorment, vam seguir el mateix procés però fent servir un model de 'Deep Learning' per preparar-nos per a futurs reptes o objectius.
El següent pas és desenvolupar un software que em permeti identificar quina paraula individual està dient una persona. Per aconseguir això, hem de fer ús d'altres mètodes i eines més adequats a la nostra problemàtica. Hem considerat que, com una paraula és una seqüència de lletres amb significat, el millor seria utilitzar una xarxa neuronal recurrent, coneguda com a RNN (Recurrent Neural Network). Hem escollit aquesta opció perquè és un tipus d'algoritme de Deep Learning que s'implementa molt adequadament al nostre problema, ja que té com a input dades seqüencials o dades de sèries de temps. També farem ús de les capes de convolució 3D de la llibreria de Keras, ja que s'ha demostrat que és una eina molt eficient a l'hora d'entrenar models de predicció de vídeos. L'objectiu final i ideal seria aconseguir identificar les diferents paraules que formen una frase.

Per aconseguir desenvolupar aquest tipus de model, necessitarem una base de dades adequada. Hem tingut l'oportunitat de poder utilitzar el conjunt de dades 'The Oxford-BBC Lip Reading in the Wild (LRW) Dataset' [10],

que conté 1000 expressions de 500 paraules diferents, pronunciades per centenars de parlants diversos de la BBC. Farem servir aquest conjunt de dades per entrenar el nostre model i aconseguir els nostres objectius.

6 DESENVOLUPAMENT

Durant aquest tema, es detallarà el procés de desenvolupament, començant amb la recopilació i la preparació de dades, la implementació de models i algoritmes, i finalment, l'avaluació del sistema desenvolupat. Es discutiran els reptes trobats durant el desenvolupament i es presentaran els resultats obtinguts, destacant les contribucions significatives i les lliçons apreses en el procés.

6.1 Model de predicció de vocals

Per tal de concloure el meu primer objectiu d'elaborar un programari que classifiqui les vocals, he utilitzat la funció 'FaceMesh()' de la llibreria 'Mediapipe'. Aquesta funció, en primer lloc, detecta els rostres de les persones i després retorna una malla de punts del rostre expressada en coordenades i profunditat [x, y, z]. Ens centrarem únicament en els 80 punts que formen els llavis per tal d'obtenir el gest en si mateix (Figura 2).

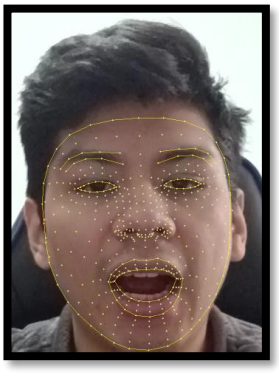

Figura 2. Foto feta per mi mateix i processada per la funció 'FaceMesh()'

Per poder generar una entrada per entrenar el model que implementi aquesta funció, he hagut de crear el meu propi conjunt de dades. Per fer-ho, he gravat diversos vídeos amb diferents distàncies i fons mentre vocalitzava les vocals. Utilitzant l'eina de reproducció de vídeos 'VLC media player', he convertit els fotogrames d'aquests vídeos en imatges i els he classificat segons la vocal que s'està pronunciant. Aquest conjunt de dades és on he implementat la funció i on hem extret els 80 punts que defineixen la forma dels llavis en cada imatge. L'estructura d'aquest dataset es pot veure en Fig. B. 2 de l'Apèndix.

Per obtenir un conjunt de dades adequat per a l'entrenament, afegiré un punt addicional que estarà situat al centre de la boca. Per aconseguir-ho, simplement calcularé la mitjana dels valors dels 80 punts existents, obtenint així el punt 81 també en format [x, y, z]. Per prescindir del format de llibreria dels punts i, així, poder dur a terme l'entrenament del model, substituiré les coordenades dels punts per la distància respecte al punt 81, que representa el centre de la boca. D'aquesta manera, en finalitzar aquest procés, disposarem només de 80 distàncies per a cada imatge que analitzem. A partir d'aquí, només queda entrenar el model.

Després d'haver dut a terme aquesta primera tasca, vaig adonar-me que les prediccions del meu model variaven segons la distància a la qual em trobava de la càmera. Quan estava lluny, només podia detectar les vocals 'O' i 'U', mentre que quan estava a prop, era impossible detectar aquestes últimes. Per resoldre aquest problema, vaig desenvolupar dues opcions. La primera consistia a ajustar les coordenades extretes de la funció 'FaceMesh()' en funció d'un marc que envoltava el rostre de la persona a la imatge, en lloc de la mida de la mateixa imatge. La segona solució, que va donar millors resultats, va ser tenir en compte la distància interpupil·lar, és a dir, la distància mitjana entre les pupil·les de les persones. Per fer-ho, vaig modificar el meu codi per extreure dos punts addicionals de la malla facial generada per la funció 'FaceMesh()', corresponents a la meitat de cada ull. Vaig calcular la distància entre aquests dos punts dels ulls de la mateixa manera que ho feia amb els 80 punts dels llavis al centre de la boca. A continuació, vaig dividir aquesta distància per la mida real de la distància interpupil·lar, obtenint una escala de mida real per píxel. Amb aquesta escala, vaig ajustar les 80 distàncies que representaven la distància al centre de la boca. D'aquesta manera, independentment de la proximitat o la distància a la càmera, les distàncies es van ajustant adequadament.

Els models d'aprenentatge que vaig utilitzar per a entrenar en aquest primer objectiu van ser el SVM (Support Vector Machine) i el Random Forest. Per avaluar la qualitat del

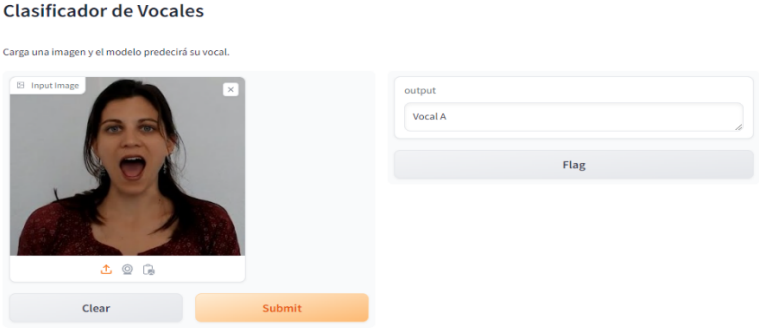

Figura 3. Imatge extreta del resultat de Gradio amb el model de reconeixement de vocals exportat utilitzant el model de SVM (Support Vector Machine). Selecciona una imatge i obtindràs com a sortida quina vocal està pronunciant.

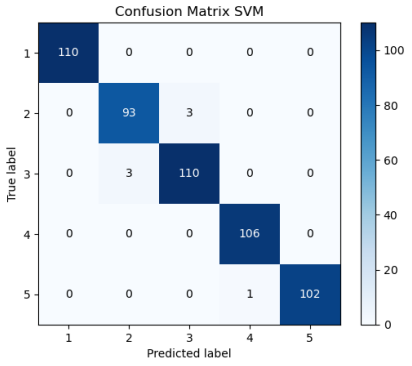

Figura 4. Matriu de confusió del model SVM, entrenat amb els paràmetres extretes del 'GridSearchCV()'

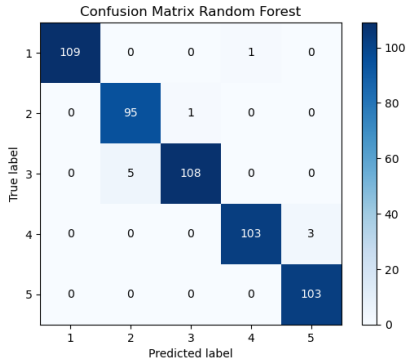

Figura 6. Matriu de confusió del model Random Forest

seu entrenament, vaig fer servir el 'GridSearchCV()' i la validació creuada ('Cross Validation') pel cas del SVM (Figura 4), mentre que pel cas del Random Forest vaig analitzar la seva precisió ('accuracy') en la predicció (Figura 6).

Per visualitzar el resultat d'aquesta primera tasca, farem servir la llibreria 'Gradio', que mostrarem a la Figura 3.
Gràcies a Gradio, podem crear fàcilment una interfície web per mostrar els resultats d'un model d'aprenentatge, aquestes característiques el converteix en una eina perfecta per a demostracions de les capacitats del model.
Per fer-ho més interessant i preparar-me per a l'objectiu final, també utilitzaré aquest mateix procés de creació del conjunt de dades com a entrada per a un model de xarxes neuronals. En aquest cas, faré servir les dades obtingudes mitjançant la metodologia que escala les distàncies en funció de la distància entre els ulls de les persones.
Com que l'objectiu és predir bàsicament 5 grups clarament diferenciats, faré servir la biblioteca 'Keras', que proporciona blocs modulars per al desenvolupament de models complexos d'aprenentatge profund. Utilitzaré aquesta llibreria per distingir entre diferents grups de dades que comparteixen similituds, com per exemple predir el tipus de flors en una imatge.
Després d'entrenar el model indicant-li que només hi ha 5 grups de dades diferents, vaig observar que les prediccions eren sempre les mateixes, és a dir, que el model només donava com a resposta una de les cinc vocals. Per avaluar la qualitat del model, vaig generar una Matriu de Confusió, que indica si les prediccions són correctes i en quina

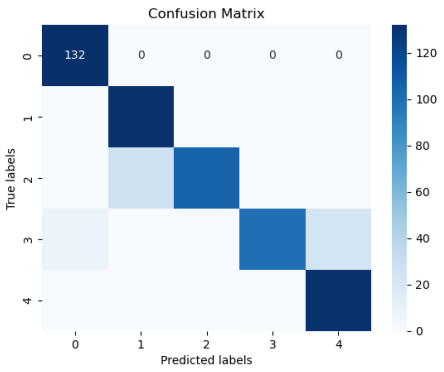

Figura 5. Matriu de confusió del model entrenat amb la llibreria 'Keras'

mesura fallen les prediccions errònies (Figura 5).

A primera vista, la matriu de confusió sembla prometedora pels seus resultats, però en provar-ho en casos reals, no proporcionava una resposta correcta en la majoria dels casos. Després d'intentar trobar la raó d'aquest problema, vaig reflexionar sobre la meva situació, i això em va fer adonar de dues coses. En primer lloc, per enfrontar-me al repte de predir paraules, hauria de replantejar el problema d'una manera diferent de com ho havia fet per predir vocals. En segon lloc, vaig veure la necessitat d'adquirir un conjunt de dades més gran i més adequat per evitar tants problemes durant l'entrenament del model. Per aquestes raons, vaig trobar les xarxes neuronals recurrents i el 'The Oxford-BBC Lip Reading in the Wild (LRW) Dataset' [10], que ja havia anomenat en la meva planificació. Vaig optar per aquestes opcions, ja que predir paraules de la mateixa manera que ho havia fet amb les vocals no seria ni adequat ni eficient.

6.2 Model de predicció de paraules en angles

Per començar amb l'objectiu de predir paraules individuals, primer he d'extreure les dades proporcionades per l'organització de la BBC. El dataset està compost per 500 carpetes, cadascuna corresponent a una paraula en anglès, i cada carpeta conté tres subcarpetes: una per a proves, una altra per a l'entrenament i l'última per a la validació. Dins d'aquestes subcarpetes hi ha vídeos, i cada vídeo té un arxiu '.txt' associat amb el mateix nom que el vídeo '.mp4', per a una millor comprensió de l'estructura mirar la imatge Fig. B. 1 de l'Apèndix.
Aquest arxiu '.txt' conté cinc línies amb informació rellevant, però l'única línia que ens interessa és l'última, que indica la durada de la paraula pronunciada per la persona del vídeo. Aquesta informació és crucial, ja que tots els vídeos del dataset tenen la mateixa durada exacta, 1,16 segons, o 29 frames. Gràcies a aquesta informació, podem calcular l'inici i el final exactes del fragment que volem extreure dins d'aquests 1,16 segons, pel fet que els vídeos contenen altres continguts a més de la paraula objectiu.

Començant a dur a terme aquesta tasca, va sorgir un problema, vaig adonar-me que els fragments de paraula que

volia extreure dels vídeos no sempre estaven exactament al mig, com assegurava la pàgina web del dataset. Aquesta situació complicava les coses, ja que significava que les paraules podien trobar-se més al principi o al final dels vídeos, situacions que no es podien detectar a simple vista.
Quan vaig utilitzar l'eina Audacity, que serveix per gravar o editar pistes d'àudio, per analitzar els vídeos i poder obtenir una resposta, em vaig adonar que els vídeos tenien una durada real d'1,21 segons, en lloc dels 1,16 segons que afirmava la pàgina web. Vaig arribar a la conclusió que ells també havien utilitzat la llibreria CV2 d'OpenCV per analitzar els vídeos i determinar la seva durada. Per fer-ho, cal utilitzar la funció 'cv2.VideoCapture()' en un vídeo, que permet obtenir els fotogrames per segon i el nombre total de fotogrames. Si utilitzes aquestes dues característiques per calcular el temps (frames_totals / fps), obtens 1,16 segons.

Un cop sabem que els vídeos tenen una durada real d'1,21 segons, podem extreure amb més precisió els clips objectius que ens serviran com a dataset real. Per exemple, si l'arxiu '.txt' del vídeo ens indica que la durada de la pronunciació d'una paraula és de 0,27 segons, només hem de restar aquesta durada dels 1,21 segons, resultant en 0,94 segons. Dividirem aquest últim temps en dos per obtenir el temps d'inici i final del clip objectiu: l'inici seria al segon 0,47 (0,94/2) i el final al segon 0,74 (0,47 + 0,27).
Al trobar la manera d'extreure els clips de qualsevol vídeo, ho vam aplicar a una col·lecció petita de paraules, ja que érem conscients que utilitzar les 500 paraules de cop faria que qualsevol prova o experiment trigués dies a realitzar-se. Així doncs, vam utilitzar les paraules [ATTACK, BLACK, FINAL, IMPACT, LATER, MEDIA, OFFICE, PRESS, SPEND, WEEKS]. Vam escollir aquestes perquè tenien un nombre de lletres molt similar i, el més important, diverses vocals i eren fàcilment reconeixibles en pronunciar-les. Cada una d'aquestes paraules comptava amb aproximadament 2200 vídeos: 100 per al test, 100 per a la validació i 2000 per a l'entrenament del model.
En extreure els clips de tots els vídeos de les 10 paraules en anglès anteriorment mencionades i transformar-los en frames, obtenim un total de 10.641 imatges per processar. Tot i que podríem utilitzar aquestes imatges per a l'entrenament, no seria una bona idea fer-ho directament. Per tant, realitzarem un preprocessament d'aquestes imatges per obtenir millors resultats en l'entrenament. En aquest preprocessament, farem un 'face alignment' i una extracció del requadre que conté la boca de la persona.

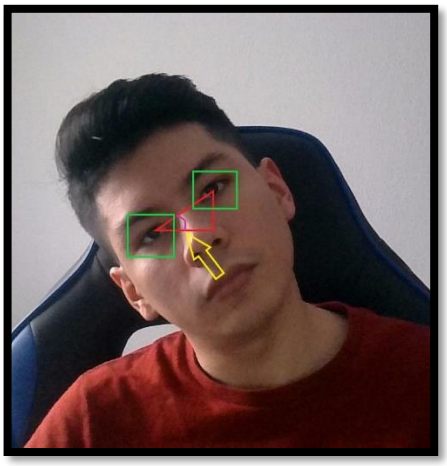

Figura 8. Foto feta per mi, la fletxa groga apunta a l'angle objectiu de color morat a calcular.

6.2.1 Preprocessament de les imatges

El primer pas del nostre preprocessament d'imatges, és fer un 'face alignment', que consisteix a alinear els ulls d'una imatge com el de la Figura 7.
Per aconseguir l'angle objectiu de referència en la Figura 8, primer hem d'obtenir les coordenades dels ulls respecte a la imatge. Després, calculem la diferència entre aquestes coordenades i utilitzem el resultat dins de la funció 'np.arctan2()' de la llibreria NumPy. Aquesta funció ens retornarà l'angle objectiu en radians, que convertirem a graus per obtenir l'angle de rotació necessari. No obstant això, no ens interessa només realitzar una rotació; necessitem efectuar una transformació que mantingui la cara centrada en la imatge i la modifiqui el mínim possible.

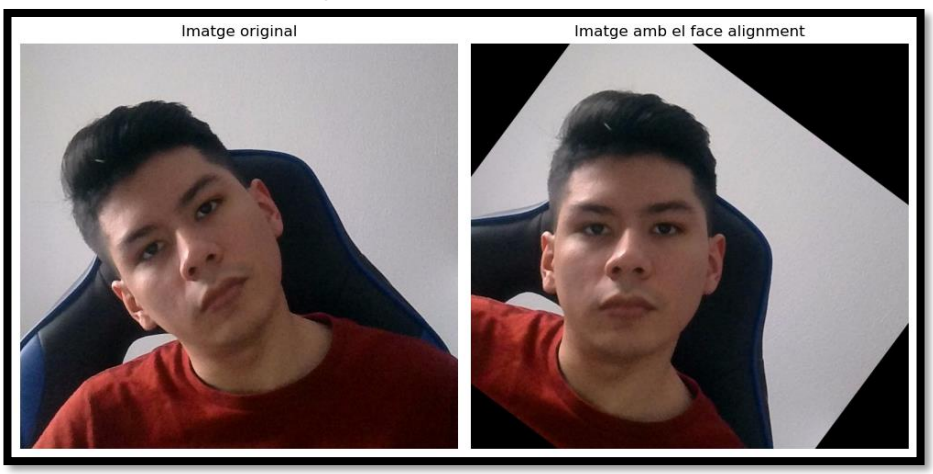

Figura 7. En la imatge de l'esquerra s'ha aplicat la funció per fer el 'face alignment', el resultat es mostra a la dreta.

Després d'obtenir l'angle de rotació, calculem el centre de rotació de la transformació mitjançant els punts mitjans de les coordenades dels ulls. Amb aquestes dades, utilitzem la funció 'cv2.getRotationMatrix2D(center, angle, 1.0)' per crear una matriu de transformació. El pas final del 'face alignment' és aplicar aquesta matriu de transformació amb la funció 'cv2.warpAffine(image, M, ...)'. Vaig descobrir com calcular-lo gràcies a una pàgina de 'Geeksforgeeks' que et donava un exemple de com fer-ho amb OpenCV i Python [11].

Vaig descobrir diverses maneres d'aplicar el procediment de 'face alignment', així que vaig decidir provar tres mètodes diferents i comparar-los pel temps que triguen, ja que aquesta característica és molt important per processar les 10.641 imatges. Si cada imatge requereix 1 segon, trigaríem gairebé 3 hores, i extrapolant això al dataset complet, ens ocuparia 150 hores.
El primer mètode utilitza la funció 'cv2.CascadeClassifier()' [12], que detecta el rostre i els ulls d'una persona en una imatge. Si utilitzem aquesta funció per obtenir les coordenades dels ulls i realitzar el procediment mencionat, el temps requerit és de 0,038 segons per imatge.
El segon mètode utilitza la funció 'dlib.shape_predictor("shape_predictor_68_face_landmarks.dat")' [13], que detecta el rostre d'una persona en una imatge i calcula els diferents punts de referència d'una plantilla que utilitza, incloent-hi els ulls. Utilitzant aquesta funció per calcular l'angle de rotació, el temps estimat és d'aproximadament 1,5 segons per imatge.
La tercera i última opció va ser una combinació dels dos mètodes anteriors. Del primer mètode vaig adoptar l'estructura per calcular l'angle de rotació, i del segon mètode vaig utilitzar la funció 'face_mesh()' de Mediapipe, que ja havíem emprat en el model de predicció de vocals. Aplicant aquesta combinació, el temps necessari és de 0,057 segons per imatge.

Taula 1. Temps que triguen a processar 1 imatge cada mètode.

	Cascade Classifier	Shape Predictor	Face Mesh
Segons	0,038	1,5	0,057

Tot i que el primer mètode, 'Cascade Classifier', és el més ràpid, vaig optar pel tercer mètode. La raó principal és que ja havia utilitzat aquesta funció anteriorment i la comprenia millor que el primer mètode.
Per completar aquesta primera etapa de preprocessament, apliquem el mètode de 'face alignment' utilitzant la funció 'face_mesh()' en tots els frames obtinguts de les 10 paraules en anglès. Els resultats d'aquest mètode aplicats a les imatges es poden veure a la Figura 7 (Per raons de privacitat, no puc mostrar imatges o informació que no doni la pàgina web del dataset LRW, per això utilitzo imatges de mi mateix per mostrar els exemples.).

En finalitzar aquesta primera fase del preprocessament, vam continuar amb la segona fase, el 'mouth extraction', que consisteix a extreure un requadre que contingui únicament la boca de la persona, evitant incloure altres elements, com el nas.
Aplicarem aquest procés a les imatges ja rotades en la fase anterior, utilitzant de nou la funció 'face_mesh()'. Aquesta funció proporciona diversos punts i les seves coordenades en diferents zones dels llavis. Per visualitzar quins punts identifica 'face_mesh()', es pot tornar a veure la Figura 2. Gràcies a aquests punts, podem determinar les coordenades mínimes i màximes de 'X' i 'Y', les quals coincideixen amb les cantonades del requadre que volem obtenir. Per exemple, la cantonada superior esquerra es correspondria amb la coordenada (x_min, y_min) i la cantonada inferior dreta amb la coordenada (x_max, y_max).

Un cop hàgim trobat la manera de calcular les coordenades del requadre, expandirem aquest requadre un 20% en tots els costats. És a dir, prendrem la mida del requadre en l'eix 'X' i calcularem el seu 20% per ampliar-lo tant per l'esquerra com per la dreta, i farem el mateix amb l'eix 'Y' però per a dalt i per a baix,aquest últim serà el que menys creixerà, ja que la boca és més llarga que alta. Un cop tinguem la mida del requadre final, redimensionarem el requadre perquè tots tinguin la mateixa mida de 100px en 'X' i '50px' en 'Y'. Vam escollir aquesta mida perquè és una mida mitjana entre la mida més gran de tots els requadres i el més petit dels dos eixos, de manera que, a l'hora de redimensionar, no es modifiqui massa. L'últim pas després de redimensionar és normalitzar les imatges. Aquest procés fa que l'entrenament del model sigui més efectiu, ja que permet que els models convergeixin més ràpidament, arribant a les millors característiques amb més rapidesa i

Figura 9. Procés pas a pas per fer el 'Mout Extraction'.

proporcionant una major estabilitat numèrica, prevenint així possibles problemes en reduir el rang dels números. Realitzem tot aquest procés en les 10.641 imatges que tenim i així obtenim el nostre dataset definitiu que utilitzarem per entrenar el nostre model. En la Figura 9 es pot veure el process pas a pas del 'Mout Extraction'.

6.2.2 Entrenament del model

Per a l'entrenament, hem decidit utilitzar dos tipus de xarxes neuronals: les 'Convolutional Neural Networks (CNN)', més específicament el 'CONV3D layer' i les 'Recurrent Neural Networks (RNN)', més específicament el Long short-term memory (LSTM). Aquestes dues arquitectures són les més adequades per desenvolupar models capaços de predir seqüències d'entrades, com en el nostre cas, una seqüència d'imatges.

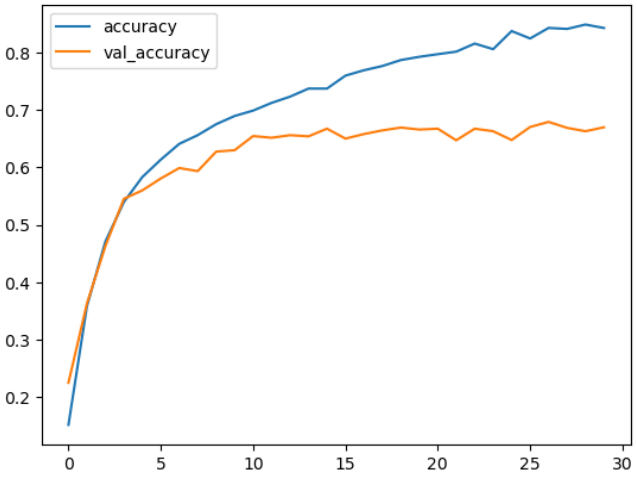

Figura 10. Mètriques de precisió en cada època de l'entrenament del model utilitzant 'Conv3D layer'.

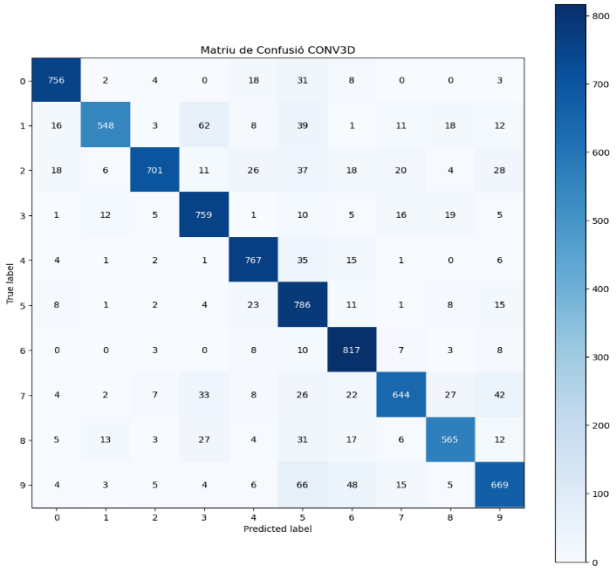

Figura 11. Matriu de confusió del model utilitzant 'CONV3D layer'.

Per començar a utilitzar la capa 'CONV3D' de la llibreria Keras, primer hem de crear el dataset mitjançant el preprocessament explicat anteriorment. Una vegada hem obtingut totes les imatges dels requadres de la boca, les hem dividit en conjunts d'entrenament i de validació amb la funció 'train_test_split()'.
A continuació, hem dissenyat l'estructura del model per ajustar-se a les nostres necessitats específiques: els inputs consisteixen en seqüències de fins a 12 frames per clip, cada frame amb una resolució de 50px d'altura per 100px d'amplada i amb 3 canals RGB. El model està configurat per predir 10 classes diferents. Amb aquestes especificacions, l'arquitectura de la xarxa neuronal queda com es mostra a la Fig. C. 1 de l'Apèndix.
Després d'entrenar aquesta estructura durant 30 èpoques, hem obtingut una precisió del 85,34% i una precisió de validació del 66,98% (Figura 10) per a les 10 classes. Per a una visualització més clara de l'eficàcia del model, la Figura 11 mostra la matriu de confusió.

Seguint aquest entrenament, vam procedir a utilitzar el 'LSTM', amb una estructura bastant similar al 'CONV3D layer'. També vam haver d'especificar un nombre màxim de frames, les seves dimensions i el nombre de classes a predir. Seguint aquest entrenament, vam passar a utilitzar el 'LSTM', amb una estructura bastant semblant a la del 'CONV3D layer'. Vam especificar un nombre màxim de frames, les seves dimensions i el nombre de classes a predir. Això va donar com a resultat una arquitectura similar a la que es mostra a la figura. Desafortunadament, en entrenar aquest model amb LSTM, els resultats no han estat satisfactoris, ja que la precisió i la validació han girat al voltant del 10%.

7 Conclusió

Aquest projecte s'ha conclòs amb dos models: un model que prediu quina vocal pronuncia una persona utilitzant el 'Support Vector Machine' i la funció 'face_mesh()' de la llibreria Mediapipe per extreure les característiques d'un rostre humà; i l'altre model, amb el qual es va iniciar aquest treball, que prediu quina paraula està dient una persona mitjançant una seqüència d'imatges d'un vídeo. Per a aquest últim model, vam haver d'extreure un clip de cada vídeo del dataset proporcionat i aplicar-hi un preprocessament, com s'ha explicat anteriorment, per millorar l'entrada a l'hora d'entrenar el model.

Vam considerar dues metodologies per a l'entrenament del model predictiu de paraules i vam optar per la que va donar millors resultats, que va ser el 'CONV3D layer'. Els resultats dels dos models finals obtinguts han estat acceptables, amb possibilitats de millora. Com que només hem tingut en compte 10 paraules del conjunt de 500 del dataset, el procés que hem seguit es pot aplicar perfectament a les 490 restants. I amb l'enfocament

d'aquest projecte, també es pot explorar la possibilitat d'entrenar el model amb altres idiomes, ja que ens hem centrat en les característiques dels llavis, que és independentment del so o de l'accent que puguin tenir les persones. És important destacar que la longitud de la paraula, tant en nombre de lletres com en durada, no suposa cap problema per al nostre model, perquè només caldria ajustar el nombre màxim de frames a l'entrada per a això.

Espero que aquest projecte pugui ajudar a proporcionar un punt de partida viable i menys complicat per a aquelles persones interessades a iniciar-se en el 'Lip Reading' o en altres tipus de processament o entrenament similar al que hem realitzat.

AGRAÏMENTS

Vull expressar el meu més profund agraïment al meu tutor, Coen Antens, per la seva guia i suport constants durant tot aquest trajecte. La seva disponibilitat per reunir-se i els seus consells en moments d'incertesa han estat inavaluables. També vull agrair a la meva família més propera i a la meva parella per ser sempre un punt de suport ferm.

BIBLIOGRAFIA

[1] Yannis Assael, Brendan Shillingford, Shimon Whiteson and Nando de Freitas (2016, Novembre) 'LipNet' [Online] URL: (https://arxiv.org/abs/1611.01599)

[2] Inclusive Interaction Lab (2021, Gener 12) 'LipType' [Online] URL: (https://github.com/theiilab/LipType)

[3] Meta Research (2022, Gener 6) 'AV-HuBERT' [Online] URL: (https://github.com/facebookresearch/av_hubert)

[4] Pingcuhan Ma (2023, Juny 16) 'Auto-AVSR: Lip-Reading Sentences Project' [Online] URL: (https://github.com/mpc001/auto_avsr)

[5] Pingcuhan Ma (2020, Juny 25) 'Lipreading using Temporal Convolutional Networks' [Online] URL: (https://github.com/mpc001/Lipreading_using_Temporal_Convolutional_Networks)

[6] Pingcuhan Ma (2022, Març 1) 'Visual Speech Recognition for Multiple Languages' [Online] URL: (https://github.com/mpc001/Visual_Speech_Recognition_for_Multiple_Languages)

[7] OMES (2021, Maig 26) 'Malla Facial (MediaPipe Face Mesh) | Python - MediaPipe – OpenCV' [Online] URL: (https://www.youtube.com/watch?v=TCUi-pOXuCBQ&list=PLcz5q4ASTYQGKfBzjFBX5nVPkv-TUYn2Ro&index=3)

[8] Nachiketa Hebbar (2021, Abril 20) 'Image Classification Web App in Python| Keras +Gradio' [Online] URL: (https://www.youtube.com/watch?v=aZ4wV4V_p9E)

[9] Spanish Pronunciation Academy (2017, Maig 24) 'Los sonidos del español: Vocal A' [Online] URL: (https://www.youtube.com/watch?v=E7PKLMBAUtk)

[10] J. S. Chung, A. Zisserman (2016) 'The Oxford-BBC Lip Reading in the Wild (LRW) Dataset' [Online] URL: (https://www.robots.ox.ac.uk/~vgg/data/lip_reading/lrw1.html)

[11] Geeksforgeeks (2023, Març 22) 'Face Alignment with OpenCV and Python' [Online]URL: (https://www.geeksforgeeks.org/face-alignment-with-opencv-and-python)

[12] Paul Viola and Michael Jones (2001) 'Cascade Classifier' [Online] URL: (https://docs.opencv.org/3.4/db/d28/tutorial_cascade_classifier.html)

[13] Italo José (2018, Juny 19) 'Facial landmarks recognition' [Online] URL: (https://github.com/italojs/facial-landmarks-recognition)

[14] Lonnie (2023, Febrer 23) 'Video Classification' [Online] URL: (https://www.kaggle.com/code/lonnieqin/video-classification/notebook)

APÈNDIX

A. DIAGRAMA DE GANNT

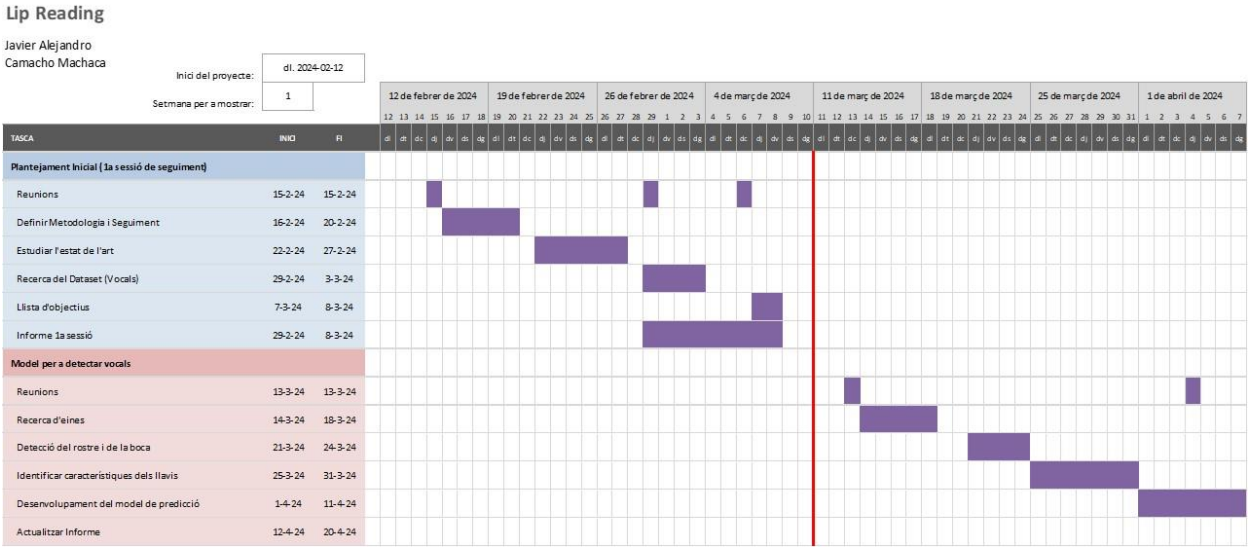

Fig. A. 3. Primera part del diagrama de Gantt, on les línies verticals vermelles són el límit de cada entrega del seguiment

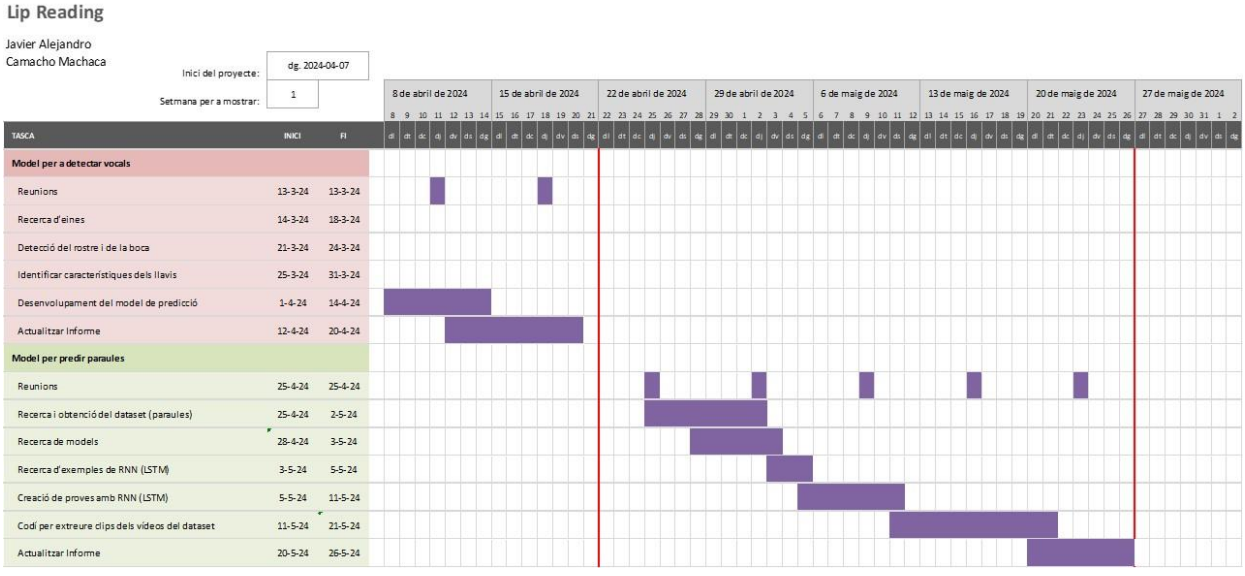

Fig. A. 1. Segona part del diagrama de Gantt, on les línies verticals vermelles són el límit de cada entrega del seguiment.

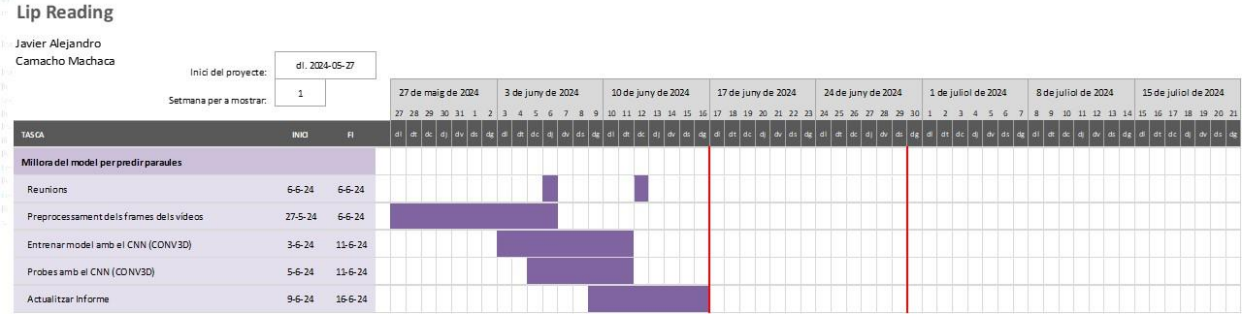

Fig. A. 2. Tercera part del diagrama de Gantt, on les línies verticals vermelles són el límit de cada entrega del seguiment.

B. ESTRUCTURA DELS DATASETS

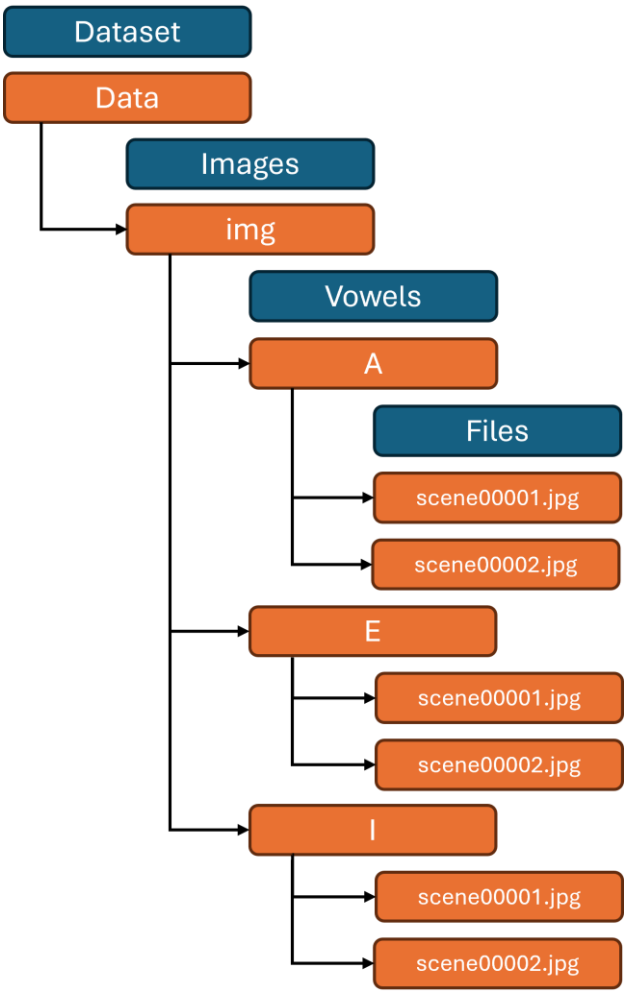

Fig. B. 2. Estructura del dataset per al model de predicció de vocals.

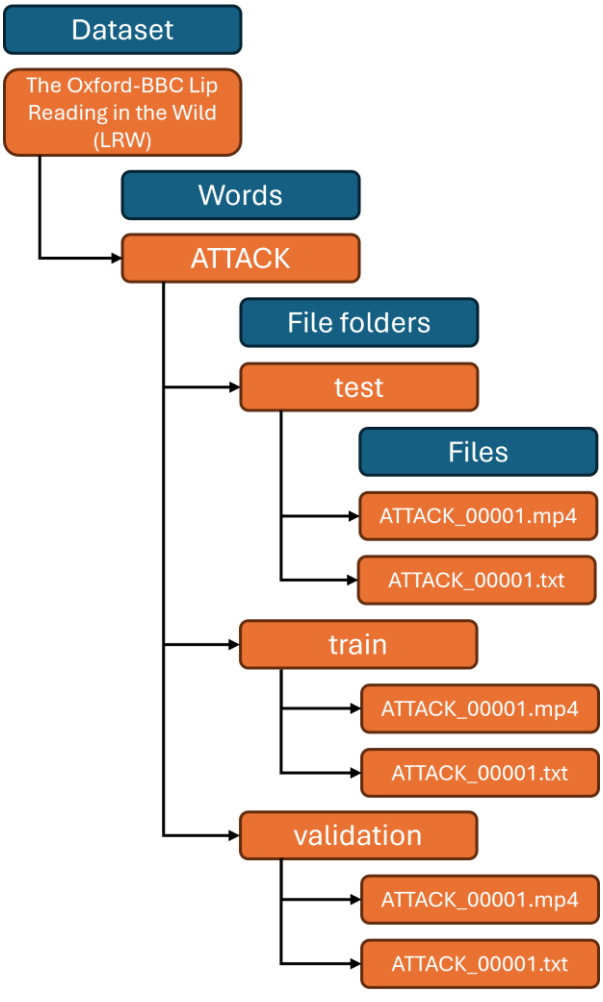

Fig. B. 1. Estructura del dataset per al model de predicció de paraules en angles.

C. Arquitectura dels models de xarxa neuronal.

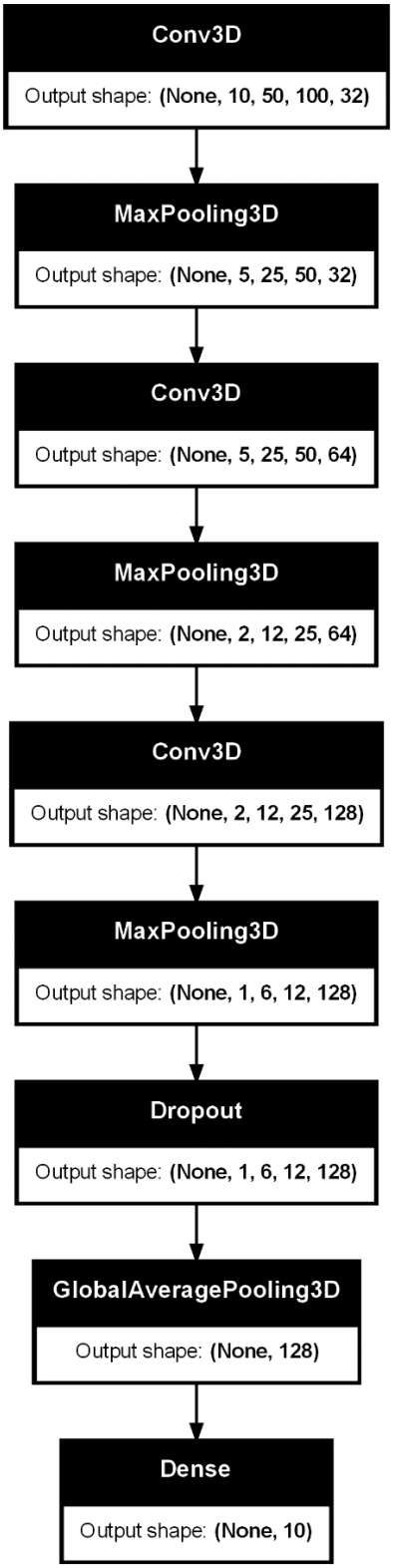

Fig. C. 1. Arquitectura del model amb CONV3D layer.

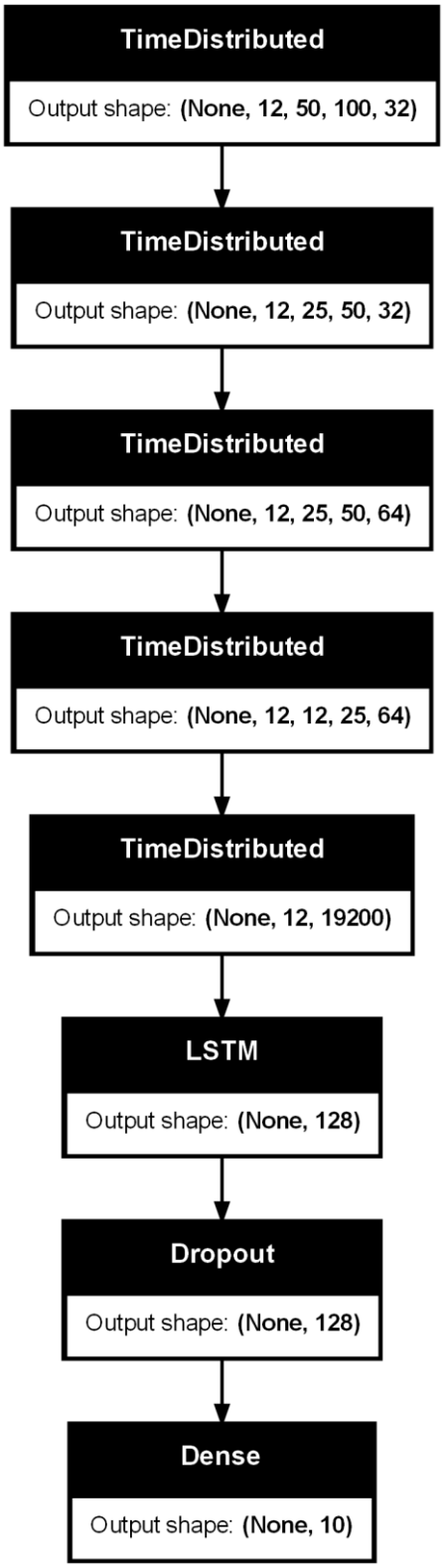

Fig. C. 2. Arquitectura del model amb LSTM.