Méthode par apprentissage de lecture sur les lèvres CNN+LSTM

Timothé Mazard

Département de génie logiciel et des TI

Ecole de technologie Supérieure

Montréal, Canada

timothe.mazard.1@ens.etsmtl.ca

***Abstract*—This electronic document is a "live" template and already defines the components of your paper [title, text, heads, etc.] in its style sheet. **CRITICAL: Do Not Use Symbols, Special Characters, Footnotes, or Math in Paper Title or Abstract.* (*Abstract*)**

Keywords—LSTM, CNN, Haar Cascade, Jpg, ImageNet

I. Introduction

Ce projet d'apprentissage profond a pour but de mettre en valeur le fait qu'une machine est capable par visualisation d'interpréter les mouvements des lèvres. Cette méthode se base sur l'assemblage d'un réseau de neurones (CNN) et réseau récurrent un LSTM où on retrouve la connexion de retour d'information. Ce dernier peut traiter aussi bien les données isolées telles des images ou un enregistrement audio, d'une séquence ou une vidéo au complet. Cette application sera de manière générale destinée à la reconnaissance vocale ou bien à la reconnaissance d'écriture manuscrite. Il y sera question dans ce projet de plusieurs grandes phases. Tout d'abord, le traitement de données est un point crucial car ici nous traitons des images. Par la suite une phase de test est mise en place en passant par le transfert d'apprentissage, nous reviendrons sur ce point ultérieurement.

II. Contexte

Pour ce qui est des données, celles-ci ont été extraites sous format mp4 par un smartphone. Une meilleure qualité et une meilleure extraction des caractéristiques des lèvres ce sont fait ressentir. Les caractéristiques des lèvres sont au format jpg. Nous voulons être en mesure de pouvoir authentifier les chiffres allant de 0 à 9 avec la meilleure précision. Mais cela ne veut pas dire que cette méthode s'arrête ici. Ce domaine voudrait faire en sorte que des informations en temps réel soient analysées afin de faire de la retranscription. Selon, l'Organisation Mondiale de la Santé plus de 5% de la population souffre de mal audition. Les solutions auditives médicales ne sont pas en mesure de résoudre entièrement les problèmes de ces personnes. Dans de telles situations la reconnaissance vocale par l'articulation des lèvres est un moyen de contourner ce problème. Il sera peut-être plus tard de traduction vocale d'une langue à une autre en temps réel.

D'après l'article de Garg et al [5], ils comparent différentes méthodes afin de voir les plus adéquates à ce genre de problème. Dans l'une des méthodes, ils utilisent un LSTM pour extraire l'information de temporalité mais les résultats ne sont pas relevés . Ils pensent que c'est dû au manque d'entraînement en plus du fait que celui-ci soit entraîné après les cartes de caractéristiques. Cependant ici nous allons nous servir de MobilNet qui est un modèle beaucoup plus compact et donc demande moins de temps d'apprentissage. Un espoir de lueur se fait cependant ressentir dans l'article de Pyateava et Dzyuba [4], le modèle comporte un CNN basé sur MobilNet et un LSTM. Ils obtiennent un score moyen de 68% de précision basé sur sept mots en russe. Nous pourrions voir si celui-ci performe mieux que celui avec VGG d'autant plus que nous ne connaissons pas le modèle VGG utilisé dans l'article précédent.

III. Objectifs

Les objectifs ont évolué depuis la proposition de projet. Nous allons créer de A à Z un environnement, en d'autres termes nous avons créé de zéro un dataset, mais aussi bien le modèle utilisé. Les objectifs portent aussi bien la pré-modélisation que la post-modélisation

A. Création du dataset

Les objectifs sont toujours à l'ordre du jour, le traitement des vidéos afin d'isoler les lèvres se fait par la méthode de Viola Jonas qui utilise les caractéristiques de Haar, figure 1 dans l'annexe. Cette méthode extrait des caractéristiques sans l'utilisation de ressources abondantes, algorithme facile à mettre en place. Grâce à ces filtres il est donc possible en passant par la librairie Opencv d'extraire la région d'intérêt de la bouche. Une image représentative se trouve en annexe. Plusieurs images sont extraites et cela a pour but que le LSTM garde une trace temporelle des informations et puisse en juger sur le choix de la classe à laquelle cette séquence d'images appartient en se basant uniquement sur les informations

visuelles et non plus sur les informations auditives comme il en était encore le cas dans la proposition de projet.

B. Modélisation

Le deuxième objectif porte sur l'entraînement d'un modèle ainsi que sur les résultats. Pour ce qui est du modèle, celui-ci va être pré entrainé sur Imagenet. Base de données regroupant des centaines de milliers d'images de toutes sortes allant de l'humain aux roues de vélo. Nous allons concevoir un modèle basé sur MobilNet puis un LSTM prend place après les couches denses du CNN, annexe 3. Le transfert learning se fait sur les couches denses, seulement cette dernière couche est entraînée. Nous jugeons que nos images sont assez similaires aux images présentent dans la base de données ImageNet.Par ailleurs nous avons très peu de données disponibles ce qui privilégie la piste de l'apprentissage par transfert. Cela permettra également d'avoir un gain sur le temps d'entraînement qui sera moins conséquent. Cependant si des courbes anormales se font ressentir dans les données de pertes entre la validation et d'entraînement alors il sera à reconsidérer les données d'entraînement comme non similaires aux données originales c'est-à-dire celles présentent dans ImageNet. Cela engendrera un apprentissage non pas seulement des couches denses de MobileNet, évoquant la phase de classification mais aussi les quelques dernières couches du réseau à convolution. Pour affiner l'apprentissage au type de données traités.

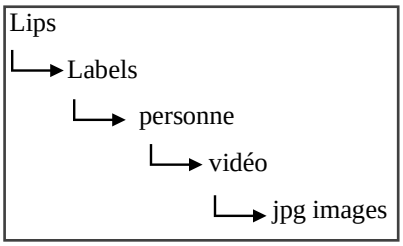
Afin de déterminer si le modèle a bien appris les classes. Il sera plus judicieux que d'utiliser une simple méthode de précision il est plus adéquate de partir sur des score WER et/ou BLEU qui sont des mesures de scores de prédiction de bon assemblage des mots afin de former une phrase. Pour ce projet il est question de mots, il nous vaudra vérifier s'il est possible d'ajuster ces méthodes pour résoudre notre problème. Mais ces métriques répondent à la façon dont un humain évaluerait le même texte, le constat sera beaucoup plus représentatif de la vérité sur la prédiction de phrase.

IV. MÉTHODES

Dans cette partie il sera question de la collection des données car ces données ne proviennent pas de données trouvées dans des bases de données mais bien des données extraites à partir de mon smartphone. Je me suis rendu compte que la qualité de la caméra embarquée dans l'ordinateur ne suffisait pas à la collection des images des lèvres, collection de zone d'intérêt aléatoire, non représentatif des lèvres.

Pour la collecte des données, nous utilisons des données vidéos pré-enregistrées mais il est tout à fait possible d'utiliser directement des séquences d'images provenant d'autres sources comme CNN ou base de données dans ce domaine. J'ai voulu choisir mon propre jeu de données car je voulais avoir des classes spécifiques, suite des chiffres de 0 à 9. D'ailleurs, il est difficile de se procurer réellement des données comme des séquences d'images de lèvres. Néanmoins, il est tout à fait possible de créer son jeu de données à partir de ces vidéos. Environ 7 vidéos par classe sont créées pour chaque individu. Soit 140 vidéos au total, pour le moment seulement deux visages de personnes ont été filmés. Ces vidéos ont été enregistrées au format MP4, le format AVI avait été suggéré mais aucun codec vidéo installé de base sur windows 10. Par la suite, la méthode de cascade de HAAR est utilisée afin d'extraire la région d'intérêt qui est la bouche. Cette méthode est incluse dans la librairie de OpenCV. Elle utilise le classifier de bouche, se composant des caractéristiques de HAAR nécessaire à la détection de cette dernière. Des méthodes de 'Backproject' et de 'Meanshift', ce sont des techniques qui nous ont permis d'éviter les erreurs de détection de zone d'intérêt. Sans rentrer dans les détails, la première méthode permet de faire ressortir les caractéristiques saillantes d'une image et l'autre de pouvoir corriger le pas de déplacement ce qui minimise le fait que la zone d'intérêt initialement sur la bouche passe sur l'œil. L'œil ayant des caractéristiques similaires que la bouche pour la méthode de HAAR.

Ces images sont enregistrées dans le format jpg, un format compressant les données mais conçu pour le traitement de données. Les images des lèvres sont enregistrées dans un dossier avec une taille de 64 x 64 afin de pouvoir passer en images d'entrées de MobilNet. Cependant, il a été relevé sur le site officiel de Keras[8] que mobilNet supporte toute taille d'image supérieure à 32 x 32, en revanche des tailles plus grandes offrent de meilleurs résultats. Pour être dans une optique de performance nous avons redimensionné les images en taille 128*128 en essayant de conserver au mieux les détails avec une interpolation cubique. En tout cela ne fait pas moins de 30 500 images enregistrées qui vont être passées en revue par le CNN+LSTM. Les données sont ensuite stockées dans cette hiérarchie.



Ensuite vient la tâche de création de jeu de données pour se faire nous utilisons ImageDataGenerator de Keras pour générer les données avec les images d'entrées en spécifiant la taille des images d'entrées mais aussi la taille du batch. Nous utilisons Image DataGenerator pour modifier les images d'entrées avec des rotation, des cisaillements et des flips haut/bas, gauche/droite. Cela permet de fortifier l'apprentissage des données d'entraînement. Or avec notre modèle les résultats n'étaient pas concluants. Le code est fourni en Annexe pour la création du Dataset. Les labels sont enregistrés au format One Hot c'est-à-dire une représentation vectorielle binaire. Si nous avons 3 labels, le label 2 est au format [0,1,0].

Le modèle a été créé en suivant la lecture [4] de Pyataeva et Dzyuba. C'est-à-dire qu'au modèle de base MobilNet nous rajoutons une couche Dense de 1024 suivit d'une autre couche Dense de 128 avec pour activation ReLu. Cette couche sera l'entrée du LSTM. Qui lui va faire un lien temporel entre chaque image de la vidéo et cela grâce aux connexions faites avec les données d'images précédentes. De plus, il a été démontré dans [3] développé par Karan Shrestha que

l'augmentation du Dropout procure de meilleur mais aussi la taille du kernel. Les couches Denses ont été testées avec deux différentes valeurs de Dropout 0.3 et 0.5.

Après cette phase de modélisation vient la visualisation des résultats pour se faire les pertes ainsi que la précision du modèle pourront être affichés. Pour ce qui est de la visualisation classe par classe, il a été fait appel à une matrice de confusion, montrant la répartition des données sur chaque classe. C'est un moyen de mesure de performance, qui peut être amené à être vu lors de phase de présentation des résultats.

Mais aussi nous verrons d'autres outils de performance, plus à même d'établir une idée claire sur cette tâche de prédiction de mots, qui sont les score WER et BLEU. Relevant non pas des scores de TP, FP, TN et FN mais plutôt de comment le modèle a pu identifier le mot et ce avec quel degré d'exactitude.

Le WER [7] est la méthode la plus utilisée dans la reconnaissance vocale. Il mesure la différence entre deux mots. Il prend en considération aussi bien les mots insérés, supprimés que substitués. Si nous fonctionnons avec un seul mot il sera alors question du score CER score sur les lettres qui agit avec cette même méthodologie.

$$WER = \frac{Inserted + Deleted + Substitued}{Total\ words\ in\ script}$$

V. RÉSULTATS

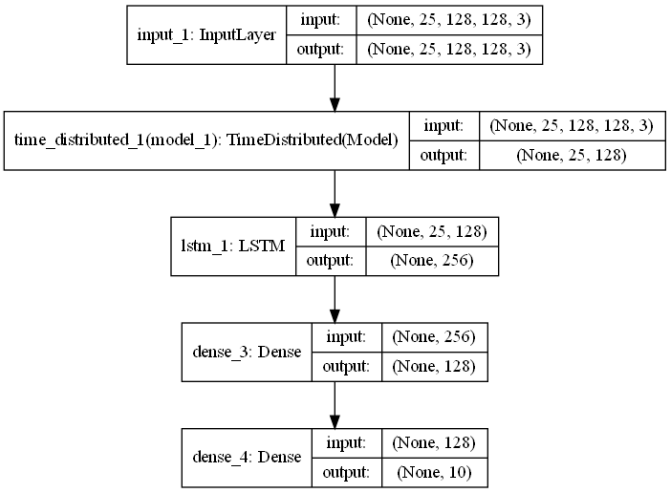
A. *Extractions de données*

En matière d'extraction de données les résultats escomptés sont acquis. Par l'algorithme développé par Viola Jones, les ondelettes de Haar extraient les caractéristiques locales de la bouche par la localisation des traits saillants. Bien entendu, l'utilisation d'une vidéo à rendu la tâche de prétraitement plus complexe, il a été découvert que le clignement des yeux peut être confondu avec la bouche faussant la détection de la zone d'intérêt de la bouche. Il a été question de faire appel à des méthodes de Meanshift et de BackProject pour maintenir la zone d'intérêt de la bouche. Finalement, toutes les données ont pu être extraites sans données aberrantes. Il est vrai que parfois les lèvres ont été quelque peu rognées mais sans grande incidence sur la visualisation de l'articulation de la bouche.

B. *Modélisation*

Nous avons ensuite conçu un modèle sur la base de ce qui a été présenté par A. D. Anna Pyataeva, pour se faire nous avons utilisé MobilNetV2 à qui nous avons ajouté un LSTM en sortie afin de parvenir à la tâche de classification pour la lecture sur les lèvres. Le modèle traite en sortie du CNN, la notion de temps, de succession de données. Les séquences d'images se composent de 25 images les unes à la suite des autres. Le LSTM va ensuite traiter les séquences une à une, il va ensuite interpréter les caractéristiques saillantes dans une dimensionalité temporelle pour comprendre le « mécanisme » de prononciation des différentes classes.

Voici une vue représentant le modèle dans son ensemble.



L'analyse des résultats comportera l'ajustement de quelques paramètres. La première analyse a été faite avec uniquement les données originales avec un Dropout de 0,3.Une deuxième analyse avec un Dropout de 0,5. Une dernière analyse est faite avec un Dropout de 0,5 et une augmentation de données. La base de données a subi des modifications, il est question du mélange de celle-ci, cela aide le modèle à mieux apprendre. D'ailleurs les données ont été séparées en données d'entrainement et de test avec un répartition de 80% pour les données d'entrainement et de 20% pour le test. Il y a eu également un séparation du dataset d'entraînement, dont 20% représente les données de validation, permettant de valider les données d'entraînement par la visualisation des pertes et de la précision du modèle.

C. *Visualisation des résultats*

Les résultats se présentent sous la forme des scores de performance et de pertes du modèle.

Pour la première analyse le score ne s'élève pas au-dessus de 40%, comme le montre le graphe à gauche les performances relevées sont en dessous de ce que proposent les autres auteurs. En effet, nous constatons que le modèle apprend vite mais que lors de la validation les données semblent être mal interprétées.

En suivant l'idée de Karan Shrestha [3], le fait d'avoir un dropout plus élevé minimise l'impact des premiers batchs de données sur le modèle. Il permet également d'éviter le sur apprentissage.

Le score ici atteint au plus 60% ce qui est bien meilleur et talonne ce qui a pu être observé dans d'autres articles.

Une augmentation des données permettant de doubler celle-ci en passant de 140 dataset à 280 dataset. L'augmentation a eu recours à l'inversion horizontale et à une rotation de plus ou moins 10°. Nous pensions que cela permettrait d'éviter le sur-apprentissage.

En observant le premier graphe, avec un Dropout de 0,3 lors de la phase de validation les pertes et la précision stagne. Alors que sur le graphe b) les pertes diminuent et la précision quant à elle augmente. En matière de sur-apprentissage il y a une petite évolution, avec le Dropout de 0.5 le

surapprentissage se fait ressentir à partir de la 15eme époque alors qu'avant c'était à partir de la 10eme époque.
L'augmentation qui a eu recours à un dropout de 0.5, en graphe c) n'a pas porté ses fruits, comme l'atteste le graphique. Le score de performances semble beaucoup plus aléatoire entre chaque époque d'entraînement.

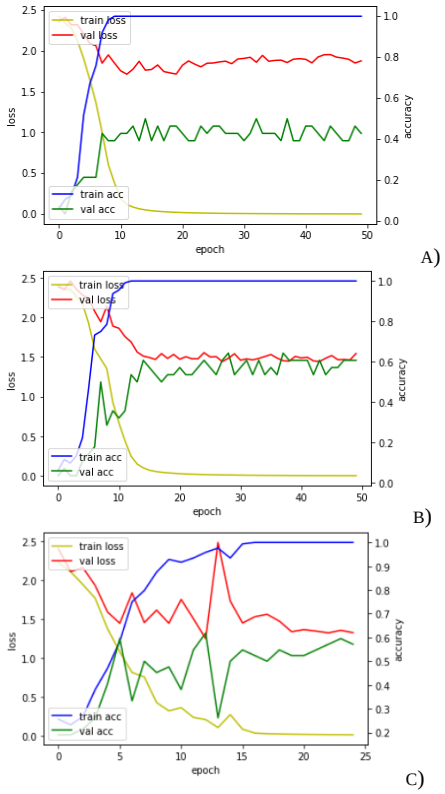


Fig. 1. Score de performance et pertes selon les époques d'entraînement

Afin d'être en mesure de visualiser l'impact de chaque classe sur le modèle, ce dernier a été testé et a fait une prédiction sur la base de données de test. Le score F1 est le score le plus représentatif de chaque classe. Car il est une moyenne de la précision et du rappel. On remarque que certaine classe on un bon score comme la classe 5 et 2. Néanmoins certaines comme les classes 9 et 4 sont mals prédites.

La matrice de confusion atteste que la classe 9 est mal prédites car peut de données sur la diagonale, stipulant normalement une bonne prédiction. De manière générales les données se situe assez bien sur la diagonale.

```
[9 4 9 2 8 5 3 6 7 3 6 8 9 0 2 3 3 6 6 2 4 3 0 0 4 4 5 7 4 9 0 8 0 9 2]
[9 1 8 2 9 5 3 6 7 1 6 8 3 0 2 3 3 6 6 2 4 3 0 0 4 6 5 2 1 7 1 8 1 6 2]
              precision    recall  f1-score   support

           0       1.00      0.60      0.75         5
           1       0.00      0.00      0.00         0
           2       0.80      1.00      0.89         4
           3       0.80      0.80      0.80         5
           4       1.00      0.40      0.57         5
           5       1.00      1.00      1.00         2
           6       0.67      1.00      0.80         4
           7       0.50      0.50      0.50         2
           8       0.67      0.67      0.67         3
           9       0.50      0.20      0.29         5

    accuracy                           0.66        35
   macro avg       0.69      0.62      0.63        35
weighted avg       0.78      0.66      0.68        35
```

Fig. 2. Score F1 par classe

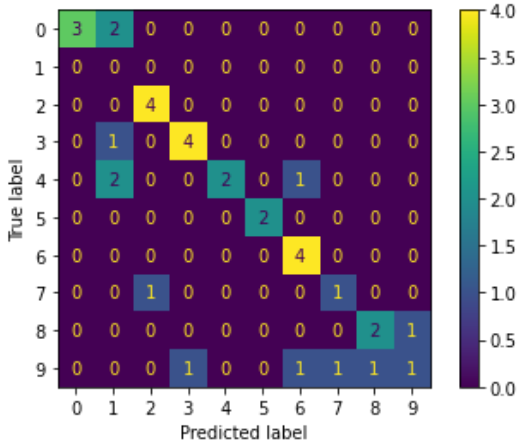


Fig. 3. Matrice de confusion

Enfin nous nous sommes penchés sur l'impact de l'élocution sur les performances du modèle, nous avons effectué les mêmes tests que précédemment mais de manière séparée. Ce test n'a en effet relevé aucune différence distincte entre les deux jeux de données, comme le peuvent le souligner les graphes de performances ainsi que les matrices de confusion. D'ailleurs les éléments sur la matrice de confusion de la personne ayant une élocution rapide sont plus situés sur la diagonale, soulevant une meilleure performance.

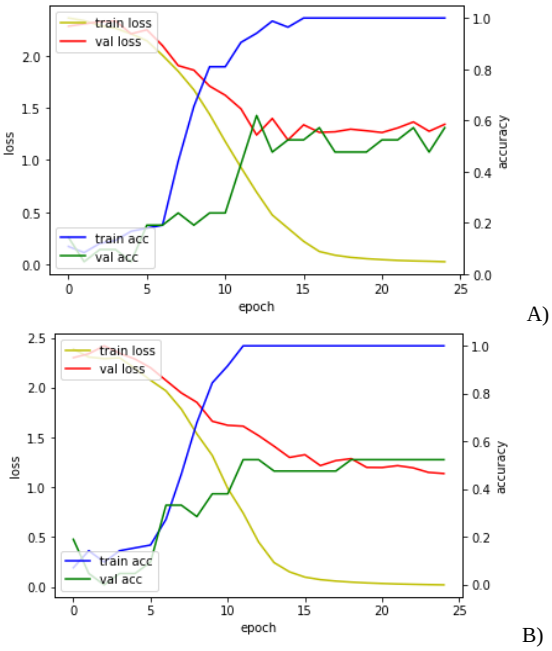


Fig. 4. Graphe de performance éloction normale a) et élocution rapide b)

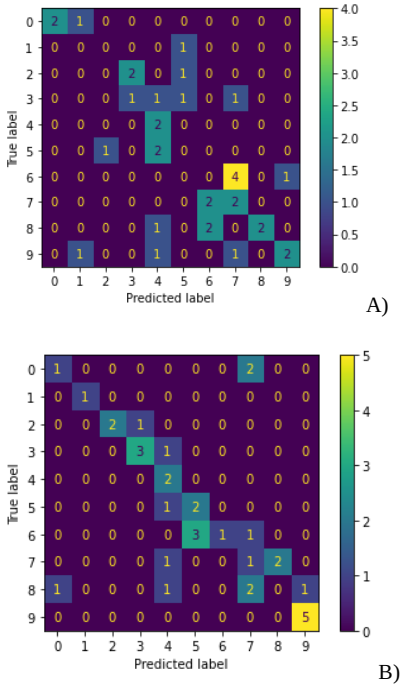


Fig. 5. Matrice de confusion éloction normale a) et élocution rapide b)

VI. Discussion

Nous obtenons un score de 68% dans les meilleures conditions nous permettant de valider l'approche faite par [4] stipulant que l'association de MobilNet et d'un LSTM permette d'obtenir des scores favorables d'au moins 68% en générale sur la détection de mots. Or, il est tout aussi intéressant de se pencher également sur le travail fait par Garg et al [5], mettant en évidence que RNN tel le LSTM obtient de meilleur résultat que la combinaison du CNN et du LSTM. Ils obtiennent quant à eux en score allant jusqu'à 73%, montrant que son utilisation du LSTM performe mieux tout seul qu'avec le CNN. Un de leur graphique de comparaison est en annexe.

VII. Conclusion

En ce jour, il est difficilement concevable d'avoir une idée claire quant aux résultats. Il est vrai que nous avons une idée de ce que peuvent être les résultats. Mais le fait d'avoir préparé soi-même la collection de données, peut apporter une autre source d'erreur. Les données n'ont peut-être pas été prises dans les meilleures conditions extérieures (source de lumière, contre-jour, saturation de l'image, ROI) pouvant avoir des coïncidences sur les résultats. Mais restons optimiste sur la suite du déroulement. Le forage des données a été produit par la méthode Haar Cascade qui offre des résultats recevables avec quelques ajustements faits par les méthodes de Backproject et MeanShift. Il aurait été tout à fait possible de les extraire grâce aux landmarks points d'intérets pour le tracking de visage. Toutefois Haar Cascade était beaucoup plus maîtrisé pour ma part.

Grâce à l'api de Keras il a été assez simple de modéliser un modèle reprenant les caractéristiques de celui voulant être traité au cours de ce projet. Tout est à disposition pour assembler différentes couches les unes avec les autres de manières élémentaires.

Par la suite, si le temps est en notre faveur, il sera envisageable de traiter des données avec des phrases qui sont pour moi beaucoup plus pertinentes que des chiffres. Quelques difficultés ont été rencontrées comme la création du modèle sur Keras, utilisant une ancienne version de MobilNet. Il sera judicieux d'essayer les nouvelles versions qui peuvent apporter notamment des améliorations de compatibilités avec Python mais aussi des gains en performance et en temps.

VIII. Conclusion

En ce jour, il est difficilement concevable d'avoir une idée claire quant aux résultats. Il est vrai que nous avons une idée de ce que peuvent être les résultats. Mais le fait d'avoir préparé soi-même la collection de données, peut apporter une autre source d'erreur. Les données n'ont peut-être pas été prises dans les meilleures conditions extérieures (source de lumière, contre-jour, saturation de l'image, ROI) pouvant avoir des coïncidences sur les résultats. Mais restons optimiste sur la suite du déroulement. Le forage des données a été produit par la méthode Haar Cascade qui offre des résultats recevables avec quelques ajustements faits par les méthodes de Backproject et MeanShift. Il aurait été tout à fait possible de les extraire grâce aux landmarks points d'intérets pour le

tracking de visage. Toutefois Haar Cascade était beaucoup plus maîtrisé pour ma part.

Grâce à l'api de Keras il a été assez simple de modéliser un modèle reprenant les caractéristiques de celui voulant être traité au cours de ce projet. Tout est à disposition pour assembler différentes couches les unes avec les autres de manières élémentaires.

Par la suite, si le temps est en notre faveur, il sera envisageable de traiter des données avec des phrases qui sont pour moi beaucoup plus pertinentes que des chiffres. Quelques difficultés ont été rencontrées comme la création du modèle sur Keras, utilisant une ancienne version de MobilNet. Il sera judicieux d'essayer les nouvelles versions qui peuvent apporter notamment des améliorations de compatibilités avec Python mais aussi des gains en performance et en temps.

IX. Utilisation néfaste

Une des utilisations néfastes qu'on pourrait faire avec ce genre de modèle, si nous arrivions à le rendre portatifs et de pouvoir enregistrer à l'insu des personnes des enregistrements qui pourraient les nuire.

Remerciement

Nous remercions K. Ditsch d'avoir sacrifié de son temps pour la collection de données.

References

[1] G. T. Themos Stafylakis, «Combining Residual Networks with LSTMs for Lipreading,» 2017.

[2] C. K. H. J. Y. WoMa, «Read My Lips,» 2019.

[3] K. Shrestha, «Lip Reading using Neural Network and Deep learning».

[4] A. D. Anna Pyataeva, «Artificial neural network technology for lips reading,» 2021.

[5] J. N. S. B. Amit Garg, «Lip reading using CNN and LSTM,» 2015.

[6] M. Tyagi, «Towardsdatascience - Viola Jones Algorithm and Haar Cascade Classifier,» 13 Juillet 2021. [En ligne]. Available: https://towardsdatascience.com/viola-jones-algorithm-and-haar-cascade-classifier-ee3bfb19f7d8. [Accès le 10 juillet 2022].

[7] K. Doshi, «Towardsdatascience,» 9 Mai 2021. [En ligne]. Available: https://towardsdatascience.com/foundations-of-nlp-explained-bleu-score-and-wer-metrics-1a5ba06d812b. [Accès le 10 Juillet 2022].

[8] «Keras API,» [En ligne]. Available: https://keras.io/api/applications/mobilenet/.

Annexe

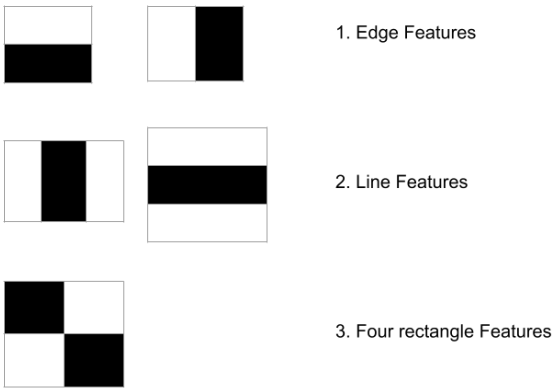


Fig 1. Filtre de Haar

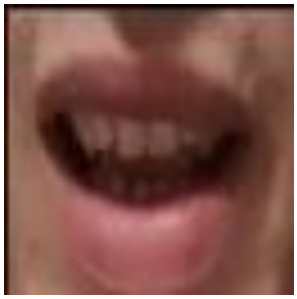


Fig 2. Exemple ROI lèvres

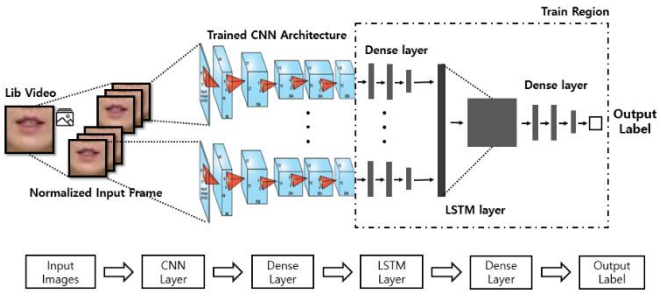


Fig 3. Modèle MobilNet