RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE
MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITÉ DES SCIENCES ET DE LA TECHNOLOGIE
HOUARI BOUMEDIENNE
FACULTÉ D'INFORMATIQUE

Rapport projet OCR

Architecture de deep learning pour la reconnaissance du manuscrit arabe

AMEZIANE Abdelghani 181831092072
SEMMAR Hichem 181832056972

Table des matières

Table des figures

Introduction

L'OCR, ou reconnaissance de caractère optique, est l'une des premières tâches de vision informatique adressées, car à certains égards, elle ne nécessite pas de deep learning. Par conséquent, il y a eu différentes implémentations d'OCR avant même le boom du deep learning en 2012, et certains sont même revenus à 1914.

Cela fait que beaucoup de gens pensent que le défi de l'OCR est «résolu», il n'est plus difficile. Une autre croyance qui provient de sources similaires est que l'OCR ne nécessite pas de deep learning, ou en d'autres termes, l'utilisation du deep learning pour l'OCR est une approche exagérée.

Quiconque pratique la vision par ordinateur ou l'apprentissage automatique en général sait qu'il n'y a pas de tâche résolue, et ce cas n'est pas différent. Au contraire, l'OCR donne de très bons résultats uniquement sur des cas d'utilisation très spécifiques, mais en général, il est toujours considéré comme difficile.

De plus, il est vrai qu'il existe de bonnes solutions pour certaines tâches OCR qui ne nécessitent pas de deep learning. Cependant, pour vraiment avancer vers de meilleures solutions plus générales, l'apprentissage en profondeur sera obligatoire.

Dans ce travail nous allons présenter notre approche de deep learning pour traiter ce problème, suivie d'une comparaison avec d'autre approche populaires.

Chapitre 1

Pré-traitement et augmentations des Données

1.1 Pré-traitement

L'ensemble de données qui nous a été fourni est déjà prétraité et nettoyé, ce qui signifie que la quantité de traitement que nous devions ajouter était minime.

La seule modifications que nous avons appliquer sur la dataset est l'unification des dimensions des images. On a opter pour une dimensions de 320*190, Parce que nous avons constaté que cette taille est le meilleur compromis entre la qualité de l'image et des temps d'entraînement raisonnables.

1.2 Augmentations

L'augmentation des données est une technique pour augmenter la diversité d'un ensemble de données sans effort pour collecter des données réelles en appliquant des transformations aléatoires (mais réalistes), cela contribuera à améliorer la précision du modèle et à empêcher le sur-apprentissage du modèle.

Dans notre travail on a appliquée les techniques d'augmentations suivantes :

1.2.1 Distorsion

la distorsion est une déformation géométrique des différents éléments qui composent l'image. Elle est le plus souvent engendrée par la conception optique de l'objectif.

Pour chaque image du dataset, une deuxième copie déformée est générée.

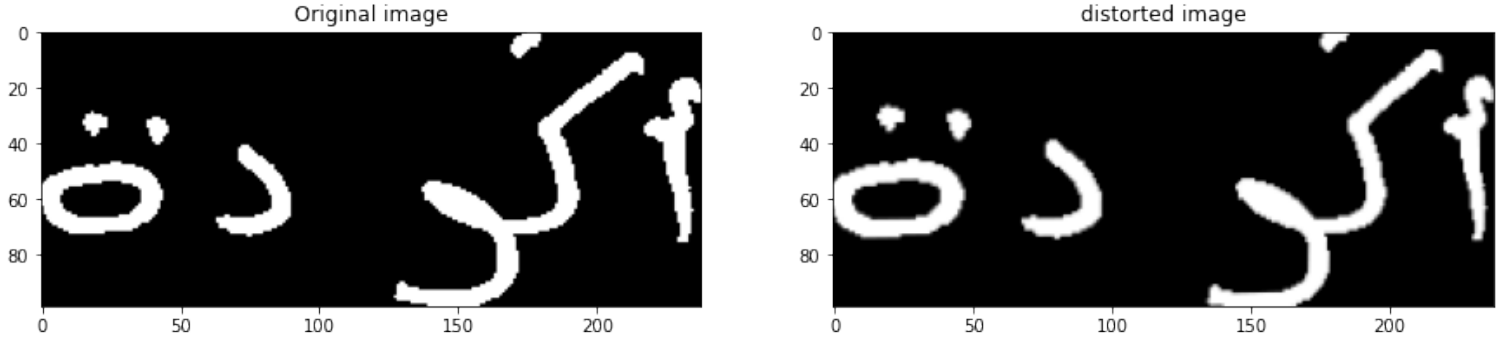

FIGURE 1.1 – Image originale (à gauche), Image déformée (à droite)

1.2.2 Décalage

On génère deux autres version de chaque image, tel que la première est une image décalée de 20 pixels en haut, et la deuxième est une image décalée de 20 pixels en bas.

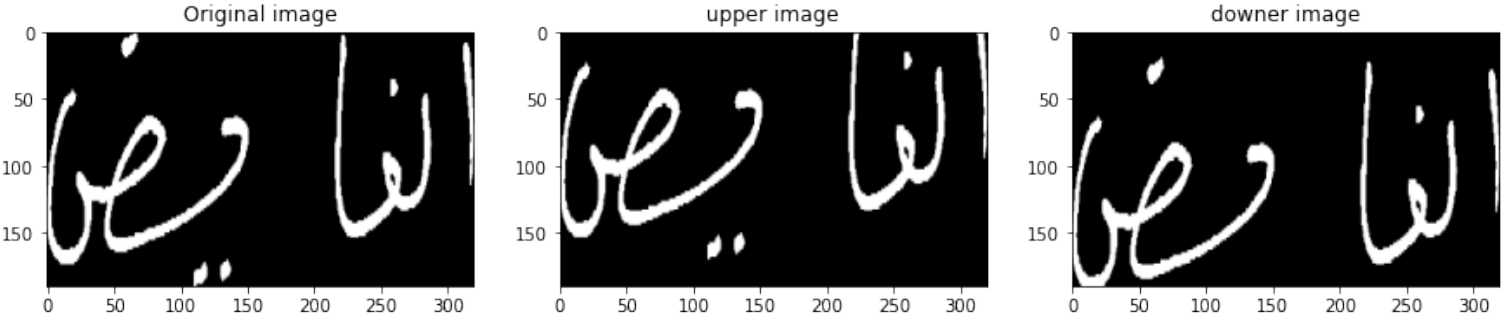

FIGURE 1.2 – Image originale (à gauche), Image décalée en haut (milieu), Image décalée en bas (à droite)

1.2.3 Rotation

Pour chaque image du dataset, une deuxième copie tournée est crée avec un angle de rotation de 20 degrés.

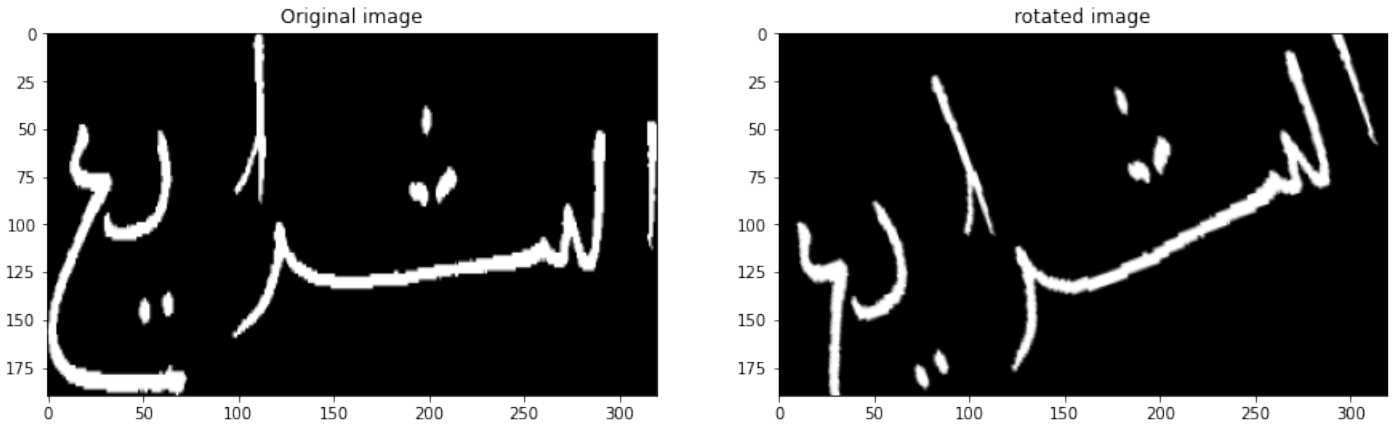

FIGURE 1.3 – Image originale (à gauche), Image tournée à 20 dégrées (à droite)

1.2.4 Filtres morphologiques

La morphologie est une opération de traitement d'images basée sur les formes. La valeur de chaque pixel dans l'image de sortie est basée sur une comparaison du pixel correspondant dans l'image d'entrée avec ses voisins. L'érosion et la dilatation sont ses deux opérations élémentaires :

— L'érosion : effectue un « et » logique entre les voisins d'un pixel (diminue le contour de l'ordre d'un pixel).

— la dilatation : effectue un « ou » logique entre les voisins d'un pixel (augmente l'épaisseur d'un contour).

Pour chaque image du dataset, on génère une version érodée avec un noyau de taille $4x4$, et une version dilatée avec un noyau de taille $5x5$.

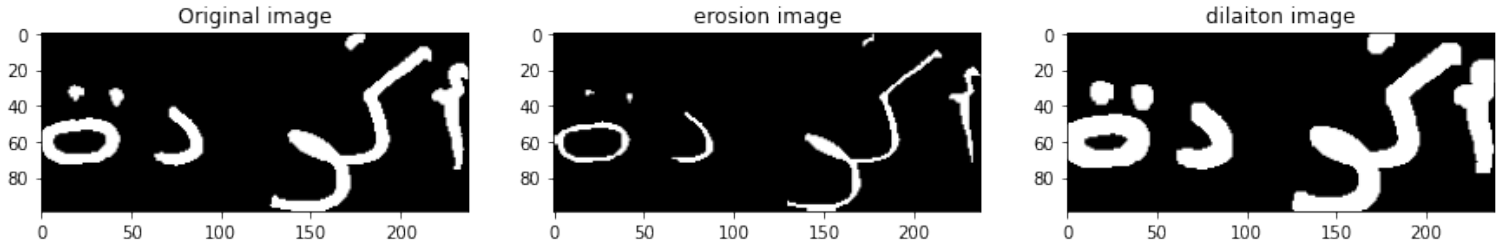

FIGURE 1.4 – Image originale (à gauche), Image érodée (milieu), Image dilatée (à droite)

1.2.5 Squelettisation

La squelettisation est un processus de réduction des régions de premier plan dans une image binaire à un reste squelettique qui préserve largement l'étendue et la connectivité de la région d'origine tout en jetant la plupart des pixels de premier plan d'origine.

La squelettisation est largement utilisée dans la représentation, la récupération, la manipulation, la correspondance, l'enregistrement, la reconnaissance et la compression d'objets.

Il peut faciliter un traitement d'image rapide et précis sur le squelette léger au lieu d'une opération autrement grande et à forte intensité de mémoire sur l'image d'origine, et pour cette raison on ajoute dans le dataset le squelette de chaque image originale.

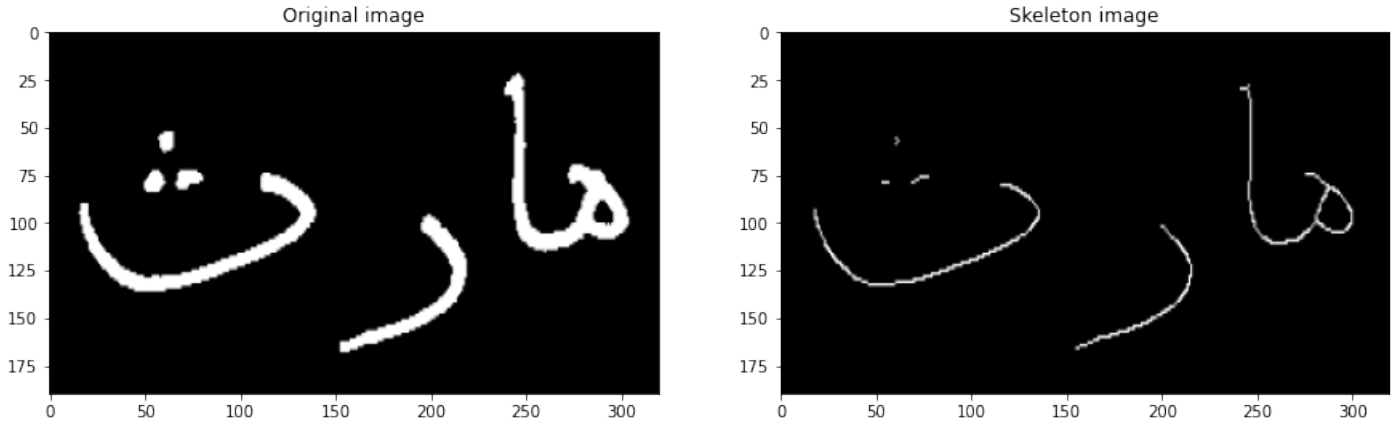

FIGURE 1.5 – Image originale (à gauche), Image tournée à 20 dégrées (à droite)

Chapitre 2

Architecture préexistantes de CNN pour l'OCR

2.1 Introduction

Dans ce chapitre nous allons présenter les architectures populaires souvent utilisées pour l'OCR.

2.2 RESNET50

RESNET, abréviation de Residual Network est un réseau de neurrones utilisé pour de nombreuses tâches de vision par ordinateur. Ce modèle a été le gagnant de ImageNet Challenge en 2015. La percée fondamentale avec RESNET était qu'elle nous a permis de former avec succès des réseaux de neurones extrêmement profonds avec plus de 150 couches. Avant RESNET, l'entraînement des réseaux de neurones très profonds été difficiles en raison du problème de disparition de gradient.

RESNET50 est une variante du modèle RESNET qui a 48 couches de convolution avec une couche maxpool et une couche average pool. Il a $3,8x10^9$ opérations de virgule flottante.

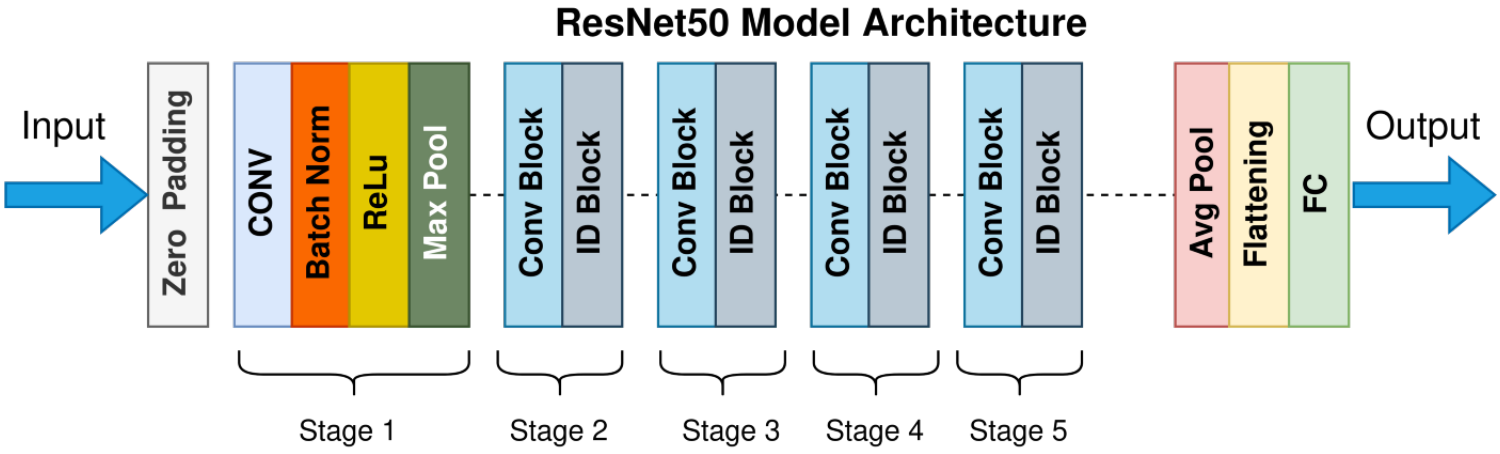

FIGURE 2.1 – Architecture RESNET50

2.3 Inception v3

Inception V3 est un réseau neuronal convolutionnel pour aider à l'analyse d'image et à la détection d'objets, et a fait ses débuts en tant que module pour Googlenet.

Il s'agit de la troisième édition du réseau neuronal de convolution Inception de Google, initialement introduit lors du défi de reconnaissance ImageNet.

La conception de Inception V3 visait à permettre des réseaux plus profonds tout en empêchant le nombre de paramètres de croître trop grand : il a "moins de 25 millions de paramètres", contre 60 millions pour Alexnet.

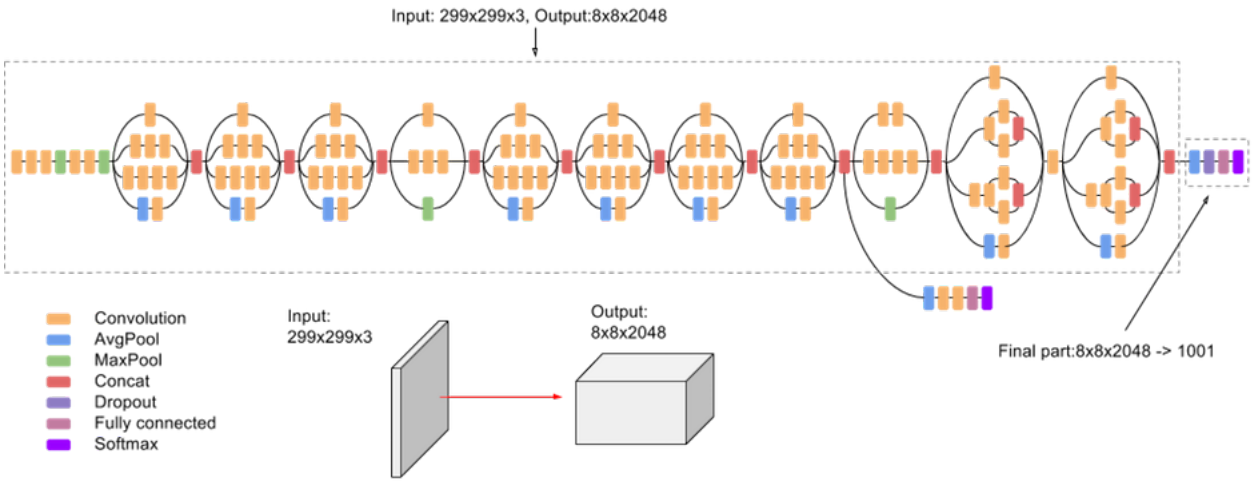

FIGURE 2.2 – Architecture Inception v3

2.4 VGG16

VGG16 est un type de CNN qui est considéré comme l'un des meilleurs modèles de vision informatique à ce jour.

Le 16 en VGG16 fait référence aux 16 couches qui ont des poids. Dans VGG16, il y a treize couches convolutionnelles, cinq couches de maxpool et trois couches denses, ce qui nous donnent 21 couches, mais il n'a que seize couches de poids, c'est-à-dire les couches de paramètres apprenables.

La chose la plus unique à propos de VGG16 est qu'au lieu d'avoir un grand nombre d'hyper-paramètres, ils se sont concentrés sur les couches de convolution de filtre 3x3 avec une stride de 1 et ont toujours utilisé le padding same et la même couche maxpool de filtre 2x2 et de stride 2.

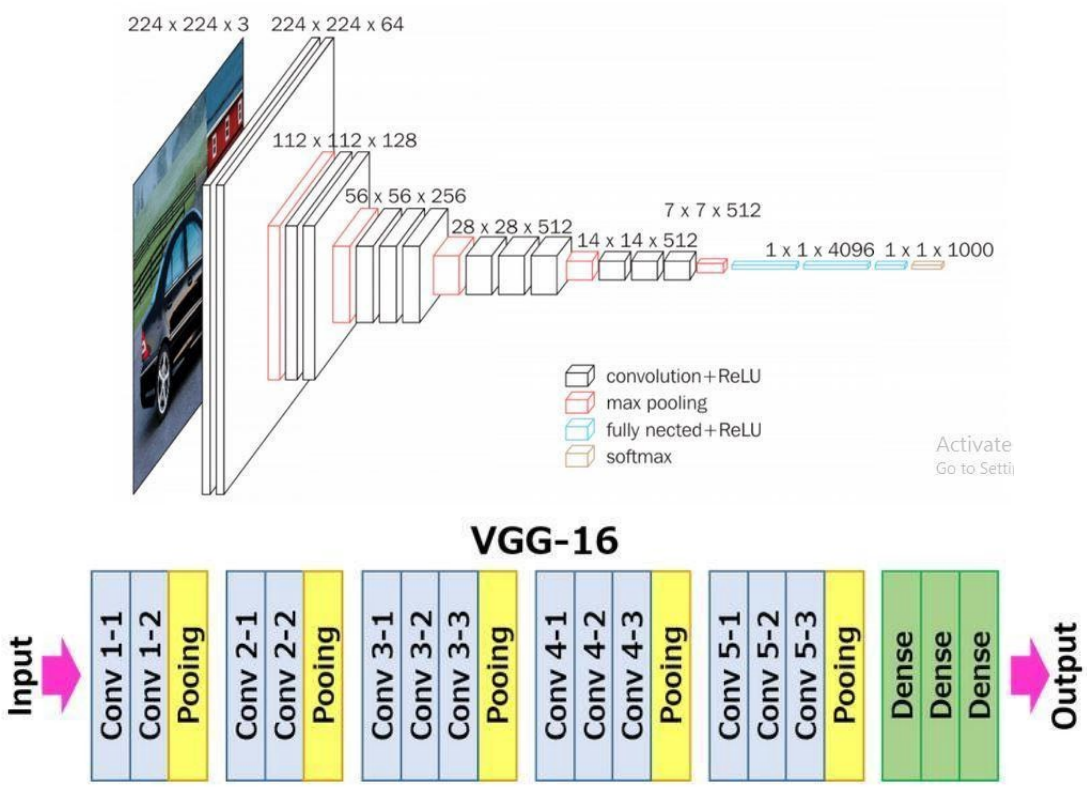

FIGURE 2.3 – Architecture VGG16

Chapitre 3

Solution deep learning proposée

3.1 Introduction

Durant ce chapitre nous allons présenter et expliquer notre propre solution de deep learning pour le problème de l'OCR.

3.2 Couche d'extraction de caractéristiques

La première partie du modèle se compose couche d'extraction de caractéristiques qui reçoit une image de (190 × 320). L'architecture est divisé en 9 parties :

— Une couche convolutionnelle avec un noyau de 3 × 3 se déplaçant avec un stride de 1 × 2 produisant 60 filtres activés avec la fonction RELU.
— Une couche de max pooling d'une taile de 2 × 2

— Une couche convolutionnelle avec un noyau de 3 × 3 se déplaçant avec un stride de 1 × 2 produisant 30 filtres activés avec la fonction RELU.
— Une couche de max pooling d'une taile de 2 × 2

— Une couche convolutionnelle avec un noyau de 3 × 3 se déplaçant avec un stride de 1 × 2 produisant 20 filtres activés avec la fonction RELU.
— Une couche de max pooling d'une taile de 2 × 2

— Une couche convolutionnelle avec un noyau de 3 × 3 se déplaçant avec un stride de 1 × 2 produisant 20 filtres activés avec la fonction RELU.
— Une couche de max pooling d'une taile de 2 × 2.

— Couche flatten qui a 20 vecteurs de longueur 11 comme entrée, et qui donne un vecteur de 220 paramètres qui seront inséré dans le réseau de neurones qui va être détaillée dans la deuxième partie.

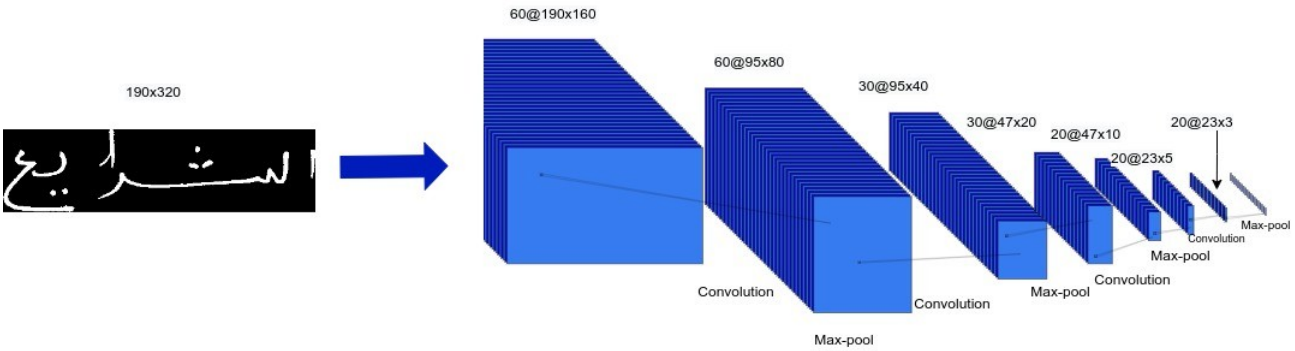

FIGURE 3.1 – Couche d'extraction de caractéristiques de notre CNN

3.3 Couche de classification

La deuxième partie du modèle est une architecture de réseau de neurrones de 5 couches :

- couche d'entrée qui se compose de 220 paramètres qui représente le résultat de la couche d'extraction de caractéristiques.
- Une couche cachée de 256 neurones activés avec la fonction RELU et avec un dropout de 0,2%.
- Une couche cachée de 128 neurones activés avec la fonction RELU et avec un dropout de 0,2%.
- Une couche cachée de 64 neurones activés avec la fonction RELU et avec un dropout de 0,2%.
- Couche de sortie finale avec 21 neurones activés avec une fonction softmax indiquant la classification.

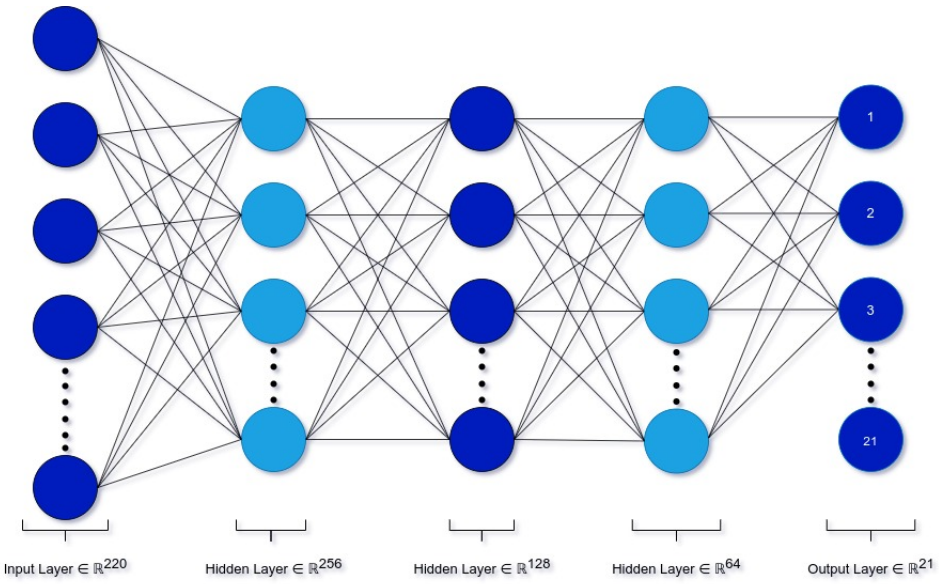

FIGURE 3.2 – Couche de classification de notre CNN

3.4 Conclusion

Dans ce chapitre nous avons détaillée l'architecture de notre solution proposée, et au final on obtient un modèle avec 124,963 paramètres qui sont tous entraînables.

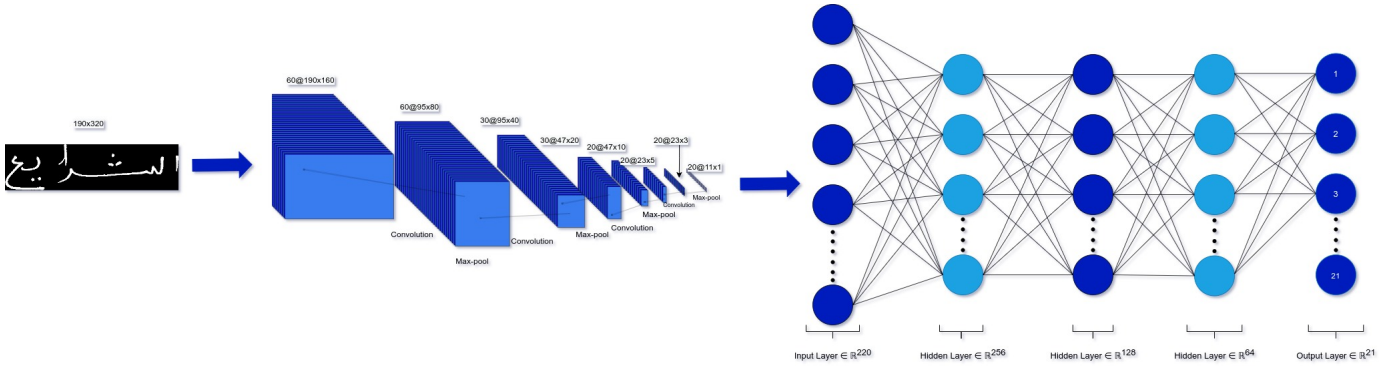

FIGURE 3.3 – Schéma générale de notre CNN

Chapitre 4

Résultats finaux et comparaison

4.1 Introduction

Dans ce chapitre nous allons présenter les résultats de l'entraînement et de l'évaluation de chacun des modèles mentionnées jusqu'à présent, et donner une comparaison globale a la fin de ce chapitre.

On voudrait aussi préfacer ce chapitre en précisant qu'en cause de limitation de mémoire, nous avons été forcée a utiliser seulement deux méthodes d'augmentation de données, qui sont la déformation et la dilatation. au final en consacrant 80% du dataset à l'entraînement, et 20% à l'évaluation, on obtient les distributions suivantes :

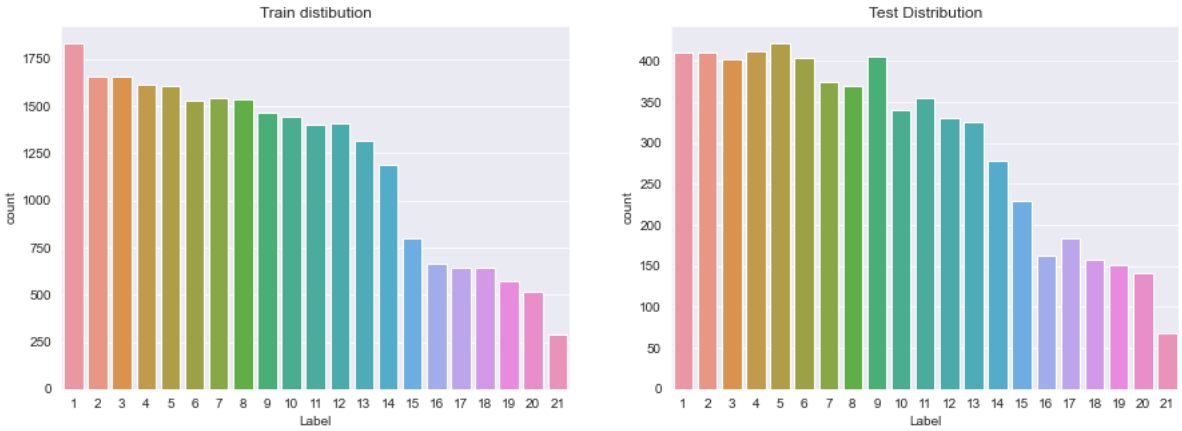

FIGURE 4.1 – Distribution du sous-ensemble d'entraînement (à gauche , Distribution du sous-ensemble d'évaluation (à droite).

Finalement, on précise qu'on a utilisée l'algorithme d'optimisation RMSPROP, l'early stopping, ainsi que l'entropie croisée catégorique pour le calcul du coût.

4.2 Modèles préexistants

Tout les modèles préexistants ont été entraînée au préalables avec les dataset d'ImageNet, puis ils ont été nourris avec notre ensemble de données via un apprentissage par transfert.

4.2.1 VGG16

— **Précision :** 0.94 pour l'entraînement et 0.53 pour l'évaluation.
— **Coût :** 0.4 pour l'entraînement et 18 pour l'évaluation.

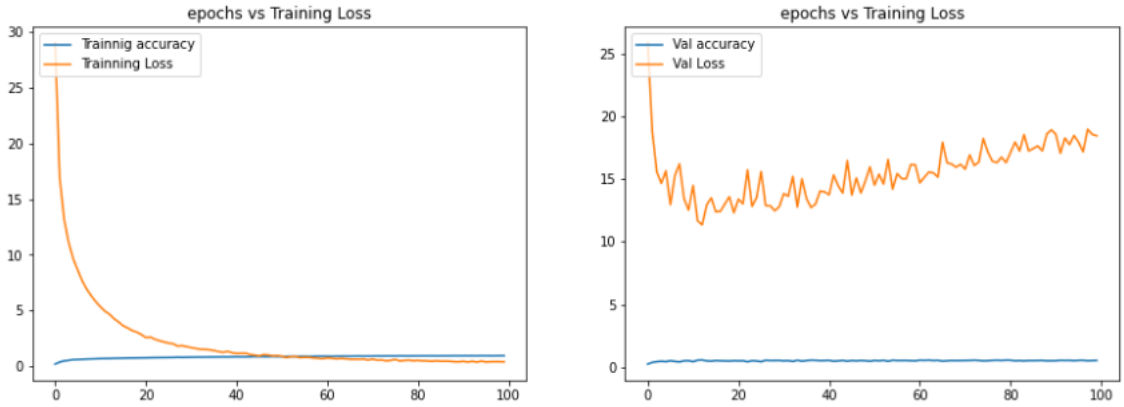

FIGURE 4.2 – Variation du coût et de la précision en fonction de l'itération pour l'entraînement (à gauche) et l'évaluation (à droite) du modèle VGG16

.

4.2.2 Inception v3

— **Précision :** 0.89 pour l'entraînement et 0.53 pour l'évaluation.
— **Coût :** 3.35 pour l'entraînement et 44 pour l'évaluation.

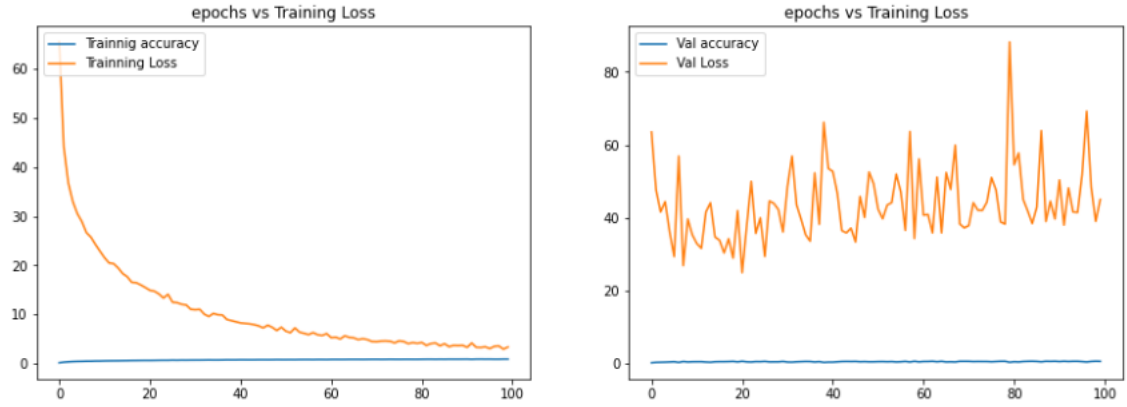

FIGURE 4.3 – Variation du coût et de la précision en fonction de l'itération pour l'entraînement (à gauche) et l'évaluation (à droite) du modèle Inception v3

.

4.2.3 RESNET50

— **Précision :** 0.98 pour l'entraînement et 0.68 pour l'évaluation.
— **Coût :** 0.08 pour l'entraînement et 5 pour l'évaluation.

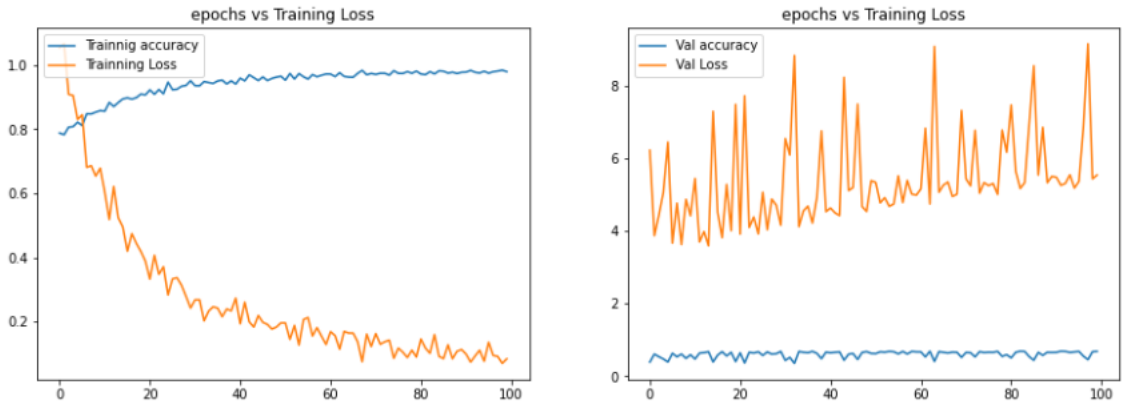

FIGURE 4.4 – Variation du coût et de la précision en fonction de l'itération pour l'entraînement (à gauche) et l'évaluation (à droite) du modèle RESNET50

.

Comme on peut le constater, les modèles préexistants souffre tous d'un problème de sur-apprentissage car les précisions sur les sous ensembles d'entraînement sont tous très élevées, alors que le contraire est vrai pour les sous ensembles d'évaluation.

4.3 Architecture personnelle

— **Précision :** 0.81 pour l'entraînement et 0.77 pour l'évaluation.
— **Coût :** 0.71 pour l'entraînement et 1.2 pour l'évaluation.

Contrairement au modèles préexistants, non seulement notre modèle ne souffre pas d'un problème de sur-apprentissage, mais il possède aussi une précision bien meilleur sur l'ensemble d'évaluation.

4.4 Conclusion

Durant ce chapitre nous avons évaluée tout les modèles mentionnées, et nous les avons comparées, au final, on a trouvée que notre solution est la plus optimale.

Chapitre 5

Conclusion

Au cours de ce projet, nous avons mentionnées les différents modèles populaires qui sont souvent utilisées pour l'OCR, ainsi que donner une description générale de leur architectures

Nous avons également présenté notre propre architecture et expliquée en détails ces différentes couches d'extraction de caractéristiques ou de classification.

Enfin, nous avons évaluée tout les modèles mentionnées, et nous les avons comparées, pour conclure au final que notre modèle a été le plus performant.