université virtuelle Burkina Faso

Sujet	Exercices n°1
Professeur	Dr. Issouf OUATTARA
Titre	Introduction au Deep Learning
Auteur	Coulibaly Cheick Ahmed

a. Détails sur le jeu de données

Pour cet exercice je choisie le dataset sur la classification des animaux, téléchargeable à travers le lien suivant : (https://www.kaggle.com/datasets/antobenedetti/animals).
Ce dataset est une version épurée du jeu de données Animal-5 Mammal de SHIV28.
Il contient 14 989 images d'animaux, regroupé en 5 classes, à savoir: chats (cat), chiens (dog), éléphants (elephant), chevaux (horse) et lions (lion).

Le dataset est divisé en de sous-ensembles d'entraînement (train), de validation (val) et de test final/l'inférence (inf).
inf ne contenant que 5 images, il sera utilisé uniquement pour illustrer des prédictions sur de nouvelles images, et val est utilisé comme validation/test.

Pour rendre le code portable, les données ont été importer depuis Kaggle en utilisant kaagle hub.

b. Détails sur le modèle d'apprentissage profond

Le jeux de données contenant 5 classes d'images, il est question de développer un modèle qui puisse reconnaître chacune de ces classes, il s'agit donc d'un problème de classification multi-classes.
Un réseau de neurones convolutif (CNN) est donc le type de réseau de neurone adapté pour la classification d'images.
Notre choix se porte donc sur le CNN comme modèle.

Le modèle commence par une étape de normalisation des pixels suivie d'une augmentation de données (data augmentation). Les transformations opérées sont entre autre RandomFlip, RandomRotation, RandomZoom.

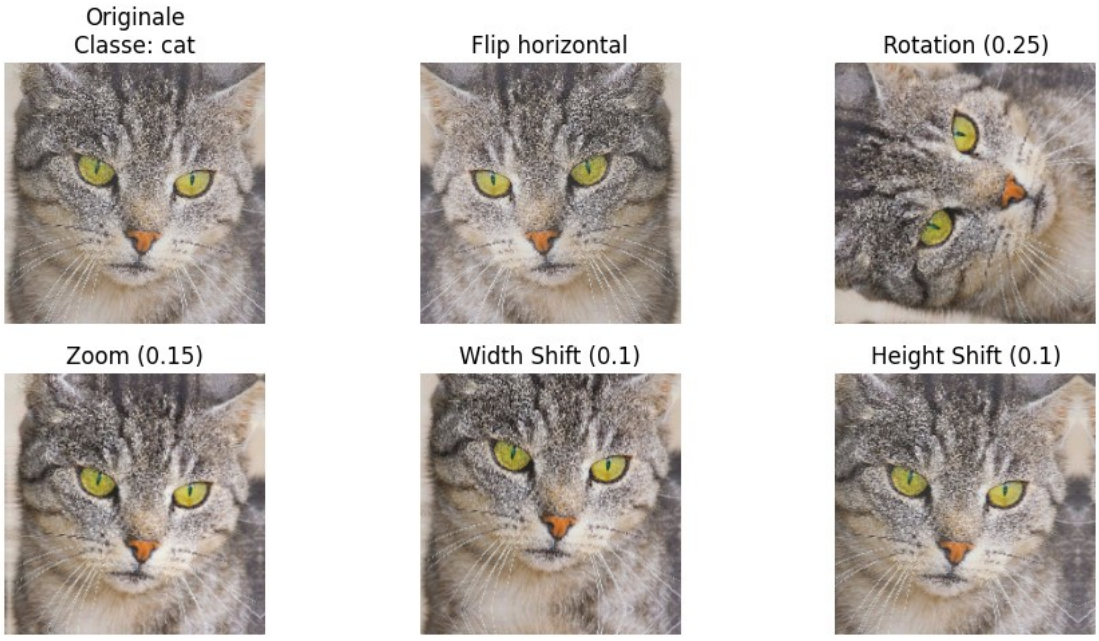

Figure 1: Illustration du data augmentation

L'architecture est composée de plusieurs couches convolutives avec des filtres de tailles croissantes (16, 32, 64 et 128). Chaque convolution utilise la fonction d'activation ReLU, et certaines sont suivies d'une normalisation de lot (Batch Normalization) pour stabiliser et accélérer l'entraînement. Après chaque convolution, une opération de max pooling réduit la dimension des cartes de caractéristiques afin de conserve les informations les plus pertinentes.
La partie finale du réseau est constituée d'un aplatissement (Flatten) suivi de couches denses entièrement connectées (128 et 64 neurones avec ReLU).
Pour limiter le surapprentissage, une couche de Dropout est utilisée. La sortie est une couche dense avec autant de neurones que de classes, utilisant la fonction Softmax pour obtenir une distribution de probabilités sur les différentes catégories.

Model: "sequential_47"

Layer (type)	Output Shape	Param #
rescaling_7 (Rescaling)	(None, 224, 224, 3)	0
data_augmentation (Sequential)	(None, 224, 224, 3)	0
conv2d_29 (Conv2D)	(None, 224, 224, 16)	448
batch_normalization_6 (BatchNormalization)	(None, 224, 224, 16)	64
max_pooling2d_24 (MaxPooling2D)	(None, 112, 112, 16)	0
conv2d_30 (Conv2D)	(None, 112, 112, 32)	4,640
max_pooling2d_25 (MaxPooling2D)	(None, 56, 56, 32)	0
conv2d_31 (Conv2D)	(None, 56, 56, 64)	18,496
max_pooling2d_26 (MaxPooling2D)	(None, 28, 28, 64)	0
conv2d_32 (Conv2D)	(None, 28, 28, 128)	73,856
max_pooling2d_27 (MaxPooling2D)	(None, 14, 14, 128)	0
flatten_6 (Flatten)	(None, 25088)	0
dense_18 (Dense)	(None, 256)	6,422,784
dropout_11 (Dropout)	(None, 256)	0
dense_19 (Dense)	(None, 128)	32,896
dropout_12 (Dropout)	(None, 128)	0
dense_20 (Dense)	(None, 64)	8,256
dense_21 (Dense)	(None, 5)	325

Total params: 6,561,765 (25.03 MB)
Trainable params: 6,561,733 (25.03 MB)
Non-trainable params: 32 (128.00 B)

Figure 2: Summary du modèle

c) Détails sur l'entraînement du modèle
Le transfert learning n'a pas été utilisé,
l'optimiseur choisi est Adam, avec un learning rate par défaut de 0.001 et les coefficients de moments standards (beta1=0.9, beta2=0.999), avec un mécanisme d'early stopping (patience=5) afin d'éviter la surapprentissage.
Le nombre d'epochs a été fixé à 100, mais l'entraînement s'est arrêté au 33^{e} epoch suite au mécanisme d'early stopping.

d) En observation les courbes de précision (accuracy) et de perte (loss) on peut dire que le modèle a bien appris et généralisé jusqu'à un certain point ; de plus le modèle atteint une précision de 85.10% sur le jeu de test, ce qui confirme que le modèle a une bonne capacité de généralisation.

D'après la matrix de confusion les catégories «dog» (94%) et «cat» (99%) sont bien reconnues, tandis que les classes «lion» (82%), «horse» (78%) et «elephant» (72%) sont parfois confondues entre elles.
Quant aux prédictions sur de nouvelles images, sur un lot de 25 images, le modèle a mal prédit une image de « chat » en « chien », mais arrive à bien prédire les autres images.

Les illustrations correspondantes à cette analyse se trouvent dans le notebook joint pour éviter de surcharger le rapport.

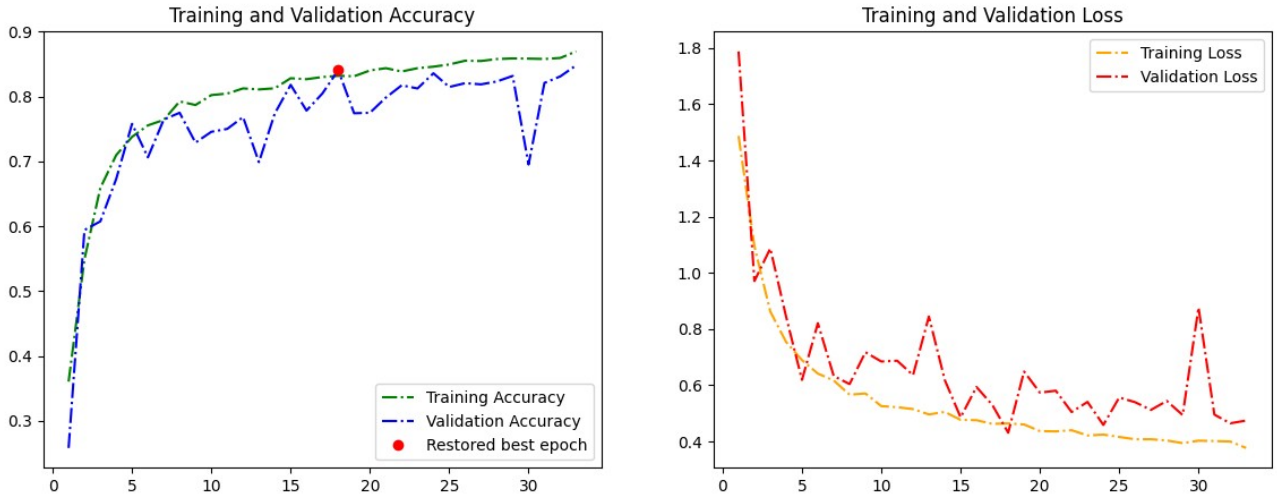

Figure 3: Courbes Accuracy et Loss

e) En conclusion, on peut dire que cet exercice a été un entraînement pour nous pour la construction d'un CNN, et nous a permis de percevoir l'intérêt de cette architecture pour la vision par ordinateur.
Les performances peuvent tout a fait être améliorées en augmentant soient le nombre d'images, environ 15 000 images qui est quand même peu pour des applications d'IA, et aussi en testant d'autre types d'architecture avec du transfert learning, comme le VGG16 qu'on a eu l'occasion d'étudier dans le résumé des articles, qui permet d'atteindre de très bons résultats même avec peu de données.
Vu la situation sécuritaire au Burkina Faso, avec des images réels de groupes armées on pourrait passer d'une simple classification d'animaux à un système de reconnaissance et détection de menaces qui aiderait à la situation sécuritaire du pays. Par exemple, distinguer nos forces de défense des terroristes qui s'habillent souvent avec les tenus de l'armée.