RECONNAISSANCE DE CARACTÈRES MANUSCRITS ARABES BASÉ SUR RÉSEAUX DE CONVOLUTION.

LAGSOUN Abdel Motalib[1]

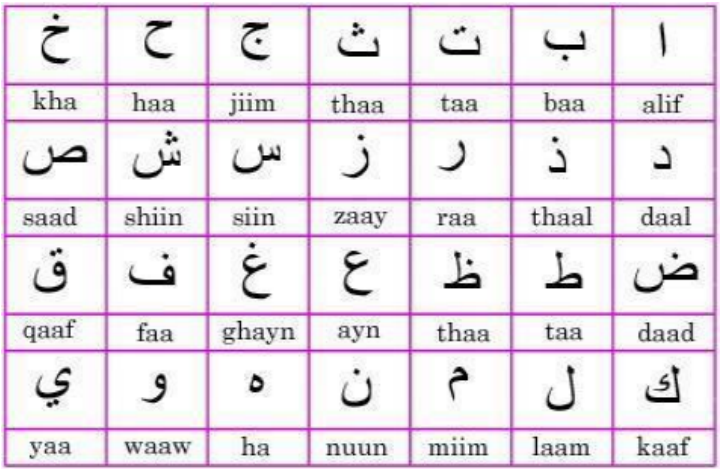
Résumé

L'analyse et la reconnaissance automatiques des caractères manuscrits arabes hors ligne à partir d'images sont un problème important dans de nombreuses applications. Même si avec l'évolution des recherches dans le domaine vision par ordinateur il y'a encore des problèmes non résolus en particulier pour les caractères arabes. L'utilisation des réseaux de neurones profonds peut garantir une solution solide à certains de ces problèmes. Dans ce document nous présentons un réseau neuronal profond pour le problème de reconnaissance de caractères arabes manuscrits qui utilise des modèles de réseau neuronal convolutif (CNN) avec des paramètres de régularisation et des couches dropout pour éviter le sur ajustement. Nous avons appliqué ce modèle sur un ensemble de données composé de 16800 caractères écrits par 60 participants, la tranche d'âge se situe entre 19 et 40 ans et 90% des participants sont de droite. Chaque participant a écrit chaque caractère (de «alef» à «yeh») dix fois, La base de données est partitionnée en deux ensembles: un ensemble d'apprentissage (13 440 caractères à 480 images par classe) et un ensemble de test (3 360 caractères à 120 images par classe). Dans une section expérimentale, nous avons montré que les résultats étaient prometteurs avec un taux de précision de classification de 93.84% sur les images de test.

1. Introduction :

Le domaine de la reconnaissance optique de caractères (OCR) est très important, en particulier pour les systèmes de reconnaissance manuscrite hors ligne. La reconnaissance optique de texte arabe connaît un développement lent par rapport aux autres langues.
Un problème avec la reconnaissance de l'alphabet arabe est que de nombreux caractères ont des formes similaires mais avec des emplacements de points variables par rapport à la partie principale du caractère. La figure-1 montre l'alphabet isolé de la langue arabe.
Comme on peut le voir au milieu, les deux caractères "ayn" et "ghayn" ont une partie principale similaire mais le point sur "ghayn" le est au-dessus tandis que, pour le"ayn" n'a pas du tout de points.
Il est à noter que les caractères manuscrits sont plus difficiles, car les écrivains humains ont tendance à combiner des points et à utiliser des tirets à la place ou à changer la forme des caractères comme le montre la figure 2, qui montre 48 échantillons manuscrits de la même lettre «Ayn»

[1] Etudiant en deuxième année master SDBD à ENSIAS, Rabat, Maroc, Email : abdelmotalib_lagsoun@um5.ac.ma



خ	ح	ج	ث	ت	ب	ا
kha	haa	jiim	thaa	taa	baa	alif
ص	ش	س	ز	ر	ذ	د
saad	shiin	siin	zaay	raa	thaal	daal
ق	ف	غ	ع	ظ	ط	ض
qaaf	faa	ghayn	ayn	thaa	taa	daad
ي	و	ه	ن	م	ل	ك
yaa	waaw	ha	nuun	miim	laam	kaaf

figure-1 alphabet arabe

figure-2 Échantillon de 48 lettres «Ayn» manuscrites.

Le Deep Learning (DL) est une nouvelle application de l'apprentissage automatique pour l'apprentissage de la représentation des données. Les algorithmes DL ont pris la première place dans le domaine de la reconnaissance d'objets en raison de leurs excellentes performances
l'amélioration qu'ils ont apportée.
Les réseaux de neurones convolutifs (CNN) sont un type de réseaux de neurones qui sont appliqués dans de nombreux domaines et fournissent des solutions efficaces à de nombreux problèmes, comme certaines applications de reconnaissance d'objets et de reconnaissance vocale. Cependant, les solutions CNN DL nécessitent de nombreux exemples d'entraînement, ce qui impose des exigences de calcul au

système. Néanmoins, l'accélération des progrès et la disponibilité de matériel informatique à bas prix, ont encouragé l'utilisation de techniques coûteuses en calcul.
Il existe plusieurs frameworks pour le Deep Learning. L'une des bibliothèques les plus populaires est TensorFlow, qui a été publiée par Google en 2015. C'est un code open source écrit en langage de programmation C ++ et capable d'utiliser très bien les GPU. Un autre framework plus simple est Keras , qui est une API de plus haut niveau construite sur TensorFlow. Keras utilise Python pour la programmation, ce qui rend l'écriture de programmes plus facile que les codes TensorFlow natifs.

Par conséquent, dans cet article, nous discuterons de la construction d'un modèle CNN DL pour résoudre notre problème en utilisant TensorFlow/Keras. Ce modèle devrait donner de bons résultats et peut être appliqué efficacement sur des bases de données énormes et différentes.

2. Travaux connexe:

Les algorithmes conçus pour reconnaître les caractères manuscrits ont encore moins de succès que ceux des caractères imprimés, principalement en raison de la diversité des formes et des formes des caractères manuscrits. La reconnaissance des caractères arabes est un problème important, car c'est une étape qui peut être nécessaire dans le problème plus difficile de reconnaissance de mots ou de phrases en arabe. La segmentation des caractères pour séparer le mot en caractères est un autre problème difficile. Diverses méthodes ont été proposées et des taux de reconnaissance élevés sont signalés pour les caractères manuscrits anglais et chinois. Cependant, dans cette section, nous allons présenter uniquement les travaux en relation avec notre sujet.
Les caractères arabes ont des formes différentes selon l'emplacement des caractères dans le mot. Les modèles de Markov cachés (HMM) supposent que chaque lettre est un état et l'utilisation du contexte conduit à une meilleure classification du mot manuscrit arabe comme dans. Néanmoins, des travaux récents discutent des limites de HMM en termes de nécessité d'extraction manuelle des fonctionnalités, qui nécessite une connaissance préalable du langage et est robuste à la diversité et à la complexité de l'écriture manuscrite. L'utilisation de réseaux de mémoire bidirectionnelle à long terme (BLSTM) s'est avérée utile dans d'autres langues, mais l'application à la langue arabe est d'un grand intérêt. Cependant, le manque d'ensembles de données très volumineux et la mise en page du texte arabe posent des problèmes de mise en œuvre. On peut également utiliser la segmentation des caractères suivie de la reconnaissance. Pour ce dernier, ils ont utilisé LSTM avec convolution pour construire des boîtes englobantes pour chaque caractère. Nous transmettons ensuite les caractères segmentés à un CNN pour classification, puis reconstruisons chaque mot en fonction des résultats de la classification et de la segmentation. Il a été montré que la segmentation des caractères avait donné de meilleures performances, confirmant l'intuition que la portée beaucoup plus petite du problème de représentation initiale des caractéristiques du modèle pour les caractères par opposition aux mots et le problème d'étiquetage final ont contribué à améliorer les performances.
En 2015, Lawgali a publié une enquête sur la reconnaissance des caractères arabes et aucun des algorithmes mentionnés n'utilisait l'apprentissage en profondeur. Cependant, en 2015, Elleuch a introduit une reconnaissance de caractères manuscrits en arabe à l'aide de Deep Belief Neural Networks. Il ne nécessite aucune ingénierie des fonctionnalités. L'entrée est simplement les données brutes ou les valeurs de pixels en niveaux de gris des images. L'approche a été testée sur la base de données HACDB qui contient 6600 formes de caractères manuscrits écrits par 50 personnes. L'ensemble de données est divisé en un ensemble d'apprentissage de 5280 images et un ensemble de tests de 1320 images. Le résultat était prometteur sur la tâche de reconnaissance de caractères avec

une précision de 97,9% mais décourageant sur la base de données de reconnaissance de mots avec une précision inférieure à 60%.
En 2017, El-Sawi et al. ont rassemblé l'ensemble de données de caractères manuscrits arabes (AHCD) de 16800 images de caractères isolés. Ils ont construit une architecture CNN Deep Learning pour former et tester l'ensemble de données. Ils ont utilisé des méthodes d'optimisation pour augmenter les performances de CNN. Leur proposition CNN a donné une précision de classification moyenne de 94,9% sur les données de test.

3. Architecture utlisé:

Nous appliquerons une architecture CNN aux bases de données de caractères arabes avec suffisamment d'échantillons. On utilise la capacité CNN pour extraire des caractéristiques et s'entraîner à la reconnaissance. De plus, nous utiliserons des techniques d'optimisation et de régularisation telle que régularisation L2, et des couches dropout, et pour le taux d'apprentissage a été par défaut de l'optimiseur adam.
Les CNN peuvent convertir la structure d'entrée à travers chaque couche du réseau de manière transparente pour extraire automatiquement les caractéristiques des images.

Les CNNs sont basés sur une opération mathématique appelée convolution. Une convolution est une opération de multiplication de chaque pixel de l'image avec chaque valeur dans le noyau, qui est à son tour une autre matrice puis additionnant les produits.
Le principal avantage de l'utilisation de l'opération de convolution est de générer de nombreuses images à partir de l'image originale qui améliorent différentes caractéristiques extraites de l'image originale, ce qui conduit à rendre le processus de classification plus puissant.

Dans CNN, nous utilisons différents types de couches comme cela sera expliqué brièvement. Tout d'abord, la couche de convolution extrait les entités de l'image d'entrée. Au départ, CNN ne sait pas où exactement les caractéristiques (formes) de l'image seront situées; ainsi, il essaie de les trouver partout dans l'image en utilisant une matrice appelée filtre. Chaque filtre représente une fonctionnalité spécifique. CNN applique l'opération de convolution par un filtre glissant dans l'image et multiplie chaque pixel de l'image par chaque valeur du filtre. Ensuite, cette opération est répétée pour d'autres fonctionnalités (filtres) et la sortie de cette couche sera un ensemble d'images filtrées

La couche pooling réduit la dimensionnalité de chaque image filtrée, mais préserve les caractéristiques les plus importantes de la couche précédente. La mise en commun peut être de différents types: maximum, moyen, somme,… etc. La sortie aura le même nombre d'images, mais elles auront chacune moins de pixels. Ceci est également utile pour gérer la charge de calcul. L'opération pooling est illustrée à la figure 3.

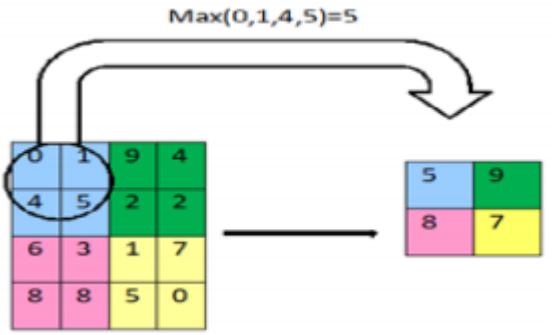


figure-3 pooling.

Les couches dropout sont également utilisées dans les réseaux de neurones convolutifs dans le but de réduire le surajustement. Cette couche «supprime» un ensemble aléatoire de neurones dans cette couche en définissant leur activation à zéro. Il garantit que le réseau peut se généraliser pour tester les données en obtenant des pondérations insensibles aux échantillons d'apprentissage. Dropout est utilisé pendant l'entraînement avec différents pourcentages du nombre total de neurones dans chaque couche.

Enfin, les couches fully connected sont les éléments de base des réseaux de neurones traditionnels. Ils traitent l'entrée comme un vecteur au lieu de tableaux à deux dimensions. Une connexion complète implique que chaque neurone de la couche précédente est connecté à chaque neurone de la couche suivante. La sortie des couches convolutives et de regroupement représente des entités de haut niveau et des couches fully connected utilisées pour classer (images d'entrée) dans la classe appropriée en fonction de l'apprentissage de l'ensemble de données.

Pour notre réseau nous avons conçu trois couches convolutives suivies de deux couches fully connected en tant que couches cachées. Après chaque couche convolution, une couche max pooling est placée. Les premières couches sont les couches d'entrée qui prennent une entrée de forme 32x32 pixels de caractères en niveaux de gris Fonction d'activation ReLU et régularisation L2. et les deux dernières couches suivi par dropout de 0.2 utilisé avant couche sortie de 29 neurons pour pour correspondre au nombre de classes (Alphabet arabe). Nous avons utilisé la fonction d'activation Softmax pour générer des probabilités comprises entre 0 et 1 pour chaque classe représentant la confiance qu'un certain caractère appartient à une classe spécifique.

Pour mettre à jour les poids pendant l'entraînement, nous avons utilisé (sparse categorical crossentropy) comme fonction de coût qui est la fonction de coût appropriée pour les problèmes de classification multi-classes. Nous avons utilisé Adam Optimizer pour trouver les minima de la fonction de coût avec un taux d'apprentissage fixe.
la figure suivante montre l'architecture de notre model :

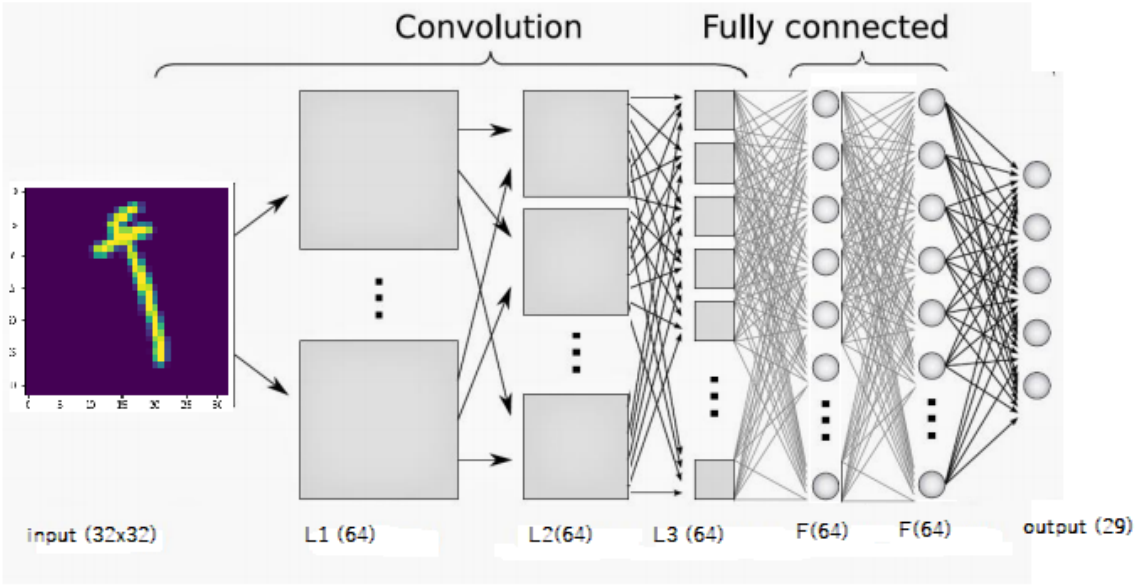


Figure-4 architecture du modèle.

4. mise en oeuvre :

- Dataset:

Les données utilisées sont écrites par 60 participants, la tranche d'âge est de 19 à 40 ans et 90% des participants sont droitiers. cette base de données existe sur Kaggle [2].
La base de données est partagée en deux ensembles: un ensemble d'apprentissage (13 440 caractères à 480 images par classe) et un ensemble de tests (3 360 caractères à 120 images par classe).

```
X_train=pd.read_csv("C:/Users/hp/Desktop/ml1/archive_2/csvTrainImages.csv", header = None)
X_test=pd.read_csv("C:/Users/hp/Desktop/ml1/archive_2/csvTestImages 3360x1024.csv", header = None)
print('train label shape :', X_train.shape)
print('test label shape :', X_test.shape)
```

```
train label shape : (13440, 1024)
test label shape : (3360, 1024)
```

Figure-5 distribution base donnée .

- Résultats :

Après plusieurs expériences notre modèle a donné comme précision 93.84% sur base de données test. dans la phase d'entraînement on a divisé les données entrainement en donné 75 % apprentissage et 25% donné de validation.

La figure 6 montre que les précisions d'apprentissage et de validation ont changé pendant l'entraînement pendant 18 époques sur un jeu de données de 10k. Il est évident que l'écart entre les

[2] les donné utilisé existe via le lien : https://www.kaggle.com/mloey1/ahcd1

deux métriques était insignifiant après les époques initiales. Le petit saut à l'époque 11 et 16 était une indication de la fonction d'arrêt précoce qui évite le surajustement et l'arrêt de l'entraînement.

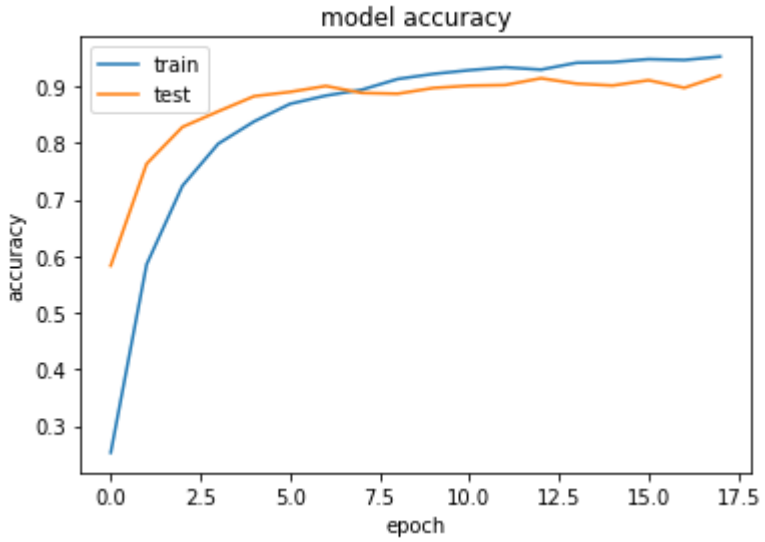


figure 6 : Précision de train_data et précision de la validation_data pendant l'entrainment du modèle

La figure 7 montre les courbes de perte d'apprentissage et de validation en tant que fonctions des époques d'apprentissage. Il est à noter que la perte de validation a diminué de manière significative à l'époque 3 et est restée très faible jusqu'à la fin de l'époque 11.

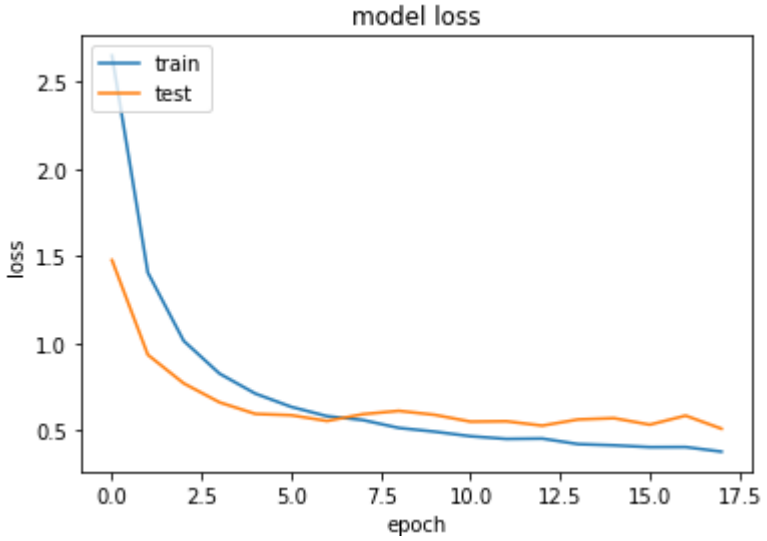


Figure 7 : Perte d'entraînement et de validation.

- tester le modèle :

Pour bien tester le modèle, on a créé une interface graphique Tkinter, d'après cette interface on peut dessiner des lettres arabe, et voir le résultat à l'instant de la prédiction, exemple de l'interface à la figure 8.

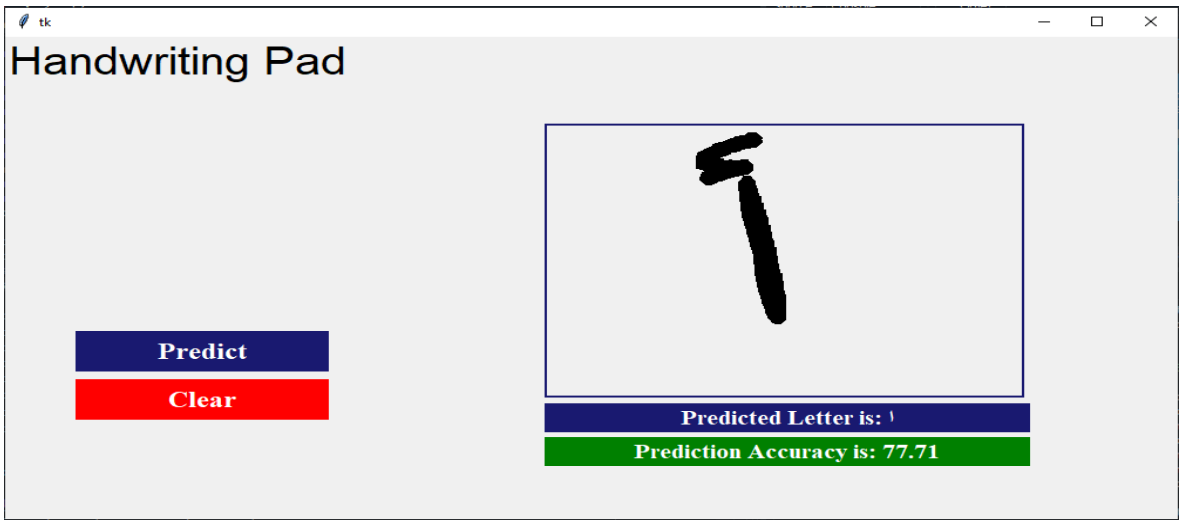


figure 8 : interface Tkinter

5. conclusion :

Les systèmes logiciels automatisés pour la reconnaissance des caractères arabes pourraient avoir d'énormes applications dans de nombreux secteurs industriels et gouvernementaux. Dans cet article, nous avons présenté un système de Deep Learning basé sur réseau de neurones convolutif capable de classer les caractères manuscrits arabes avec une précision de classification 93.33% sur l'ensemble de données utilisé.
ce graphe figure-9 montre une différence entre le nombre des caractères prédit par rapport au valeur predict sur la base de données test.

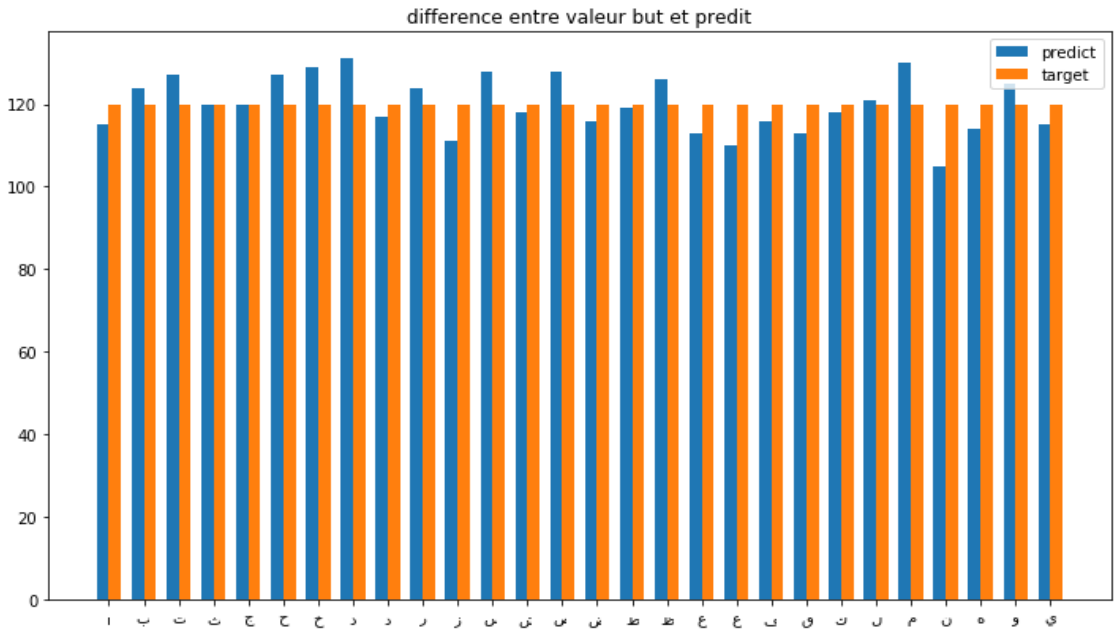


figure 9 : différence entre nombre des caractères prédit nobre attendue

d'après ce graphe on voit que notre modèle a pu prédire à 100% les deux caracter “thaa” et “jiim”, et il y’a d’autre classe qui sont un peu près à une prédiction 100% comme “taa”, “siin” et “laam”.
quelques résultats des nouvelles données par l’interface graphique.
Alif : (Acc.=80.26)
Baa : (Acc.=80,30)
Haa : (Acc.=13,25)
Haa : (Acc.=13,25)
Daal : (Acc.=27,37)
Raa : (Acc.=51,29).
Le système utilise certaines techniques de régularisation (Dropout), qui améliorent les performances et l'efficacité. Le système a été implémenté à l'aide du framework TensorFlow et Keras.

Pour un future travail, je veux utiliser mon model pour plusieur projet comme créé un éditeur de text arabe, le saisir ne serait pas par clavier mais par le canvas créé dans ce projet, et aussi je réfléchi à un projet web scraping et la recherche se fait aussi par le canvas.

RÉFÉRENCES

[1] le lien de la base de données utlisé https://www.kaggle.com/mloey1/ahcd1.

[2] A. Lawgali, "A Survey on Arabic Character Recognition, " International Journal of Signal Processing, Image Processing and Pattern Recognition, vol. 8, no. 2, pp. 401-426, 2015.

[3] https://www.tensorflow.org/api_docs.

[4]https://keras.io/guides/

[5]https://stackoverflow.com/