Détecteur de masque

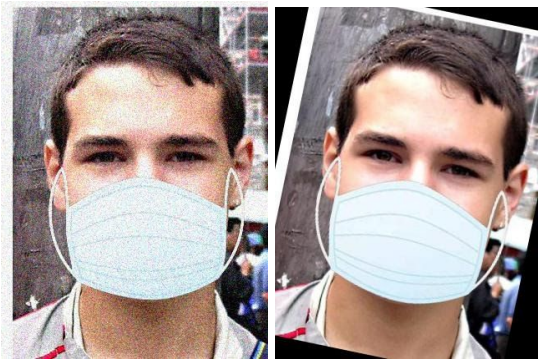
Amaury, Baptiste LG, Ludivine

A. Description des données

Les données sont des photos de taille moyenne représentant des personnes affublées de masques ou non. Les données sont séparées en deux parties train et test elles-mêmes étiquetées en deux groupes (avec masque et sans masque).

Il y a 657x2 images en train et 97x2 images en test. C'est donc un dataset assez petit, d'autant que les masques sont généralement de simples ajouts sur des photos normales. Les masques sont blancs et sans motifs.

On remarque que certaines images sont présentes en plusieurs exemplaires avec des variations de bruit ou d'inclinaison. Ce procédé s'appelle de la *Data Augmentation* et vise à mieux exploiter un jeu de données parfois de taille trop modeste. L'algorithme entraîné avec des données augmentées est normalement plus robuste.



B. Présentation de l'architecture utilisée

Prétraitement avant passage dans le modèle :

On standardise afin que chaque pixel ait une valeur située entre 0 et 1.
On redimensionne ensuite les données d'entrée pour qu'elles puissent être passées en argument `input_shape` de la première couche de convolution.
Conv2D travaille avec des images qui ont une hauteur, une largeur et un shema de couleur (1 pour greyscale, 3 pour RGB)

Création du modèle :

On cherche à prédire le port du masque sur un visage: **il s'agit d'un problème de classification binaire**, car on cherche à savoir si oui (1), la personne porte un masque, ou bien si non (0), il n'en a pas.

Nous allons utiliser un modèle de **CNN** pour résoudre ce problème de classification. Les paramètres à utiliser sont les suivants :

- La dimension de la couche d'entrée doit correspondre au nombre de caractéritiques d'entrée (= taille de l'image), soit [100,100,3] (hauteur, largeur, canaux de couleur)
- La dimension de la couche de sortie sera de 1 neurone, puisqu'on a une seule classe de sortie, la probabilité que la personne porte un masque. On utilisera une **fonction sigmoïde** en sortie pour estimer cette probabilité, et **l'entropie croisée binaire** pour la fonction de perte.
- Pour les couches cachées, nous utiliserons la fonction **d'activation ReLu,** qui est efficace dans la plupart des cas.

Différents modèles séquentiels avec différents paramètres ont été testés pour déterminer le modèle qui renvoyait les meilleurs résultats. Il est composé ainsi :

```
model2.add(Conv2D(16, 3, padding="same", activation='relu', input_shape=(100,100,3)))
model2.add(Conv2D(16, 3, padding="same", activation='relu'))
model2.add(MaxPooling2D(2))
model2.add(Conv2D(32, 3, padding="same", activation='relu'))
model2.add(Conv2D(32, 3, padding="same", activation='relu'))
model2.add(MaxPooling2D(2))
model2.add(Flatten())
model2.add(Dense(32, activation="relu"))
model2.add(Dropout(0.25))
model2.add(Dense(16, activation="relu"))
model2.add(Dropout(0.25))
model2.add(Dense(1, activation='sigmoid'))
```

Ce modèle permet d'atteindre plus de 97% d'accuracy sur la base de test en un temps relativement réduit.

C. Utilisation de cv2 pour la détection de masque

OpenCV va nous servir dans plusieurs tâches :

- capture de la vidéo
  - cv2.VideoCapture(0): va récupérer la vidéo depuis la webcam
- traitement de la vidéo pour pouvoir la traiter
  - cv2.cvtColor/cv2.COLOR_BGR2GRAY: passage de l'image de RGB en gris

    - cv2.resize: changer le format de l'image pour quelle corresponde au traitement dans le modèle
- affichages:
    - cv2.rectangle: affichage d'un rectangle autour de la tête lors de la reconnaissance faciale
    - cv2.flip: sert à 'flip' un tableau 2D. Dans notre cas, sert à créer un effet miroir
    - cv2.imshow: nous permet d'afficher la vidéo
- fonctions de la fenêtre créée
    - cv2.waitKey(1): va afficher une frame pendant 1 ms puis va fermer l'affichage automatiquement. Dans notre cas, nous avons assigné la touche code 27(ESC) à cette action.
    - cv2.destroyAllWindows: détruit les fenêtres actives

D. Conclusion

Rapidité

L'algorithme CNN a mis un peu de temps à être entraîné, en revanche il prédit quasi instantanément sur le flux vidéo.

Efficacité

L'algorithme CNN fonctionne plutôt bien pour prédire si la personne filmée porte ou non un masque. La découverte de features par l'algorithme semble s'être bien déroulée. En revanche, les fortes variations de luminosité et de contraste induites par un changement de l'environnement du pc portable jouent parfois beaucoup sur la valeur de l'array de prédiction renvoyé par le modèle.

On peut jouer sur le seuil de déclenchement. Autre possibilité : un réglage du contraste et de la luminosité directement sur l'image qui subit la prédiction permet d'y remédier mais l'algo manque peut-être un peu de robustesse en conséquence.

Perspectives

Peut-être qu'une démultiplication (*Data Augmentation*) des images d'apprentissage avec des images fortement assombries et bruitées ou bien au contraire surexposées permettrait une meilleure prédiction en conditions réelles dégradées.

En outre, l'algo ne fonctionne plus correctement lorsqu'il y a plusieurs personnes à l'écran et il reste adapté uniquement à des personnes vues de face ou très légèrement tournées.