Détecteur des masques

Julien Furiga, Jérémy Le Joncour

Contexte du projet

Avec la pandémie COVID-19, dans pas mal de pays le port de masques est devenu obligatoire pour se protéger contre ce virus mortel. Dans ce projet, vous allez développer un modèle IA - Détecteur de masque sur les images en premier temps puis en temps réel avec Python.

1. Création de la liste d'apprentissage

La première étape consistait à réaliser notre *Dataset Train* et *Test*. Après avoir affilié la direction des dossiers (avec une liste nommée *categories* [*without_mask* et *with_mask*]) des données *Train* grâce au module Os.Path, nous avons pu les afficher.

Les images ont ensuite été modifiées et affichées avec CV2 avec une taille prédéfinie de *96x96* pixels.

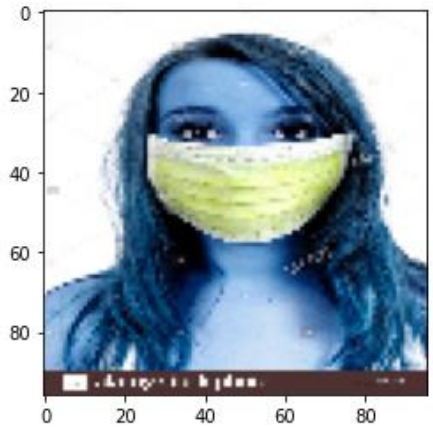

Figure 1. Affichage d'une image CV2 (With_mask Train)

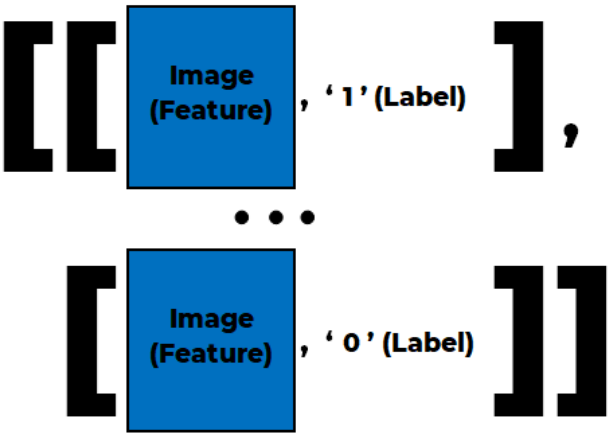

Figure 2. Schéma de la train_list

Notre *train_list* (Figure ci-dessus à Droite) a ensuite été créée par une fonction *create_train_list()* qui répertoriait ainsi toutes les images. Ces images, qu'elles proviennent du dossier *without_mask* ou *with_mask,* ont subi un *RESIZE 96x96* et assignées à leur *Label* défini par l'index de leur catégorie (soit 0 et 1). Cette liste d'apprentissage a été mélangée grâce à la fonction *random.shuffle()*. Les listes Features (*x_app*) et Label (*y_app*) ont ensuite été générées par une boucle *for* depuis la *train_list*.

Pour les besoins de notre modèle, *x_app* a été reshape en [-1, 96, 96, 3] du fait que l'image soit traitée en couleur.

2. Création de la liste de test

Une démarche similaire a été réalisée sur notre *Test-set* avec une fonction *create_test_list().*

3. Création du CNN

Notre modèle contient 3 couches :

- Pour l'initialisation de notre modèle, les couches *Conv2D* sont utilisées sur les traitements d'objets bidimensionnelles, et contiennent différentes classes comme les filtres (dont le nombre est réglé à 64), avec une définition de *kernel* de dimension (3, 3) idéale pour des images ayant une taille inférieure à *96x96* pixels.

  L'input_shape reprend les dimensions des images que nous fournissons à notre modèle (*96x96* pixels en 3 couleurs RGB). Enfin, nous avons définis une activation *reLU*, l'activation linéaire standard qui permet de fixer les valeurs négatives de nos matrices à 0. *MaxPooling2D* réduit les dimensions des images injectées et conserve les traits principaux.

- Les dernières couches *Flatten* et *Dense* font la liaison entre les couches précédentes et convertissent les données en matrice à 1 dimension. Nous utilisons enfin l'activation *Sigmoid* et la fonction *loss binary_crossentropy*, performant pour la classification binaire.

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 94, 94, 64)        1792
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 92, 92, 64)        36928
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 46, 46, 64)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 44, 44, 64)        36928
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 42, 42, 64)        36928
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 21, 21, 64)        0
_________________________________________________________________
flatten (Flatten)            (None, 28224)             0
_________________________________________________________________
dense (Dense)                (None, 64)                1806400
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 65
=================================================================
Total params: 1,919,041
Trainable params: 1,919,041
Non-trainable params: 0
```

Figure 3. CNN résumé

4. Evaluation du CNN

Afin d'évaluer notre modèle, le score de *précision* et le *loss* ont été calculés.

```
7/7 [==============================] - 6s 814ms/step - loss: 0.1914 - accuracy: 0.9692
loss :  0.19139912724494934
acc :   0.9692307710647583
```

Figure 4. Score de précision et loss

Le modèle a ensuite été évalué sur nos données *Test* avec un *predict_classes* qui va affilier un Label sur une image Test. Une matrice de confusion est générée afin de comparer la prédiction avec le Label réel de l'image.

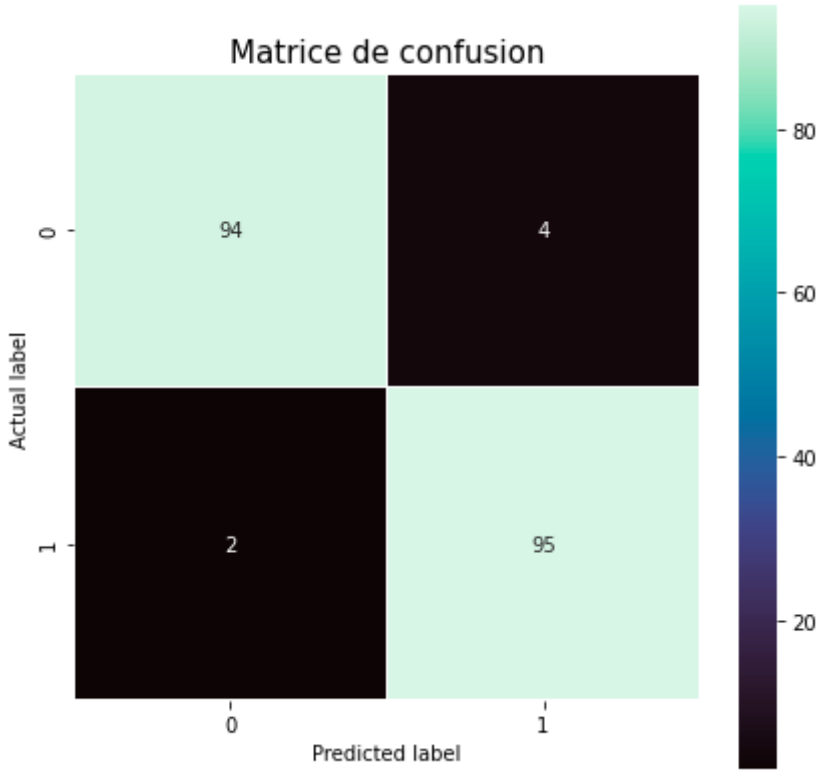

Figure 5. Matrice de confusion

Notre modèle classe correctement 189 images de notre *Dataset* Test (94 pour la classe sans masque = 0 ; 95 pour la classe avec masque = 1). Seul 6 images ne sont pas définies correctement par notre modèle :

- 4 images sans masque sont considérées comme représentant des personnes avec des masques, et inversement, 2 images représentant des gens avec masques sont considérées sans masque.

5. Test sur de nouvelles images

Le modèle a été testé sur de nouvelles images récupérées sur un moteur de recherche. Pour un affichage simplifié de la prédiction du modèle, une phrase indique si l'individu porte ou non un masque à la place du Label (0 ou 1).

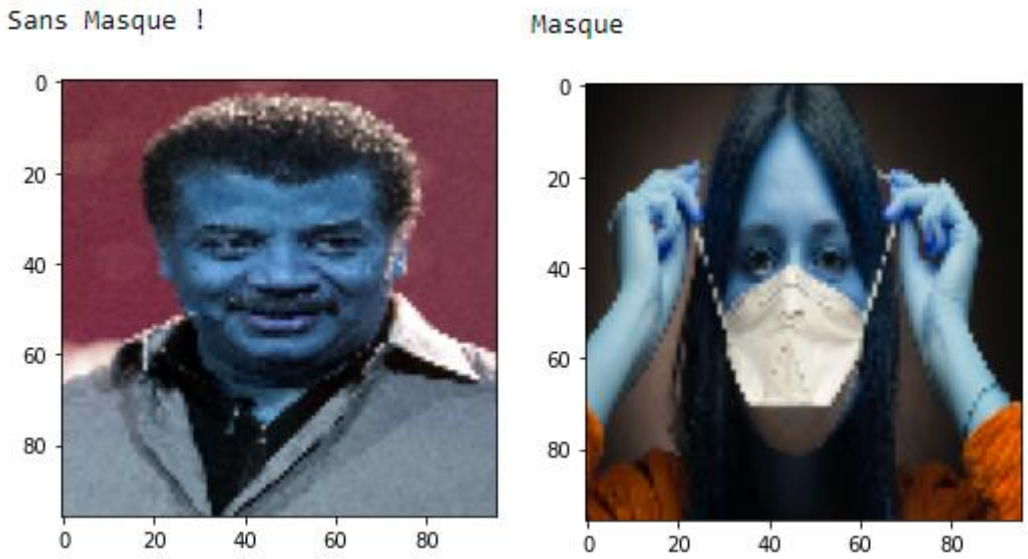

Figure 6. Test sur de nouvelles images (sans masque à gauche, masque à droite).

6. Test sur Vidéo

CV2 .VideoCapture a été utilisé pour tester notre modèle en temps réel. Afin de connaitre la prédiction du modèle, une image (smiley positif et négatif) s'affiche si on porte ou non un masque (intégrée via *cv2.addWeighted*). Une voix (définie *wave_obj*) s'active lorsque nous ne portons pas de masque.

Frame_predict est utilisé pour détecter notre visage devant la caméra. Après un ajustement de chaque image (*96x96* pixel), le modèle peut effectuer une prédiction.

Le programme peut s'arrêter en appuyant sur la « q » du clavier.

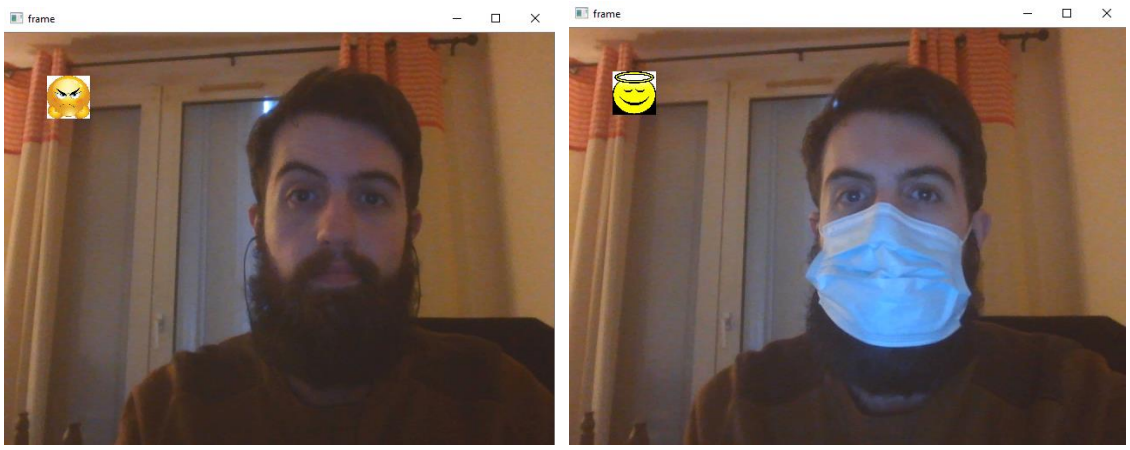

Figure 7. Test sur vidéo avec et sans masque.

7. Discussion

Le modèle créé possède un bon ratio de bonne prédiction dans l'ensemble, notamment sur les images. Sur vidéo, le modèle arrive à détecter le port du masque correctement. Toute fois, il ne détecte pas le masque sous le nez comme sans masque. De plus, il ne détecte pas le visage si l'individu porte des lunettes à cause du reflet de l'écran.