Résumé

L’objectif du cas pratique est d’améliorer une application et une IA d’authentification d’employés par la reconnaissance d’œil.

Données

Nous disposons de 5 photos de chaque œil des 45 employés de l’entreprise.

Pour les employés 8, 13 et 43, les dossiers des yeux gauches et droits étaient inversés.

Modèle

Le modèle utilisé par l’ancien développeur pour reconnaitre l’œil des employés est VGG16. Il s’agit d’un modèle CNN (Convolutionnal Neural Network) pré-entrainé sur une large quantité d’images, et qui dispose de filtres de convolution de petites dimensions, donnant de meilleures performances que les modèles avant lui.

Dans le modèle de base, une couche « flatten » (aplatissement) et une couche « dense » ont été ajoutée, contenant autant de neurones que d’employés, avec une activation « softmax ».

Les données étaient redimensionnées et standardisées lors de leur import dans l’IDE (Integrated Development Environment.

Voici les résultats obtenus en jouant sur le batch\_size et les epochs :

Tableau : Résultats

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Batch\_size = 30  Epochs = 5 | Batch\_size = 10  Epochs = 5 | Batch\_size = 60  Epochs = 5 | Batch\_size = 30  Epochs = 50 | Batch\_size = 10  Epochs = 50 | Batch\_size = 60  Epochs = 50 |
| Loss | 0.0358 | 7.53\*10^-4 | 0.0573 | 4.34\*10^-5 | 5.1\*10^-4 | 9.3\*10^-4 |
| Accuracy | 0.99 | 1 | 0.9778 | 1 | 1 | 1 |
| Val\_loss | 0.86 | 0.83 | 0.4627 | 0.29 | 1.09 | 0.6 |
| Val\_accuracy | 0.71 | 0.76 | 0.84 | 0.93 | 0.8 | 0.84 |

On voit que les résultats sont meilleurs lorsqu’on augmente la durée d’entraînement (epochs), jusqu’à 0.93 d’accuracy , en gardant le batch\_size de base qui était 30.

Data augmentation :

Une image contenant texte, Police, capture d’écran

Description générée automatiquement

5 images générées par images {insérer schéma ramification + création et sauvegarde des images}

Le modèle possède une couche de normalisation intégrée (/255)

Tableau : Résultats avec data augmentation

|  |  |  |  |
| --- | --- | --- | --- |
|  | Batch\_size = 30  Epochs = 50 | Defriezing des 2 dernières couches du modèle | Toutes les couches de VGG16 sont défriezées |
| Loss | 0.016 | 4.5\*10^-4 | 3.8 |
| Accuracy | 1 | 1 | 0.02 |
| Val\_loss | 2.03 | 3.1 | 3.8 |
| Val\_accuracy | 0.55 | 0.48 | 0.02 |

Certaines photos l’œil est à la marge (horizontale ou latérale)

Model original :

Vgg16 -> Aucune des couches n’est entrainable

Flatten

Dense -> softmax

Idée :

for layer in vgg.layers[:-2]:

    layer.trainable = False

*# Réajustement de la dimension de l'image*

  layers.Resizing(IMSIZE[0], IMSIZE[1], *crop\_to\_aspect\_ratio*=True),

*# Normalisation des pixels*

  layers.Rescaling(1./255, *input\_shape*=IMSIZE),

C8. Modifier les paramètres et composants de l’intelligence artificielle afin d’ajuster aux objectifs du projet les capacités fonctionnelles de l’algorithme à l’aide de techniques d’optimisation

C14. Améliorer l’application d’intelligence artificielle en développant une évolution fonctionnelle pour répondre à un besoin exprimé par un client ou un utilisateur

Dans le rapport et la synthèse orale, le/la candidat(e) doit démontrer les éléments suivants :

✓ à partir d’un projet existant, interpréter les indicateurs de performance de l’intelligence artificielle disponibles,

✓ à partir des éléments d’interprétation, définir les caractéristiques des améliorations à apporter

✓ intégrer les améliorations à l'algorithme d’intelligence artificielle

✓ communiquer une estimation de charge au regard du besoin d’évolution de l’application

✓ intégrer l’évolution fonctionnelle

✓ tester la non régression de l’application suite à l’intégration de l’évolution