République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieure et de la Recherche Scientifique
Université Ahmed Draïa d'Adrar
Faculté des Sciences et de la Technologie
Département de Mathématiques et Informatique

Mémoire de Fin d'étude
En vue de l'Obtention du Diplôme de Master en Informatique
Spécialité : Systèmes Intelligents
Thème :

COMBINAISON SVM ET APPRENTISSAGE PROFOND POUR LA RECONNAISSANCE DE CARACTERES ARABE

Préparé par : BAHRI Ismail
LAIB Moussa

Encadrer par : Mr. MAMOUNI El Mamoun

Année universitaire :2021/2022

Remerciements

Nous tenons tout d'abord à remercier Dieu le tout puissant, qui nous a donné la force, la patience et l'opportunité d'accomplir ce Modeste travail.

Au terme de la rédaction de ce mémoire, je tiens à remercier notre Encadrant Mr MAMOUNI pour son aide durant toute la période de travail Ses précieux conseils qui nous ont assistés pour l'accomplissement de notre projet.

Enfin, nous tenons à exprimer nos sincères remerciements à tout le personnel de département informatique surtout les enseignants qui nous ont formé durant toutes nos années d'étude.

Nous remercions tout qui mon aides de près au loin de réaliser ce travail.

Merci

Dédicace

Je dédie ce modeste travail :

À mes chers parents : à mon père : qui a m'encouragé, à ma mère : la femme qui m'a donnée la vie et qui n'a épargné aucun effort pour me satisfaire.

Je prie le bon Dieu de les bénir, de veiller sur eux, en espérant qu'ils seront toujours fiers de moi.

À mes sœurs

À mes neveux et mes nièces

À tous les membres de ma famille

À tous mes professeurs

Je n'oublie pas enseignant : Mr MAMOUNI

À toutes mes amies et mes collègues

À tous qui me connait de près ou de loin.

Ismail

Dédicace

Je dédie ce modeste travail :

À mes chers parents : à mon père : qui a m'encouragé, à ma mère : la femme qui m'a donnée la vie et qui n'a épargné aucun effort pour me satisfaire.

Je prie le bon Dieu de les bénir, de veiller sur eux, en espérant qu'ils seront toujours fiers de moi.

A Ma Femme

A mes chers frères et sœurs

À mes neveux et mes nièces

À tous les membres de ma famille

À tous mes professeurs

Je n'oublie pas enseignant : Mr MAMOUNI

À toutes mes amies et mes collègues

À tous qui me connait de près ou de loin.

MOUSSA

Résumé

Notre travail consiste à construire un système de reconnaissance des caractères arabes manuscrits à l'aide de réseaux de neurones convolutifs (CNN) et de machines à vecteur support (SVM), où nous avons proposé 03 modèles : un modèle pour chaque technique et un modèle hybride qui combine entre les deux., ensuite on a fait la comparaison des résultats en choisissant les meilleurs paramètres pour les deux techniques.

Pour faire ce travail, nous avons utilisé la base de données de caractères arabes manuscrits AHCD constituée de 16 800 caractères écrits par 60 participants, les résultats obtenus sont très encourageants.

Mots-clés : Reconnaissance de l'écriture manuscrite, caractères arabes, classification, SVM, CNN, l’extraction de caractéristiques, AHCD.

Abstract

Our work consists in building a handwritten Arabic character recognition system using convolutional neural networks (CNN) and support vector machines (SVM), where we used 03 models : one model for each technique and hybrid model for both, then we compared the results by choosing the best parameters for the two techniques.

To do this work, we used the AHCD handwritten Arabic character database consisting of 16800 characters written by 60 participants, the results obtained are very encouraging.

Keywords: Handwriting recognition, Arabic characters, classification, SVM, CNN, feature extraction, AHCD.

ملخص

يتمثل عملنا في بناء نظام للتعرف على الحروف العربية المكتوبة بخط اليد باستخدام الشبكات العصبية التلافيفية (CNN) وآلات ناقلات الدعم (SVM)، حيث استخدمنا ثلاثة نماذج: نموذج واحد لكل تقنية ونموذج هجين لكليهما، ثم قمنا بمقارنة النتائج باختيار أفضل المعلمات للتقنيتين.

للقيام بهذا العمل، استخدمنا قاعدة بيانات الأحرف العربية المكتوبة بخط اليد AHCD والتي تتكون من 16800 حرفًا كتبها 60 مشاركًا، وكانت النتائج التي تم الحصول عليها مشجعة للغاية.

الكلمات المفتاحية: التعرف على خط اليد ، الحروف العربية ، التصنيف ، SVM، CNN ، استخراج الميزات ، AHCD.

Sommaire :

Chapitre 01 : La reconnaissance optique de caractères(OCR)

Chapitre 02 : Machines à vecteurs support

Chapitre 03 : Le Réseau de neurones convolutif

Chapitre 04 : Implémentation et Résultats

Liste des tableaux :

Liste des figures :

Introduction général

L’écriture manuscrite joue un rôle important dans la vie quotidienne, non seulement comme moyen de communication, mais aussi pour la documentation juridique. Peu importe à quel point le monde devient numérique, L’écriture manuscrite restera toujours. Même si le monde évolue vers une ère numérique, il existe encore des scénarios où l'utilisation de papier et de stylo ne peut être évitée. Différents systèmes de reconnaissance de caractères sont ainsi appliqués pour identifier les informations manuscrites correctes.

Par conséquent, les technologies de reconnaissance de caractères fournissent aux utilisateurs un mécanisme automatique pour reconnaître le texte sur l'image et convertir les caractères dans leur format numérique correspondant. Ils sont utilisés dans de nombreuses applications de vérification, par exemple, la vérification des documents officiels et des chèques bancaires et aider les personnes malvoyantes à lire, par exemple, la lecture du papier-monnaie ou des panneaux de signalisation. Le système de reconnaissance de caractères peut être utilisé pour lire les scripts dactylographiés et manuscrits. L'écriture manuscrite varie, en particulier dans l'écriture cursive, où la taille et le style des caractères manuscrits des écrivains varient. Par conséquent, la reconnaissance de l'écriture manuscrite est considérée comme une tâche plus difficile en vision par ordinateur (computer vision).

Plusieurs algorithmes d'apprentissage en profond ont déjà été appliqués dans le domaine de la recherche sur la reconnaissance de l'écriture manuscrite. Ils offrent de meilleures performances que les méthodes traditionnelles et prennent moins de temps à reconnaître, en particulier lorsqu'une grande quantité d'ensemble de données est impliqué. Ce processus de reconnaissance utilise des images manuscrites comme ensemble de données. Tout d'abord, les caractéristiques de l'image sont extraites, puis une classification est effectuée. Les caractères, les chiffres, les textes cursifs sont principalement utilisés dans ce processus de reconnaissance. Cependant, le problème est que la majorité de ces expériences sont réalisées en caractères latins, par exemple en anglais, car c'est la langue la plus populaire au monde.

Par conséquent, il existe de nombreuses applications de reconnaissance de caractères disponibles pour la langue anglaise. D'autre part, l'arabe est l'une des 5 langues les plus parlées dans le monde, avec 315 millions de locuteurs natifs, ce qui signifie que maintenant, plus que jamais, les ordinateurs doivent également comprendre ce que ses locuteurs écrivent. Cependant, en raison des caractéristiques uniques de l'écriture arabe, par exemple la nature cursive, la présence de signes diacritiques.

Dernièrement, La reconnaissance de l'écriture manuscrite en arabe, a été étudiée en profondeur depuis quelques décennies par des chercheurs qui ont utilisé des algorithmes différents, comme Support Vector Machine (SVM), Multi-Layer Perceptron (MLP), Hidden Model Markov (HMM), Deep Networks (DNN), Recurrent Neural Networks (RNN) et Convolutional Neural Networks (CNN), etc. Les résultats ont été variés et satisfaisants. Ces systèmes d'apprentissage automatique (ML) ont démontré leur fiabilité et leurs performances dans un large domaine d'applications ainsi que le triomphe de la reconnaissance optique de caractères (OCR) dans les langues latines et asiatiques. L'inconvénient majeur de ces architectures est le grand nombre de paramètres, donc un sur ajustement peut se produire.

Dans ce travail nous avons centralisé notre but à réaliser un système de reconnaissance de caractères arabe manuscrits, par la combinaison de deux classificateurs très populaires SVM et CNN, et on a essayé d'augmenter le taux de performance dans chacun, en utilisant un langage de programmation très puissant qui est PYTHON 3.9 (anaconda) avec des bibliothèques très riches (Tensorflow, Keras, Numpy, Matplotlib).

Notre travail est organisé en quatre chapitres qui peuvent être résumés comme suit :

Dans le premier chapitre, nous avons présenté un état de l'art sur la reconnaissance de l'écriture arabe manuscrite, la reconnaissance optique de caractères (OCR), les caractéristiques de la langue arabe, ainsi que les domaines d'application de la reconnaissance des caractères arabes imprimés et manuscrits.

Dans le chapitre 2 on va présenter le premier classificateur qui est la machine à vecteurs de support, historique de ce classificateur, principe de fonctionnement, les marges souples, ainsi le domaine d'application et les avantages et les inconvénients de SVM.

Dans le chapitre 3 on va présenter le second classificateur est le réseau de neurones convolutifs, le principe de ce réseau, les couches de réseaux de neurones convolutifs, les architectures CNN les plus populaires et nous avons donnée quelques exemples d'applications.

Le dernier chapitre est consacré à la description et l'implémentation de notre approche de reconnaissance des caractères arabes manuscrits basée sur les deux classificateurs CNN et SVM.

Et enfin, nous terminerons ce mémoire par une conclusion générale qui résume notre travail.

Chapitre 01 :

La reconnaissance optique de caractères(OCR)

1.1 Introduction

L'écriture est une des méthodes importantes plus utilisée dans la communication humaine, l'écriture a donc occupé une place prépondérante dans la vie humaine, Il n'est donc pas surprenant de voir que de nombreux travaux scientifiques se concentrent sur sa reconnaissance automatique, L'écriture est en fait la représentation de la parole par des signes graphiques conventionnels sur une surface, ces signes sont combinées les unes avec les autres Pour nous donner le message que nous voulons livre, ces marques représentent des caractères de langue ou des alphabets, dont le but est de développer des systèmes automatisés capables de lire des textes aussi efficacement que les êtres-humains, pour prendre une décision quant au contenu sémantique du message transmis à partir de sa représentation graphique.

Dans ce chapitre nous allons présenter ce qu'est une reconnaissance de l'écriture, ainsi que les caractéristiques de la langue arabe et ses domaines d'application de la reconnaissance de caractère.

1.2 Reconnaissance optique de caractères(OCR)

La reconnaissance de l'écriture est mieux connue sous le nom d'OCR (Optical Character Recognition) c'est une opération informatique qui permet de traduire des images de texte écrit sur papier en un texte modifiable par machine codé dans un schéma de codage standard (par exemple ASCII, ASMO) [1]. Dans les années 1950, les premières solutions pour reconnaître le texte de la machine à écrire en anglais ont commencé à apparaître, aujourd'hui, Il existe de nombreuses solutions pour reconnaître avec précision le texte latin manuscrit, contrairement les solutions avec le texte arabe manuscrite.

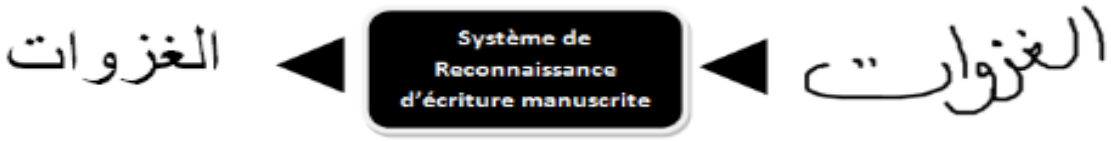

Figure 1.1 : Principe de la reconnaissance d'écriture manuscrite.

La reconnaissance de l'écriture manuscrite peut être classée en deux mode, la reconnaissance d'écriture manuscrite en ligne et hors ligne.

1.2.1 La reconnaissance en-ligne (on-line)

Dans la reconnaissance de caractères en ligne, les caractères sont reconnus en temps réel (pendant l'écriture). L'utilisateur écrit naturellement à l'aide d'un stylet électronique sur un

écran, le logiciel de la reconnaissance interprète les caractères ou les mots écrits pour les transformer en caractères numériques. La reconnaissance en-ligne présente un avantage majeur c'est la possibilité de correction et de modification de l'écriture de manière interactive vu la réponse en continu du système [2]. Les solutions de la reconnaissance de caractères en ligne utilisent l'ordre, la vitesse et la direction des traits de stylet individuels pour obtenir une bonne précision dans la reconnaissance du texte manuscrit. Maintenant, la reconnaissance de caractères en ligne est largement utilisée dans les appareils mobiles tels que les tablettes, les téléphones intelligents et les assistants numériques personnels**.**

Figure 1.2: Exemple de reconnaissance en ligne

1.2.2 Reconnaissance hors ligne (off-line)

Reconnaître un texte manuscrit hors ligne est plus difficile que de le reconnaître en ligne car le premier doit traiter des images deux dimensions du texte après qu'il a déjà été tapé [3], la reconnaissance a lieu après l'étape de numérisation du document papier et n'a pas forcément un but de traitement temps réels comme en reconnaissance en ligne. Après avoir pris l'image de texte dans des formats binaires, les étapes de reconnaissance ressemblent plus ou moins au cas de la reconnaissance d'écriture manuscrite en ligne avec quelques exceptions, le prétraitement de l'écriture hors ligne nécessite différentes activités comme la détection de la ligne du texte d'écriture manuscrite. En plus de cela, il n'y a pas d'informations temporelles attachées à l'image scannée donc, le classificateur n'a pas la moindre information sur la manière

et l'ordre de l'écriture. Bien que les systèmes hors ligne soient moins précis que les systèmes en ligne, ils sont maintenant assez bons pour des systèmes spécialisés tels que l'interprétation des adresses postales manuscrites sur les enveloppes.

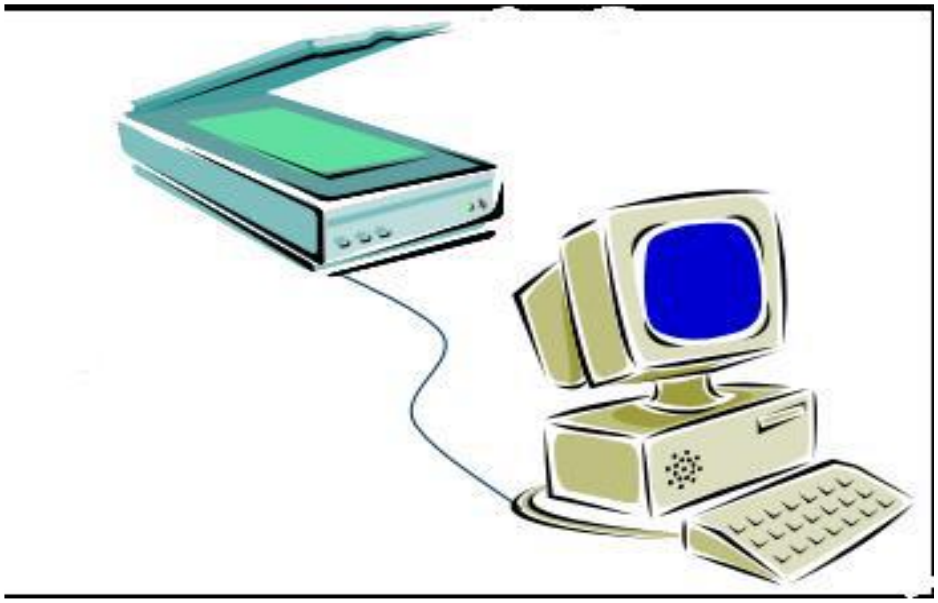

Figure 1.3: Exemple de reconnaissance hors ligne [4]

La reconnaissance hors-ligne peut être classée en plusieurs types :

A) Reconnaissance de texte ou analyse de documents

Dans le premier cas (Reconnaissance de texte), il est question de reconnaître un texte de structure limitée à quelques lignes ou mots. La recherche consiste en un simple repérage des mots dans les lignes, puis à un découpage de chaque mot en caractères [5].

Dans le second cas (analyse de document), il s'agit de données bien structurés dont la lecture nécessite la connaissance de la typographie et de la mise en page du document. Ici la démarche n'est plus un simple prétraitement, mais une démarche experte d'analyse de document il y'a localisation des régions, séparation des régions graphiques et photographique, étiquetage sémantique des zones textuelles à partir de modèles, détermination de l'ordre de lecture et de la structure du document [6].

B) Reconnaissance de l'imprimé ou du manuscrit

Il existe différentes méthodes de reconnaissance des caractères imprimés ou manuscrits. Les caractères imprimés sont dans le cas général aligné horizontalement et séparés verticalement, ce qui simplifie la phase de lecture [5]. bien que certaines fontes présentent

parfois des accolements qu'il faut défaire. De plus, le graphisme des caractères est conforme à un style calligraphique (fonte) qui constitue un modèle pour l'identification. Dans le cas du manuscrit, les caractères sont souvent ligaturés et leur graphisme est inégalement proportionné. Cela exige l'emploi de techniques de délimitation très spécifiques et souvent des connaissances contextuelles pour orienter la lecture.

1.2.3 Comparaison entre la reconnaissance des caractères en-ligne et hors-ligne

Il existe plusieurs différences entre la reconnaissance des caractères en-ligne et hors-ligne. Nous les expliquerons dans le tableau suivant :

La reconnaissance des caractères en-ligne	La reconnaissance des caractères hors-ligne
Utilisation Stylo électronique	Utilisation de papier
Taux de reconnaissance élevé	Taux de reconnaissance bas
Grande précision	Faible précision
Bruit d'image faible	Existence d'un bruit important

Tableau 1.1: Comparaison entre la reconnaissance en-ligne et hors-ligne.

1.3 Reconnaissance globale ou analytique

1.3.1 L'approche globale

Considère le mot en tant qu'entité unique et le décrit indépendamment de ses caractères constitutifs. Cette approche a l'avantage de maintenir le caractère dans son contexte environnant, ce qui permet de mieux modéliser les variations de l'écriture et la dégradation qu'elle peut subir. Cependant, en raison de beaucoup d'informations, la discrimination des mots proches est très difficile, et l'apprentissage des modèles exige une grande quantité d'échantillons qui est souvent difficile à réunir.

1.3.2 L'approche analytique

Dans l'approche de segmentation analytique, des points de segmentation hypothétiques seront générés avant le début du processus de reconnaissance, Où le mot est segmenté en caractères ou en fragments morphologiques significatifs inférieurs au caractère appelés graphèmes. La reconnaissance de mots consiste à reconnaître les entités segmentées, puis à aller vers la reconnaissance de mots, ce qui est une tâche délicate qui peut générer divers types

d'erreurs [7]. Un processus de reconnaissance selon cette approche est basé sur deux phases : la phase de segmentation et la phase d'identification des segments.

1.4 Procédures générale de l'OCR

Les systèmes OCR commencent par l'acquisition d'une image brute, puis procède à sa binarisation, et son codage. Ensuite, les images de l'écriture passent par une phase de segmentation afin d'extraire les segments de base, ces segments sont analysés pour extraire les caractéristiques essentielles. Puis, ces caractéristiques passent par une phase d'apprentissage où l'utilisateur doit introduire un ensemble de données types qui sera utilisé par un classificateur (tel que, CNN : Convolutional Neural Network ou SVM : Support Vector Machine) pour créer des modèles permettant de prendre des décisions au cours de la phase de décision.

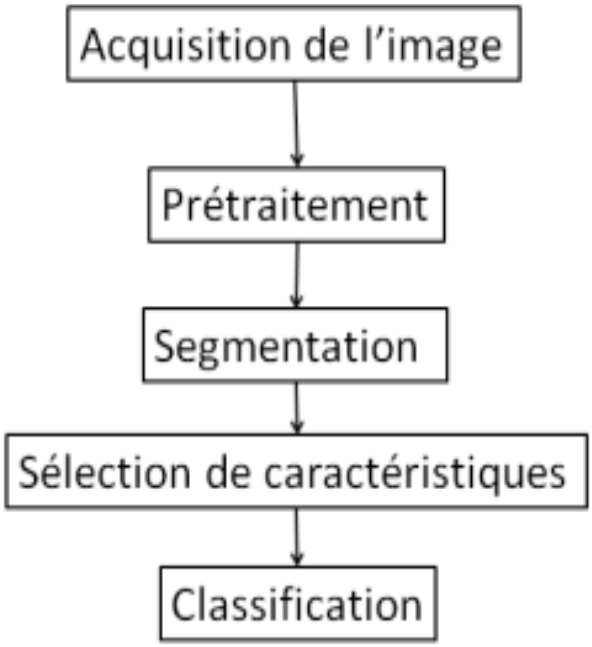

Figure 1.4 : Modèle général pour les systèmes OCR**.**

1.4.1 Acquisition de L'image

Cette étape est scannée le document et le stocker comme une image, elle se fait par balayage optique. Le résultat du balayage optique est enregistré dans un fichier de pixel sa taille dépend de la résolution. Les valeurs de pixel sont différentes selon le type d'image. Les pixels peuvent avoir comme valeurs : 0 (blanc) à 255 (noir) pour des images de niveau de gris, et trois canaux de valeurs de couleurs entre 0 et 255 pour des images en couleur.

1.4.2 Prétraitement

Lors de la saisie du document, du bruit peut se produire pendant ce processus, Il en résulte une mauvaise reconnaissance des caractères. Ce problème survenant généralement est résolu par le prétraitement. Il consiste en un lissage et une normalisation. Dans le lissage, certaines règles sont appliquées au contenu de l'image à l'aide de techniques de remplissage et d'amincissement. La normalisation est chargée de gérer la taille uniforme, l'inclinaison et la rotation des caractères.

1.4.3 segmentation

Dans les systèmes de reconnaissance des caractères la segmentation est une opération très critique. Où il peut être défini comme un processus consistant à décomposer l'image d'un texte en entités (phrases, mots, caractères), qui font partie d'un alphabet, Le problème le plus souvent rencontré dans la segmentation est le suivant : il provoque une confusion entre le texte et les graphiques en cas de caractères joints et séparés, Habituellement, les fissures et les joints dans les caractères sont dus à la numérisation.

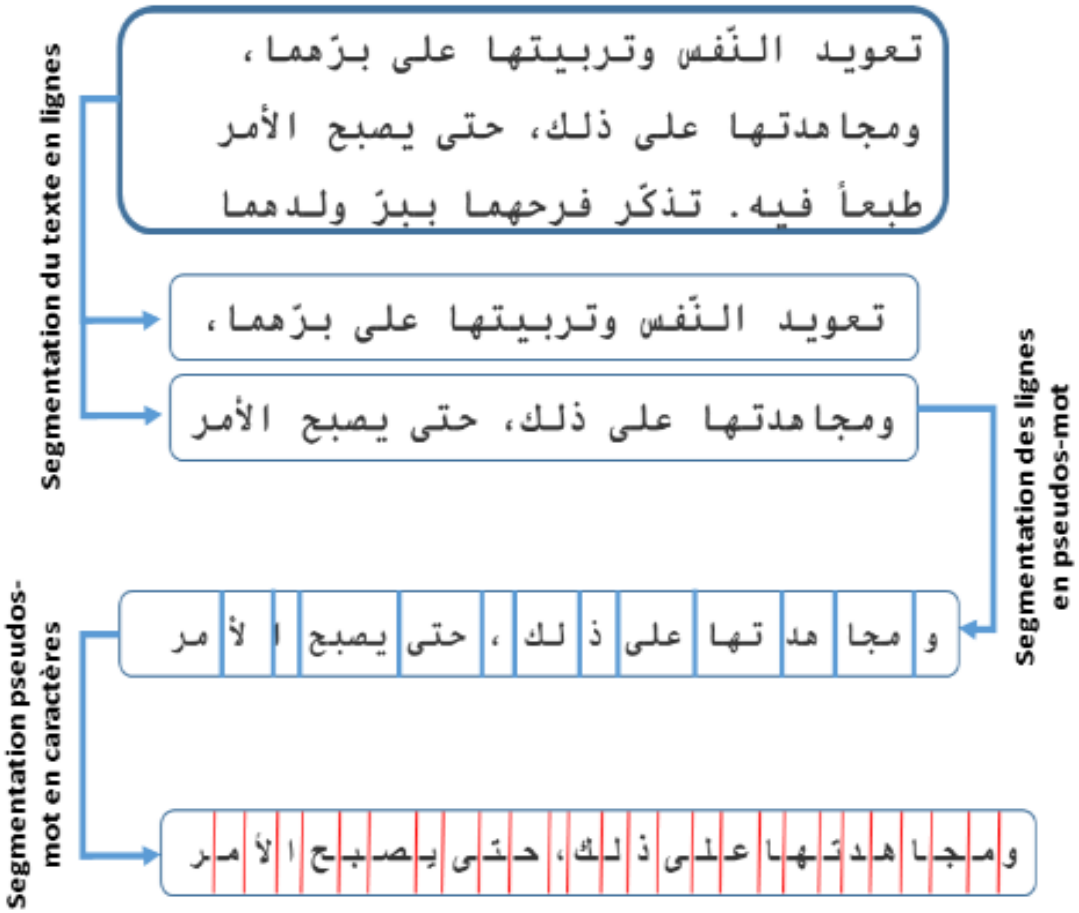

Figure 1.5 : Le processus de segmentation.

1.4.4 Sélection de caractéristique

Le processus d'extraction consiste à extrait des caractéristiques permettant de décrire de façon non équivoque les formes appartenant à une même classe de caractères tout en les différenciant des autres classes. Cette étape est réalisée pour sélection les informations la plus discriminante et de dimension limitée pour l'étape de la classification, toute en évitant le risque de perte des informations importantes et signifiantes [8].

1.4.5 La Classification

La classification est une étape très essentielle de la reconnaissance. C'est aussi une forme d'analyse de données qui extrait des modèles décrivant des classes de données. Cette analyse peut aider à mieux comprendre l'ensemble de données. Plus concrètement, la classification de données passe par deux grandes étapes (L'étape de l'apprentissage et l'étape de la classification).

1.5 Caractéristiques de la langue arabe

La langue arabe est écrite et parlé par plus de trois cent millions de gens, dans plus de vingt (20) pays différents. Elle est classée la cinquième langue parlée mondialement. Elle est née dans la péninsule Arabique aux environs du VIe siècle. Elle est de la famille des langues sémitiques et elle a deux types d'écritures possibles :

- L'écriture classique, pour le Coran et la littérature classique,
- L'écriture universelle du monde arabe actuel.

L'Arabe s'écrit de droite à gauche et est toujours cursif. Il contient 29 lettres de base et huit signes diacritiques, Le tableau1.2 montre les 29 lettres et leurs différentes formes. Chaque lettre a plusieurs formes selon sa position dans le mot. Chaque lettre est dessinée sous une forme isolée lorsqu'elle est écrite seule, et est dessinée sous trois autres formes au maximum lorsqu'elle est écrite en connexion avec d'autres lettres du mot. Par exemple, la lettre Ain a quatre formes: Forme isolée (ع) et Formes initiale, médiale et finale (ععع), respectivement de droite à gauche. De plus, les lettres Hamza, Teh et Alef ont d'autres formes, comme indiqué dans le tableau1.2.

No	Letter Name [a]	Isolated Form	Initial Form	Medial Form	Final Form
1	Hamza [b]	ء	ئ	ئ	أ
2	Beh	ب	بـ	بـ	بـ
3	Teh [c]	ت ة	تـ	تـ	ة ت
4	Theh	ث	ثـ	ثـ	ث
5	Jeem	ج	جـ	جـ	ج
6	Hah	ح	حـ	حـ	ح
7	Khah	خ	خـ	خـ	خ
8	Dal [d]	د	-	-	د
9	Thal [d]	ذ	-	-	ذ
10	Reh [d]	ر	-	-	ر
11	Zain [d]	ز	-	-	ز
12	Seen	س	سـ	سـ	س
13	Sheen	ش	شـ	شـ	ش
14	Sad	ص	صـ	صـ	ص
15	Dad	ض	ضـ	ضـ	ض
16	Tah	ط	طـ	طـ	ط
17	Zah	ظ	ظـ	ظـ	ظ
18	Ain	ع	عـ	عـ	ع
19	Ghain	غ	غـ	غـ	غ
20	Feh	ف	فـ	فـ	ف
21	Qaf	ق	قـ	قـ	ق
22	Kaf	ك	كـ	كـ	ك
23	Lam	ل	لـ	لـ	ل
24	Meem	م	مـ	مـ	م
25	Noon	ن	نـ	نـ	ن
26	Heh	ه	هـ	هـ	ه
27	Waw [d]	و	-	-	و
28	Alef [d, e]	ى ا	-	-	ى ا
29	Yeh	ي	يـ	يـ	ي

Tableau 1.2: Lettres arabes et leurs quatre formes.

Dans un mot, chaque lettre peut se connecter de la droite avec la lettre précédente. Cependant, il y a six lettres qui ne se connectent pas à partir de la gauche avec la lettre suivante (" أ","ز", "ر", "د", "ذ ", "و "). Ces lettres n'ont que les formes isolées et finales. Lorsqu'une de ces six lettres est présente dans un mot, le mot est divisé en sous-mots. Par exemple, le mot arabe (عربية) a deux sous-mots : le premier sous-mot se compose de Initial Ain et Final,

déconnecté à gauche, Reh ; et le deuxième sous-mot se compose du Beh initial, du Yeh médian et du Teh final.

Certaines séquences de lettres ont des ligatures composites spéciales lorsqu'elles se présentent en un seul mot. Par exemple, Meem suivi de Hah est souvent dessiné (محمد) plutôt que (محمد).

Dans l'alphabet arabe, 15 lettres possèdent un ou plusieurs points. Ces signes diacritiques sont situés soit au-dessus, soit en dessous de la ligne de base, mais jamais les deux à la fois. Le tabeau 1.3 illustre la variabilité des styles d'écriture des points ou groupes de points en écriture manuscrite arabe.

Position / Nombre des points	**Au-dessus**	**Au-dessous**
Un seul point diacritique	خ غ ف ز ذ ظ ض	ب ج
Deux points diacritiques	ق ت ة	ي
Trois points diacritiques	ش ث	

Tableau 1.3: Points en arabe : un, deux ou trois points.

Mis à part les points, les caractères peuvent avoir d'autres signes diacritiques (tels que Shadda), (Hamza) et (Sukun).Il existe trois voyelles courtes (Fatha, dama et casra) en arabe, ces diacritiques (en dehors des points) sont utilisées comme guide phonétique. La figure I.6 présente un exemple de texte arabe, avec signes diacritiques optionnels.

Figure 1.6 : Exemples de textes en arabe, avec les signes diacritiques optionnels

1.6 Domaines d'application

Voici quelques exemples d'applications informatiques utilisant des systèmes de reconnaissance d´écriture :

- La lecture des tickets de transport aérien : Pour d'éviter l'attente avant embarquement, de nombreuses compagnies ont recours à un système d'identification automatique qui lit le ticket et compare les indications avec la base de données de chaque vol.
- Le bureau de poste pour la lecture des adresses ainsi que le tri automatique du courrier.
- L'administration pour la lecture automatique de documents administratifs.
- Traiter les formulaires manuels : comme un sondage, un bon de commande ou un chèque.
- La lecture des passeports : Certaines douanes sont équipées de lecture de passeports afin d'identifier chaque voyageur. Le système permet de lire le nom, la nationalité, le numéro de passeport ainsi que de vérifier directement l'autorisation de séjour.
- Saisir des formules mathématiques sur l'écran de son Tablet.

1.7 conclusion

Dans ce chapitre, nous avons fait un aperçu sur la reconnaissance d'écriture manuscrite et plus particulièrement dans celui de la reconnaissance de caractères arabe, ainsi que les différents aspects d'un OCR (reconnaissance optique de caractères), Ensuite nous avons abordé les procédures générales de l'OCR. Puis nous avons présenté un aperçu sur les différentes caractéristiques de la langue Arabe. Finalement nous avons donnée quelques exemples d'applications informatiques utilisant des systèmes de reconnaissance d´écriture.

Chapitre 02 :
Machines à vecteurs support

2.1. Introduction

Les machines à vecteurs supports (abrégé en SVM) sont parmi les techniques utilisées pour la reconnaissance de l'écriture manuscrite arabe, et parmi les méthodes à noyaux développée à partir de la théorie de l'apprentissage statistique, et introduites au début des années 90 par V.Vapnik [9], qui ont un grand succès dans de nombreux domaines de l'apprentissage automatique . On peut dire que ces machines sont concurrencer les réseaux de neurones et autres techniques d'apprentissage [10].

SVMs sont des méthodes d'apprentissage supervisé populaire pour la classification, la régression et d'autres tâches d'apprentissage. Cependant, il est principalement utilisé pour les problèmes de classification dans l'apprentissage automatique.

2.2. Historique des machines à vecteurs supports

Il existe une chaîne d'événements qui ont conduit à l'invention de machines à vecteurs de support datant généralement du milieu du XXe siècle :

En 1950, Aronszajn publie la "Théorie de la reproduction des noyaux". En 1957, Frank Rosenblatt reprend cette idée et invente le perceptron, un simple classificateur linéaire.

6 ans plus tard, Vapnik et Lerner arrivent et annoncent "Generalized Portrait Algorithm" (1963). Ce fut la véritable inspiration de l'article de Boser, Guyon et Vapnik de 1992 à la conférence COLT présentant les machines à vecteurs de support. Cependant, ce modèle était linéaire et nous ne savions pas encore comment induire des limites de décision non linéaires.

En 1992, afin d'étendre svm à l'état non linéaire, Boser et al ont suggéré d'introduire un noyau non linéaire [11]. Un autre grand saut entre l'article Generalized Portrait Algorithm (1963) et SVM (1992) a été la théorie de l'apprentissage statistique de Vapnik et Chervonenkis en 1974, puis l'avancement de Vapnik à ce sujet en 1979 [12].

En 1995, Vanpik a trouvé un classificateur de marge souple avec Cortes et l'a étendu à une application de régression la même année [13]. Dans la même année, Cortes et Al ont proposé une version réglementée de SVM qui tolérerait les erreurs d'apprentissage avec des pénalités [14] [15].

En 1998, Shawe, Taylor et al. A contribué de manière significative à la généralisation des machines à vecteurs de support à marge dure. Shawe, Taylor, Cristianini ont ensuite

donné des limites statistiques à la généralisation des machines à vecteurs de support à marge souple en 2000.

2.3. Principe de fonctionnement général

L'objectif de l'algorithme SVM est de trouver une ligne ou une limite de décision, séparateur ou bien un classificateur qui peut de séparer l'espace à n dimensions en classes. Afin que nous puissions facilement placer les nouveaux points de données dans la bonne catégorie à future.

Pour notre exemple on a deux classes, le but est de trouver un classificateur qui va séparer les données et maximiser la distance entre ces deux classes. Avec SVM, ce classificateur est un classificateur linéaire appelé " hyperplan" [16]. Dans le schéma ci-dessous dans lequel deux classes différentes sont classées qui sépare par un hyperplan :

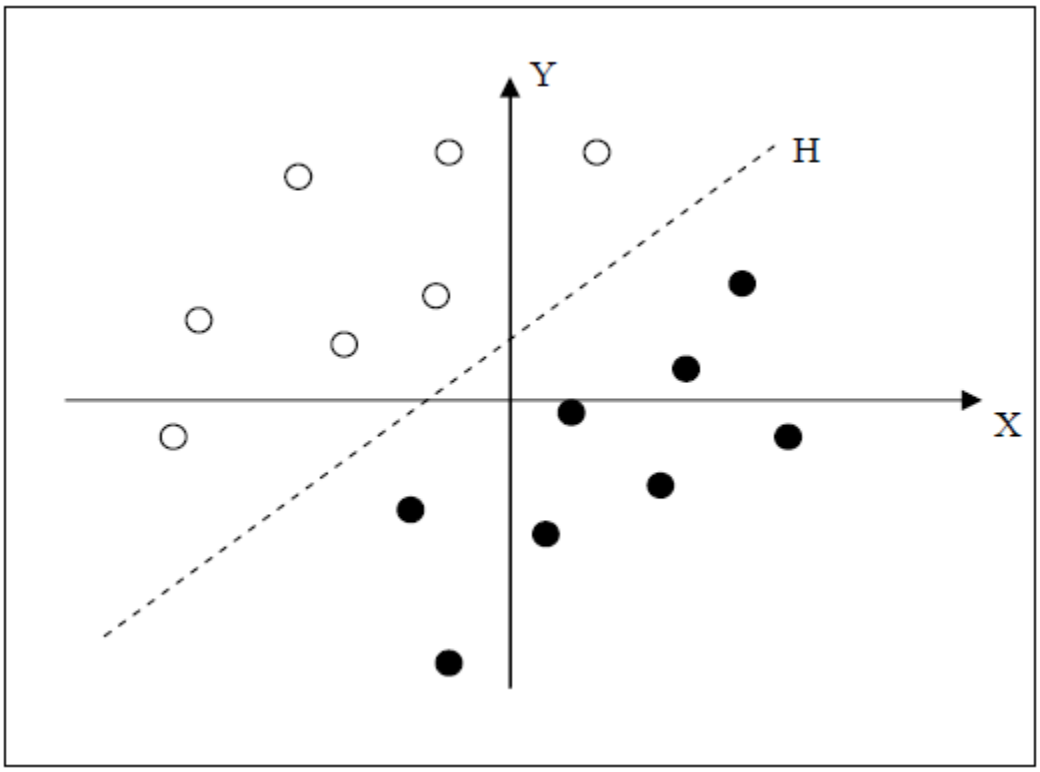

Figure 2.1 : Séparation de deux classes par un Hyperplan H

Les points de données les plus proches de l'hyperplan, qui seuls sont utilisés pour la détermination de l'hyperplan, sont appelés « vecteurs de support ».

Maintenant, dans la figure 2.2 ci-dessous, nous pouvons voir que pour la classification, nous avons de nombreuses d'hyperplans valides, qui sont pouvons de classer l'ensemble de données, les SVM choisissent seulement qui est optimal, mais la question est de savoir quel hyperplan doit être sélectionné pour qu'il soit optimal ?

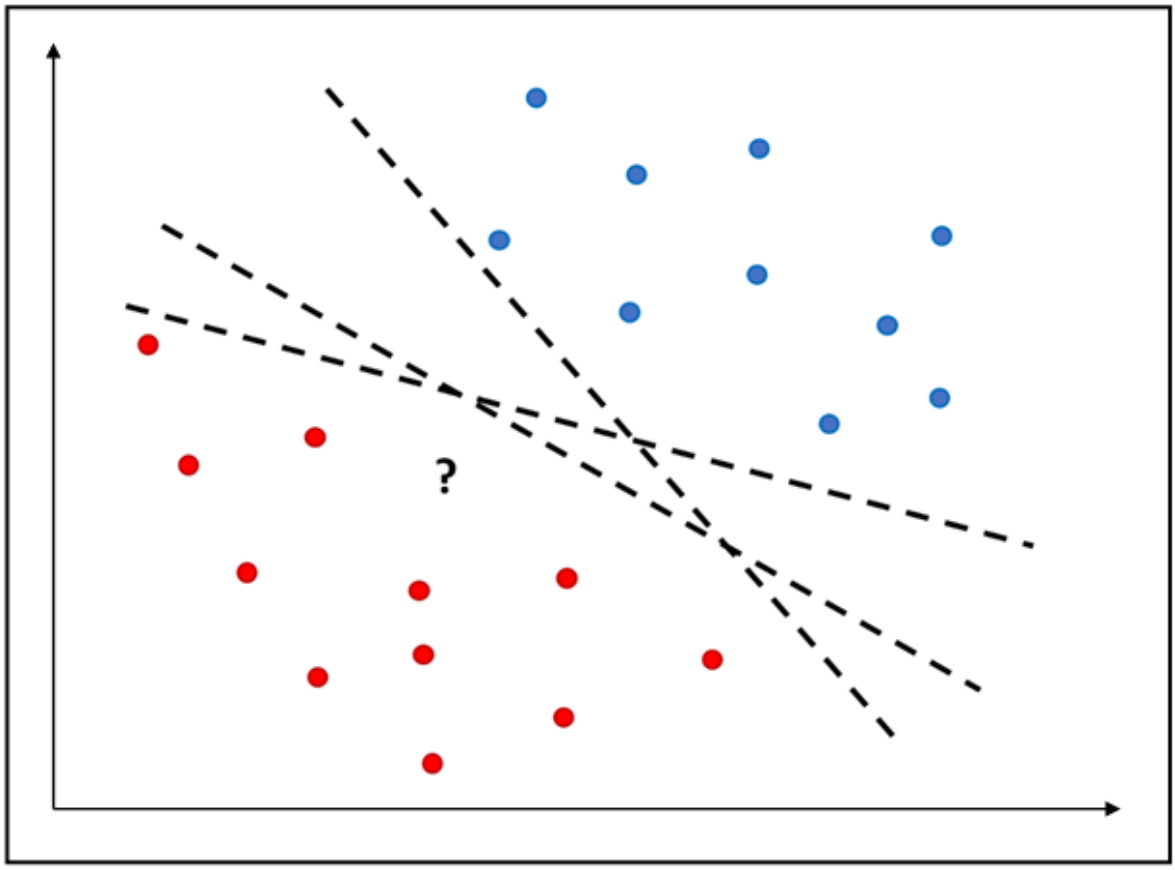

Figure 2. 2 : Problème de séparation linéaire à deux classes : quel est le meilleur hyperplan optimal parmi tous ceux qui séparent les données ?

La distance entre l'hyperplan et les exemples d'apprentissage est appelée « marge », et pour obtenir un hyperplan optimal, il faut maximiser la marge entre les données et l'hyperplan. Comme on cherche à maximiser cette marge, on appelle cela de séparateurs à vaste marge [17]. La marge plus large offre plus de sécurité lorsque l'on classe un nouvel exemple. Dans le schéma qui suit, on détermine un hyperplan optimal qui sépare les deux ensembles de points.

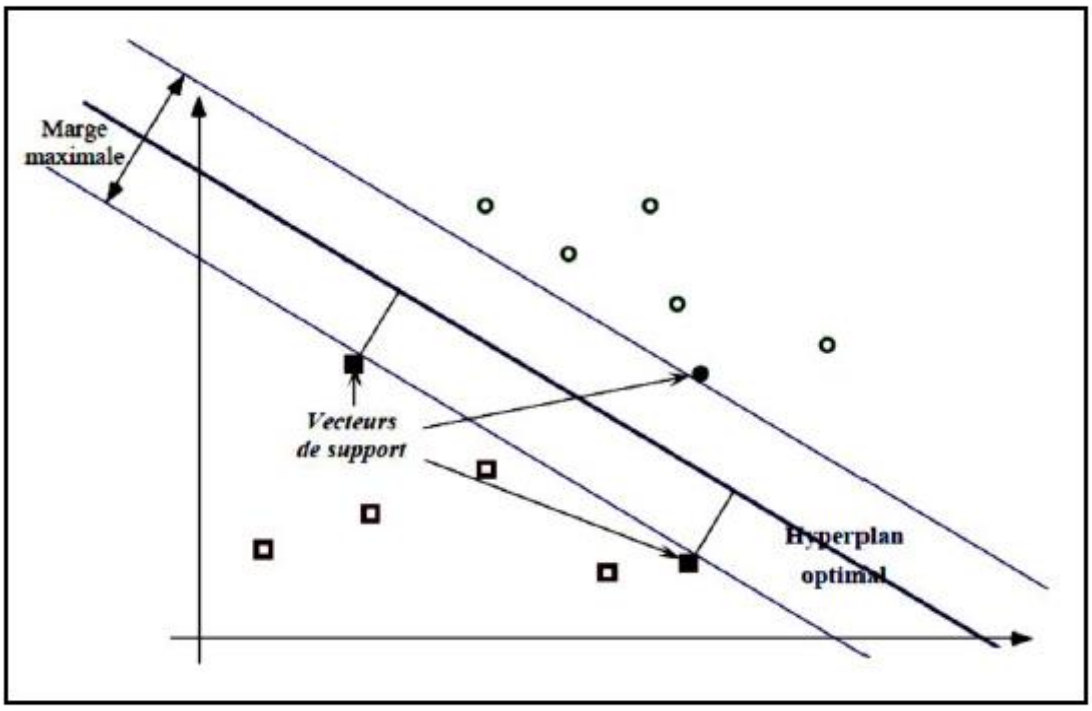

Figure 2.3 : Hyperplan optimal, marge et vecteurs de support.

2.3.1 Linéarité et non linéarité

Dans les SVM, il existe deux cas de séparabilité :

a) Cas linéairement séparable

Dans ce cas les SVM permettent de retrouver facilement le classificateur linéaire (séparateur).

b) Cas non linéairement séparable

Mais dans la plupart des problèmes réels il n'est pas possible de séparer linéairement entre les données, le classificateur de marge maximum ne peut pas être utilisé parce qu'il ne fonctionne que si les classes de données d'apprentissage sont séparables de manière linéaire [18]. (Voir la figure 2.4).

Le principe consiste à projeter les données de l'espace d'entrée (qui appartient à deux classes différentes) qui ne sont pas linéairement séparables dans une espace de dimension plus grande appelé "espace des caractéristiques ou espace de re-description" afin que les données deviennent linéairement séparables.

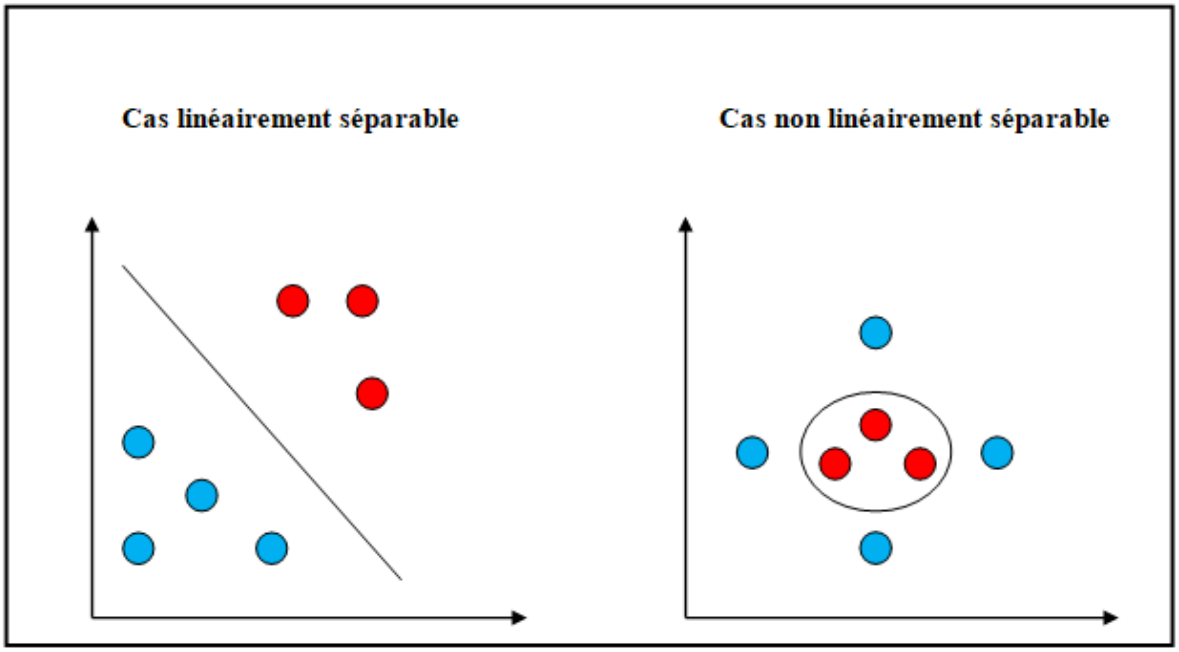

Figure 2.4 : Linéarité et non linéarité.

Nous avons donc une transformation du problème de séparation non linéaire dans l'espace de représentation en un problème de séparation linéaire dans une espace de description de dimension plus grande. Cette transformation s'effectue via la fonction noyau

Intuitivement, plus la dimension de l'espace de re-description est grande, plus la probabilité de pouvoir trouver un hyperplan séparateur entre les exemples est élevée. Ceci est illustré par le schéma suivant : (la figure 2.5).

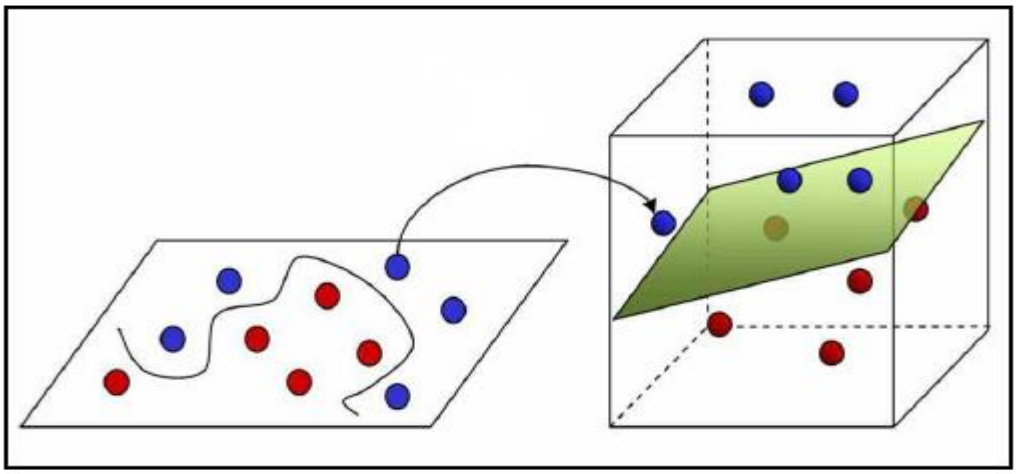

Figure 2.5 : Transformation de l'espace de représentation et l'hyperplan séparateur dans le cas non linéairement séparable.

2.4 Fondements mathématiques

2.4.1 Principe

On s'intéresse à une fonction notée f qui à toute entrée x correspond à une sortie de $y = f(x)$. Le but est d'essayer d'apprendre f à partir d'un ensemble de couple(x_i ,y_i). Dans ce problème les svm sera utilisée pour classifier une nouvelle observation x en se limitant à deux classes $y \in \{-1,1\}$. Nous allons donc construire une fonction f qui à chaque valeur d'entrée dans un ensemble R^d va faire correspondre une valeur de sortie $y \in \{-1,1\}$:

$$f: \mathrm{R}^{\mathrm{d}} \rightarrow \{-1,1\} \qquad f(x) = y$$

Dans le cas linéaire, une fonction discriminante h est obtenue par combinaison linéaire d'un vecteur d'entrée $x = (x_1, \dots, x_d)$ et s'écrit :

$$h(x) = w.x + b \tag{2.1}$$

La classe est donnée par le signe de $h(x)$: $f(x) = sign(h(x))$. si $h(x) \geq 0$ alors x est de classe 1 sinon x est de classe -1. Le séparateur est alors un hyperplan d'équation :

$$w.x + b = 0$$

Si (x_i , y_i) est un des p éléments de la base d'apprentissage notée A_p, on veut trouver le Classifieur h tel que :

$$y_i(w.x_i + b) \geq 0, i \in [1, p] \tag{2.2}$$

Dans le cas simple linéairement séparable il existe plusieurs hyperplans séparateurs. Selon la théorie de Vapnik [19], l'hyperplan optimal est celui qui maximise la marge. Cette dernière étant définie comme la distance entre un hyperplan et les points échantillons les plus proches lequel sont les vecteurs supports. La distance entre un point x quelconque et l'hyperplan est donnée par l'équation suivante :

$$d(x) = \frac{w.x+b}{\|w\|} \tag{2.3}$$

Donc maximiser la marge va revenir à minimiser$\|w\|$.

2.4.2 Forme primale

Les paramètres w et b étant définis par un coefficient multiplicatif près, nous choisissons les normaliser pour que les échantillons les plus proches x_s vérifient l'égalité suivante :

$$y_s(w.x_s + b) = 1 \tag{2.4}$$

Donc quel que soit l'échantillon x_i on obtient :

$$y_i(w.x_i + b) \geq 1 \tag{2.5}$$

Ainsi La distance entre l'hyperplan et un point support est définie par $\frac{1}{\|w\|}$. La marge géométrique entre deux classes est égale à $\frac{2}{\|w\|}$.

La forme primale (qui dépend seulement de w et b) des SVM est donc un problème de minimisation sous contrainte qui s'écrit :

$$\begin{cases} min\left(\frac{1}{2}\|w\|^2\right) \\ \forall(x_i, y_i) \in A_p, y_i(w.x_i + b \geq 1 \end{cases} \quad (2.6)$$

2.4.3 Forme duale

La formulation primale peut être transformée en formulation duale en utilisant les multiplicateurs de Lagrange. L'équation (2.6) s'écrit alors sous la forme suivante :

$$L(w, b, \alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{p} \alpha_i(y_i(w.x_i + b) - 1) \quad (2.7)$$

La formulation de Lagrange permet de trouver les extremums en annulant les dérivées partielles de la fonction $L(w, b, \alpha)$. Le lagrangien L doit être minimisé par rapport à w et b et maximisé par rapport à α. On résout ce nouveau problème en calculant les dérivées partielles :

$$\frac{\partial L}{\partial w} = w - \sum_{i=1}^{p} \alpha_i y_i x_i = 0 \quad (2.8)$$

$$\frac{\partial L}{\partial b} = \sum_{i=1}^{p} \alpha_i y_i = 0 \quad (2.9)$$

En réinjectant les deux premières dérivées partielles 2.8 et 2.9 dans l'équation 2.7 nous obtenons :

$$L(\alpha) = \frac{1}{2}\sum_{i=1}^{p} \alpha_i y_i \sum_{j=1}^{p} \alpha_j y_j x_j.x_j - \sum_{i=1}^{p} \alpha_i y_i \sum_{j=1}^{p} \alpha_j y_j x_j.x_j - \sum_{i=1}^{p} \alpha_i y_i b + \sum_{i=1}^{p} \alpha_i$$

On en extrait la formulation duale (dépendant des α_i) suivante :

$$L(\alpha) = \sum_{i=1}^{p} \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j x_i.x_j \quad (2.10)$$

On cherche donc à maximiser $L(\alpha)$ sous les contraintes $\alpha_i \geq 0$ et $\sum_i \alpha_i y_i = 0$. A l'optimal

α^* Les conditions de Karush Kuhn Tucker (conditions KTT) sont satisfaites et permettent d'écrire l'égalité suivante :

$$\alpha_i[y_i(w.x_i + b) - 1] = 0, \forall i \in [1, P] \quad (2.11)$$

Cela nous donne $\alpha_i = 0$ ou $y_i(w.x_i + b) - 1 = 0$. Ces deux possibilités impliquent que seuls les α_i associés à des exemples situés sur la marge peuvent être non nuls. Autrement dit ces exemples sur la marge constituent les vecteurs supports, qui seuls contribuent à définir l'hyperplan optimal.

Cette maximisation est un problème de programmation quadratique de dimension égale au nombre d'exemples. L'équation (2.8) nous donne la valeur optimale pour w noté w^* : $w = \sum_{i=1}^{p} \alpha_i^* y_i x_i$, avec α_i^* les coefficients de Lagrange optimaux. En utilisant l'équation (2.1) de l'hyperplan nous obtenons l'hyperplan de marge maximale :

$$h(x) = \sum_{i=1}^{P} \alpha_i^* y_i x.x_i + b \qquad (2.12)$$

2.5 Fonction noyau (kernel)

Le cas linéairement séparable est peu intéressant, parce que les problèmes de classification sont souvent non linéaires. Pour résoudre ce problème nous avons donc besoin d'une fonction connue sous le nom noyau pour projeter les données dan un espace plus grande de dimension appelé espace de redéscription noté H.

La transformation des données en espace de caractéristiques facilite la classification des données non linéaires et la définition d'une mesure de similarité basée sur le produit scalaire.

Nous allons donc appliquer une transformation non linéaire Φ(.) aux vecteurs d'entrée x_i tel que $x_i \in R^d$ et $\Phi(x_i) \in R^e, (e > d)$. Ce changement va conduire à passer d'un produit scalaire dans l'espace d'origine $x_i.x_j$ à un produit scalaire $\Phi\ (x_i).\ \Phi(x_j)$ dans l'espace de redéscription (voir la figure 2.6).

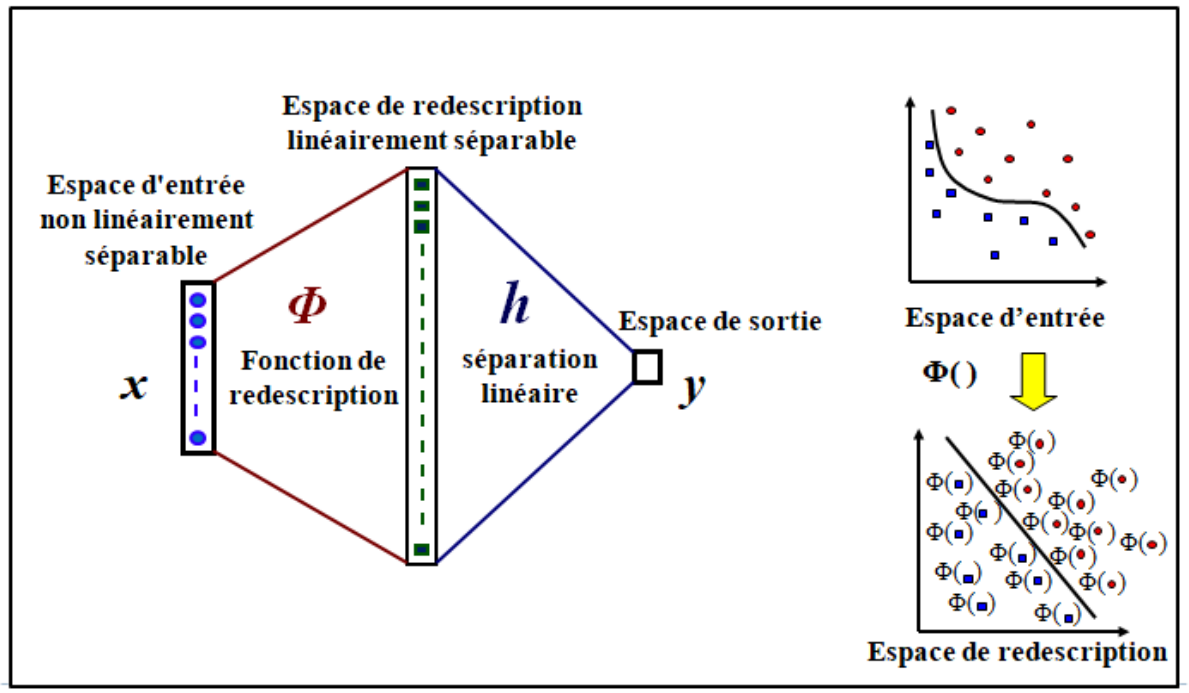

Figure 2.6 : Transformation linéaire des données en une séparation linéaire dans un nouvel espace.

Le principe est d'utiliser une fonction noyau notée K qui évite le calcul explicite du produit scalaire dans l'espace de redéscription.

Nous avons alors l'égalité suivante pour la fonction noyau définie par :

$$K(x_i, x_j) = \Phi(x_i).\Phi(x_j)$$

2.5.1 Exemple

Soit $x = (x_1, x_2)$ dans R^2 et $\Phi(x) = \left(x_1^2, \sqrt{2x_1x_2}, x_2^2\right)$ est explicite Dans ce cas, H est de dimension 3 et le produit scalaire s'écrit :

$$\langle \Phi(x), \Phi(x') \rangle = x_1^2 x_1'^2 + 2x_1x_2x_1'x_2' + x_2^2 x_2'^2$$

$$= (x_1x_1' + x_2x_2')^2$$

$$= \langle x, x' \rangle^2 = K(x, x')$$

Le calcul du produit scalaire dans H ne nécessite pas l'évaluation explicite de Φ. La fonction noyau doit satisfaire la condition de Mercer présentée ci-dessous.

2.5.2 Condition de Mercer

Une fonction K symétrique est un noyau si, pour tous les x_i possibles, la matrice de terme général $K(x_i, x_j)$ est une matrice définie positive c'est-à-dire quelle définit une matrice de produit scalaire [20]. Dans ce cas, on montre qu'il existe un espace H et une fonction Φ tels que :

$$K(x_i, x_j) = \langle \Phi(x_i), \Phi(x_j) \rangle$$

Les problèmes de cette condition sont :

- Cette condition est très difficile à vérifier.
- Elle ne donne pas d'indication pour la construction de noyaux.
- Elle ne permet pas de savoir comment est Φ.

2.5.3 Exemples de noyaux

Les fonctions du noyau les plus courantes sont :

- Linéaire : $K(x_i, x_j) = x_i . x_j$
- Polynomial : $K(x_i, x_j) = (\gamma\, x_i x_j + r)^d$ où d est spécifié par le paramètre degré, r par coefθ .
- Gaussien ou fonction à base radiale (RBF) : $K(x_i, x_j) = exp^{-\gamma \| x_i - x_j \|^2}$ où γ est spécifié par le paramètre gamma, doit être supérieur à 0.
- Sigmoïde : $K(x_i, x_j) = tanh(\gamma(x_i . x_j) + \mathrm{r})$ où r est spécifié par coefθ.
- Laplacien : $K(x_i, x_j) = exp^{-\sqrt{\gamma} \| x_i - x_j \|}$

On remarque en général que le noyau gaussien donne de meilleurs résultats et groupe les données dans des paquets nets. En pratique, on combine des noyaux simples pour en obtenir de plus complexe [16].

2.6 Marges souples

Les svm sont efficaces quand le problème est séparable. Nous avons vu que l'utilisation d'une méthode noyau permettait de traiter les cas non linéaires mais cela n'est pas utilisable pour données non séparables par exemple pour des données bruitées.

Pour gérer ce type de problème on utilise une technique dite de marge souple, qui tolère les mauvais classements comme le montre la figure (2.7). Ce concept a introduit par Cortes et Vapnik en 1995[14]. Certains exemples d'apprentissage peuvent violer la contrainte (2.5) que l'on retrouve dans l'équation (2.6) de la forme primale.

On introduit par conséquent des variables dites « essorts » $\xi = \xi_1, \dots, \xi_p$ permettant d'assouplir la contrainte pour chaque exemple. Un paramètre supplémentaire de régularisation C est ajouté pour contrôler la pénalité associée aux exemples. La nouvelle forme primale décrite en (2.6) devient alors :

$$\begin{cases} min\left(\frac{1}{2}\|w\|^2 + C\sum_{i=1}^{P}\xi_i\right) \\ \forall(x_i, y_i) \in A_p, \qquad y_i(w.x_i + b \geq 1 - \xi_i \end{cases} \tag{2.13}$$

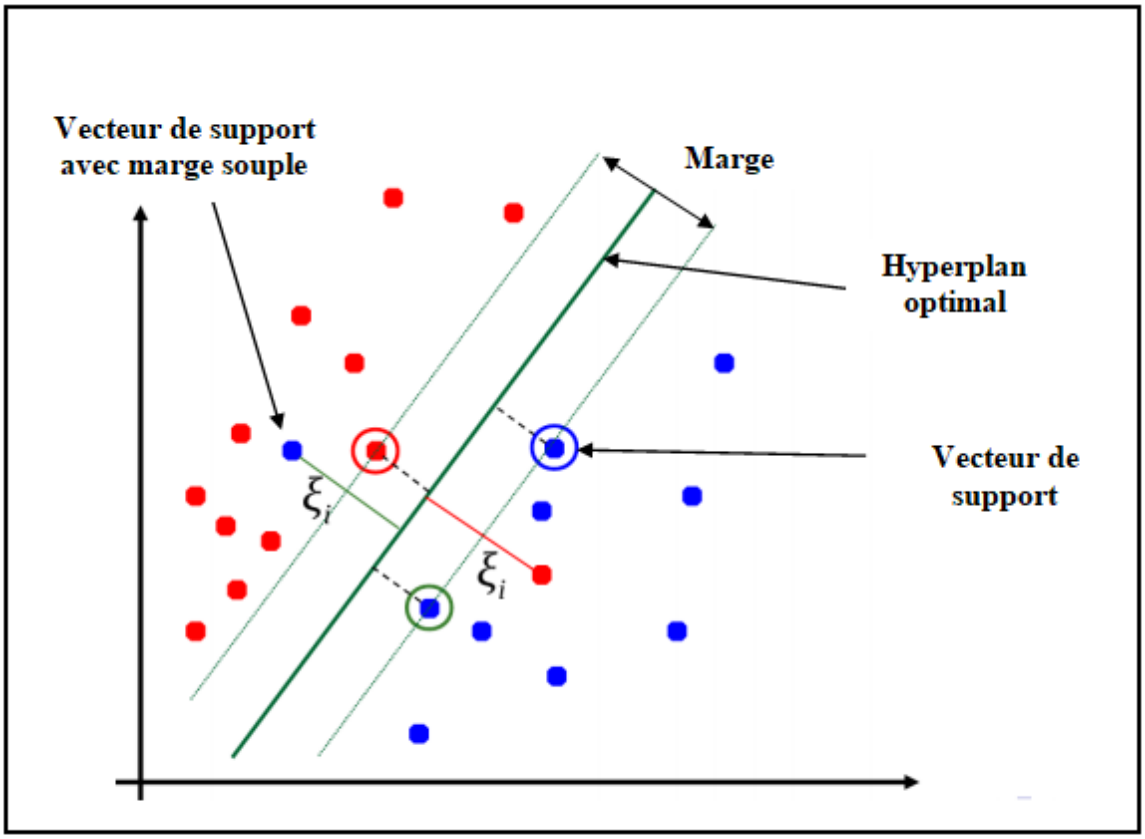

Figure 2.7 : Les données dans le cas non séparable avec marge souple et hyperplan de séparation optimale.

De même que pour la forme primale nous obtenons une nouvelle formulation duale qui est alors similaire à celle décrite précédemment. Si on utilise en plus la fonction noyau K dans la formulation duale (2.10) en appliquant la méthode des multiplicateurs de Lagrange on cherche alors à maximiser la nouvelle fonction $L(\alpha)$.

$$L(w, b, \xi, \alpha) = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{p}\xi_i - \sum_{i=1}^{p}\alpha_i(y_i(w.x_i + b) - 1 + \xi_i)$$

En appliquant la même méthode on obtient $L(\alpha)$ à partir de l'expression du lagrangien précédent.

$$L(\alpha) = \sum_{i=1}^{p}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{p}\alpha_i\alpha_j y_i y_j\, k(x_i x_j) \qquad (2.14)$$

$$\forall(x_i y_i) \in Ap, 0 \le \alpha_i \le c \; et \; \sum_i y_i\alpha_i = 0$$

Le seul changement est la contrainte supplémentaire sur les coefficients α_i, qui se traduit par une borne supérieure C. La solution de l'équation précédente (2.14) est de la forme :

$$h(x) = \sum_{i=1}^{p}\alpha_i^* y_i\, k(x, x_i) + b \qquad (2.15)$$

2.7 SVM multi-classes

A l'origine, les SVM ont été conçus essentiellement pour les problèmes binaires, pendant que les problèmes du monde réel sont dans la plupart des cas multi classe, dans de tels cas les méthodes du SVM multi classes réduisent le problème à une composition de plusieurs hyperplans à deux classes pour tracer les frontières de décision entre les différentes classes [21].

Le principe est de ces méthodes décomposent des exemples en plusieurs sous-ensembles, chacun d'eux représente un problème de classification binaire. Un hyperplan séparateur est déterminé pour chaque problème par le classificateur SVM binaire. Il existe plusieurs méthodes de décomposition dans la littérature, les plus courantes utilisées sont [10] :

2.7.1 Approche Une-contre-Tous (Une-contre-Reste) (OAA : One-Against-All)

C'est la méthode la plus simple et la plus ancienne. Selon la formulation de Vapnik (1998) elle consiste de déterminer pour chaque classe k un hyperplan Hk (wk, bk) la séparant de toutes les autres classes. Cette classe k est considéré comme la classe positive (+1) et les autres classes comme la classe négative (-1) donc pour un problème de K classes, K SVM binaire est obtenu. La figure (2.8) montre un cas de séparation de trois classes [10].

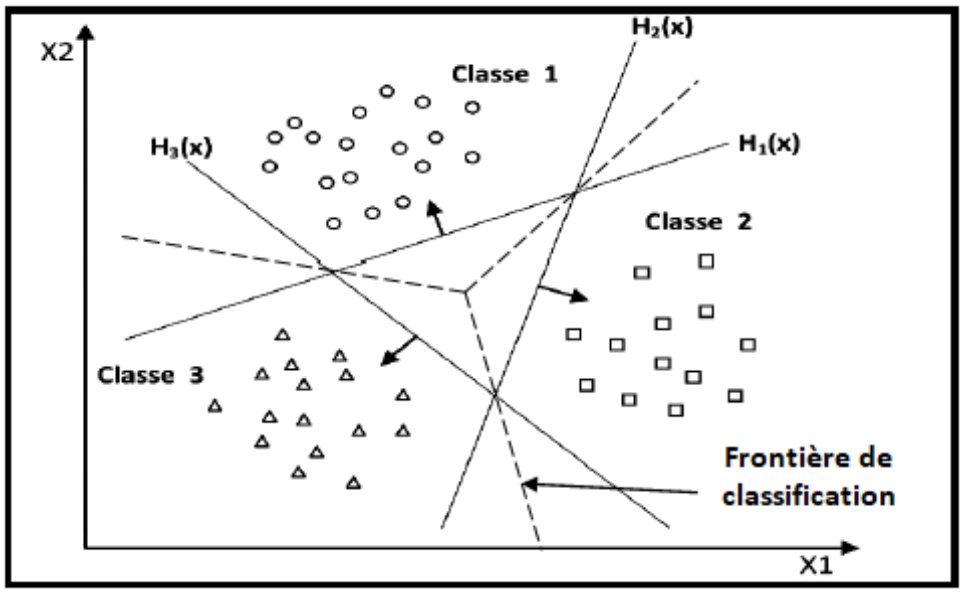

Figure 2.8 : L'approche une-contre-tous.

2.7.2 Approche Une-contre-Une (OAO : One-Against-One)

Cette approche consiste à utiliser un classifieur pour chacune des deux classes. Cette méthode discrimine chaque classe de chaque autre classe, ainsi K (K − 1) / 2 fonctions de décision sont apprises.

Pour chaque paire de classes (k, s), cette méthode définit une fonction de décision binaire. L'attribution d'un nouvel exemple se fait par liste de vote. Un exemple est testé en calculant sa fonction de décision pour chaque hyperplan. Pour chaque test, il y a un vote pour la classe à laquelle appartient l'exemple (classe gagnante) [10]. D'après la figure (2.9) on remarque que la région des points non classifiable existe toujours.

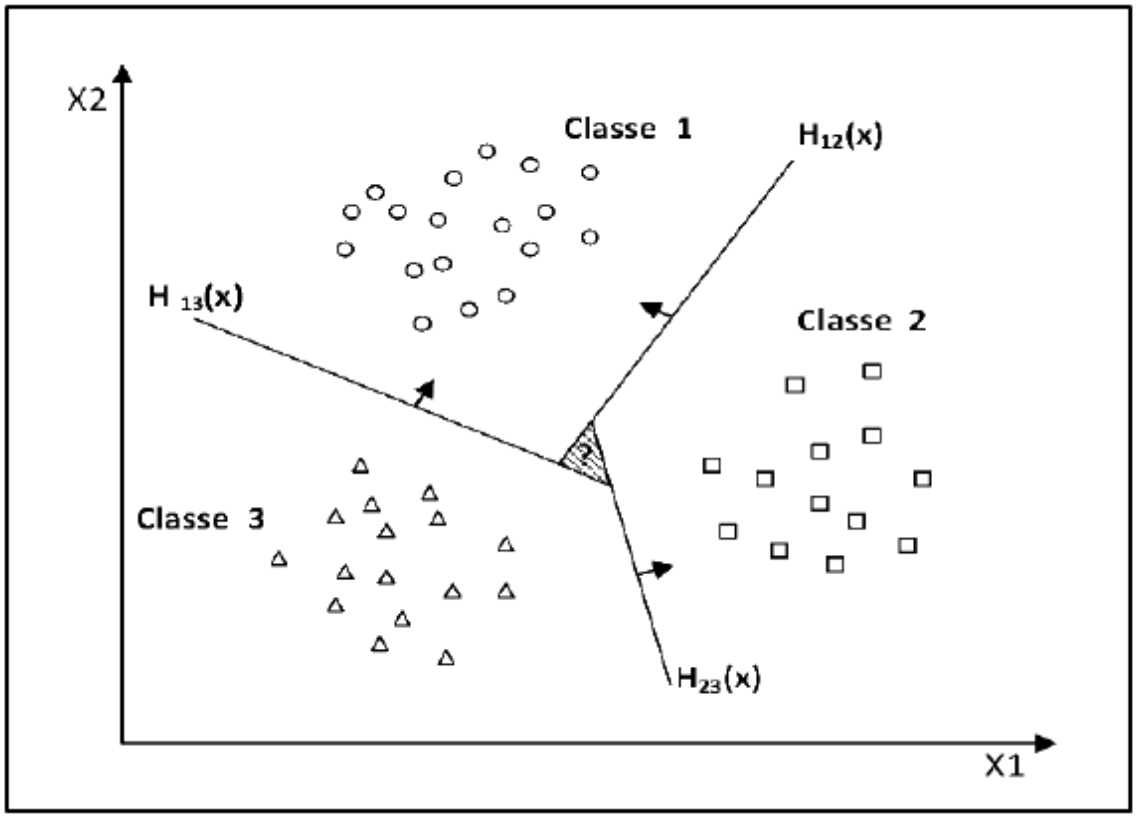

Figure 2.9 : L'approche une-contre-une

2.8 Les domaines d'applications

SVM est une méthode de classification qui montre de bonnes performances dans la résolution de divers problèmes. Cette méthode a montré son efficacité dans de nombreux domaines d'applications tels que :

- Traitement d'images : (reconnaissance de caractères manuscrits, reconnaissance de scènes naturelles, reconnaissance de visages)
- Catégorisation de textes : (classification de courriels, classification de pages web)
- Diagnostic médical : (évaluation du risque de cancer, détection d'arythmie cardiaque)
- Classification de données biologiques/physiques.
- Classification de documents numériques.
- Classification d'expressions faciales.
- Classification de textures.
- E-Learning.
- Reconnaissance de la parole (appelé aussi reconnaissance vocale).
- CBIR : Content Based Image Retrieval.

2.9 Les avantages et les inconvénients des SVM

2.9.1 Les avantages

- Un grand taux de classification et de généralisation par rapport aux méthodes classiques.
- Elle nécessite moins d'effort pour designer l'architecture adéquate (petit nombre de paramètre à régler ou à estimer).
- La résolution du problème est convertie en résolution d'un problème quadratique convexe dont la solution est unique et donnée par des méthodes mathématiques classiques de programmation quadratique.
- Traitement des problèmes non linéaires avec le choix des noyaux.
- Avec la SVM linéaire (sans noyau), la frontière de décision construite correspond à un hyperplan dont les coefficients peuvent être interprétés comme l'importance que le classeur donne aux caractéristiques d'après leur importance pour la classification [22].
- Les méthodes de SVM qui sont particulièrement bien adaptées pour traiter des données de grande dimension (telles que des représentations vectorielles de texte) [24].
- L'utilisation de la SVM linéaire est particulièrement bonne pour la classification des documents de texte pour les raisons suivantes, mentionnées par [23] :

- La plupart des problèmes de catégorisation des textes sont linéairement séparables.
- Il y a peu de caractéristiques insignifiantes.

2.9.2 Les inconvénients

- Les SVM peuvent avoir des problèmes dans le cas où beaucoup de termes sont non significatifs pour la discrimination de classe [24].
- Elles ne servent pas à calculer des probabilités d'appartenance à la classe [25].
- L'utilisation des fonctions noyaux ne permet pas de sélectionner des caractéristiques importantes pour la classification [25].
- La sélection d'une mauvaise fonction noyau (ou de ces paramètres) peut conduire à produire un effet de sur-apprentissage (overfitting) et c'est le principal désavantage des SVM [26].
- Problème lorsque les classes sont bruitées (multiplication des points supports).
- Elles utilisent des fonctions mathématiques complexes pour la classification.

2.10 Conclusion

Dans ce chapitre, Nous avons introduit une des méthodes d'apprentissage supervisé populaire pour la classification qui est " les machines à vecteurs de supports" , nous avons expliqués les différents aspects de cette méthode, commençant par la présentation du principe de fonctionnement générale des SVM et leur fondements mathématiques ; et nous avons exposé les deux cas des données linéairement séparables et non linéairement séparables.

Dans la forme la plus simple, SVMs utilisent un hyperplan linéaire qui permet de séparer au mieux des ensembles de données, dans d'autres cas, où les données ne sont pas linéairement séparables, nécessite l'utilisation d'une marge souple et une fonction noyau pour trouver un hyperplan séparateur optimal.

Ensuite, nous avons vu qu'il existe les deux principales approches des machines à vecteurs de supports multi-classes qui sont : l'approche une-contre-une et l'approche une-contre -reste. Nous avons aussi vu que le champ d'application des SVM est large et représente une méthode de classification intéressante.

Chapitre 03 :

Le Réseau de neurones convolutif

3.1 Introduction

Dans les algorithmes traditionnels de l'apprentissage automatique, la notion de caractéristiques (features) en vision est utilisée pour faire la classification des images. Ces méthodes consistent à extraire les caractéristiques de chaque image du jeu de données manuellement par un expert, ensuite à entraîner un classificateur sur ses caractéristiques. Ces techniques d'apprentissage supervisé peuvent donner de très bons résultats, et leur performance dépend fortement de la qualité des caractéristiques préalablement trouvées.

Mais en 2012, une révolution a eu lieu : lors de la compétition annuelle de vision par ordinateur ILSVRC (the ImageNet Large Scale Visual Recognition Competition), un nouvel algorithme de L'apprentissage profond obtenez des résultats étonnants ! Il s'agit réseau de neurones convolutif CNN [27] [28].

Aujourd'hui, les réseaux de neurones convolutifs, sont toujours les modèles les plus performants pour la classification d'images.

3.2 Définition de L'apprentissage profond

L'apprentissage profond ou Deep Learning (DL) est un type d'intelligence artificielle dérivé de l'apprentissage machine où la machine est capable d'apprendre par elle-même, sans supervision, sans intervention humaine, en simulant la mécanique du cerveau humain pour exécuter avec précision des applications complexes basées sur des réseaux neuronaux artificiels [29].

Le réseau peut avoir de nombreuses couches cachées et s'il y a deux couches cachées ou plus, nous appelons le réseau un réseau neuronal profond. Dans cette partie, nous nous concentrons sur un réseau très important dans réseaux neuronaux artificiels est le réseau de neurones convolutif.

3.3 Définition de réseau de neurones convolutif

CNN Réseau de neurones convolutif ou (Convolutional Neural Network en anglais) est un type de modèle d'apprentissage en profond spécialisé dans le traitement de données telles que la classification des images, s'inspirent biologiquement du cortex visuel, cette idée a été développée par une expérience guidée par Hubel & Wiesel (1962) [30]. Les CNN sont spécialement conçus pour traiter des images en entrée. Leur architecture est donc plus spécifique : elle est composée de deux blocs principaux :

- Le premier bloc fait la particularité de ce type de réseaux de neurones, puisqu'il fonctionne comme un extracteur de caractéristiques. Pour cela, en appliquant des

opérations de filtrage par convolution. La première couche filtre l'image avec plusieurs noyaux (filtre ou kernels en anglais) de convolution, et renvoie des cartes de caractéristiques (features maps), qui sont ensuite normalisées (avec une fonction d'activation par exemple **ReLU**) et/ou redimensionnées. Ce procédé peut être répété plusieurs fois : on filtre les Carte de caractéristiques obtenues avec de nouveaux noyaux (kernels), ce qui nous donne de nouvelles carte de caractéristiques à normaliser et redimensionner, et vous pouvez filtrer encore, et ainsi de suite. Finalement, les valeurs des dernières cartes de caractéristiques sont concaténées dans un vecteur. Ce vecteur définit le résultat du premier bloc, pour l'entrée du second.

- Le second bloc : Les valeurs du vecteur en entrée sont transformées (avec la fonction d'activation) pour renvoyer un nouveau vecteur en sortie. Ce dernier vecteur contient autant d'éléments qu'il y a de classes.

3.4 Le principe de réseaux de neurones convolutifs

Les données d'entrée dans un CNN sont traitées dans une topologie de type grille [31]. Par exemple, une image peut être considérée comme une grille de taille D * D * C de valeurs d'entrée, où D * D est le nombre de pixels dans l'image et C est le nombre de canaux par pixel (1 pour l'échelle de gris ou 3 pour RVB). Cette grille d'entrée est transmise à un certain nombre de couches : couches convolutionnelles, couches de regroupement (Pooling) et couches entièrement connectées. Le résultat peut être une classe unique ou une probabilité de classes décrivant la catégorie de l'image.

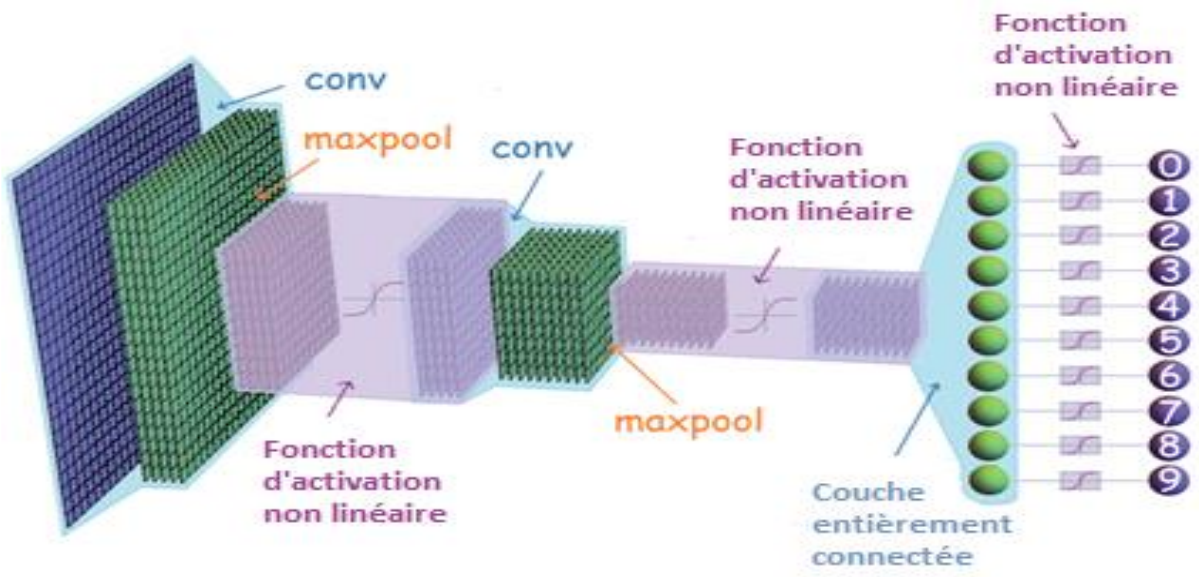

Figure 3.1 : Structure générale d'un réseau de neurones convolutifs

3.5 Les couches de réseaux de neurones convolutionnels

Il existe quatre types de couches pour un réseau de neurones convolutif : la couche de convolution, la couche de pooling, la couche de correction ReLU et la couche fully-connected entièrement connectées :

3.5.1 Couches de convolution

La couche de convolution est un composant fondamental de l'architecture CNN qui effectue l'extraction de caractéristiques [32] [33], De façon simpliste, la convolution consiste à appliquer un filtre mathématique à une image, D'un point de vue plus technique, il s'agit de faire glisser une matrice par-dessus une image, et pour chaque pixel, utiliser la somme de la multiplication de ce pixel par la valeur de la matrice. Cette procédure est répétée en appliquant plusieurs noyaux pour former un nombre arbitraire de cartes de caractéristiques. Cette technique nous permet de trouver des parties de l'image qui pourraient nous être intéressantes.

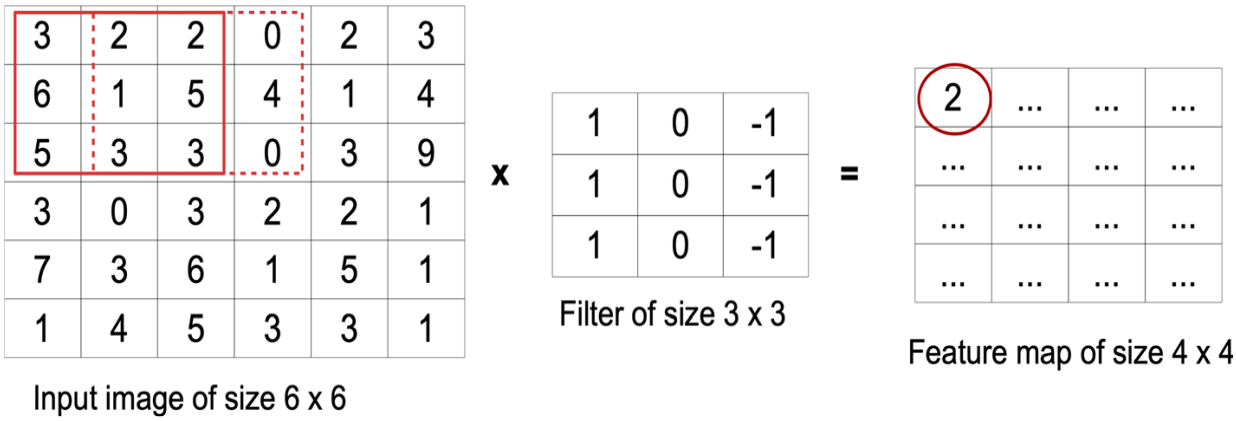

Figure 3.2 Opération d'une convolution sur image de 6*6 pixel.

Noté qu'une convolution 3x3 de profondeur 1 effectuée sur une carte de caractéristiques d'entrée 6x6, également de profondeur 1. Comme il y a seize emplacements 4x4 possibles pour extraire les tuiles de la carte de caractéristiques 6x6, cette convolution génère une carte de caractéristiques de sortie 4x4.

Le filtre doit se déplacer d'une case à chaque itération jusqu'à ce que la première ligne soit finie, Lorsque nous terminons la première ligne. Lorsque nous avons fini la première ligne le filtre passé d'une case et la même procédure se répètent pour chaque ligne et colonne.

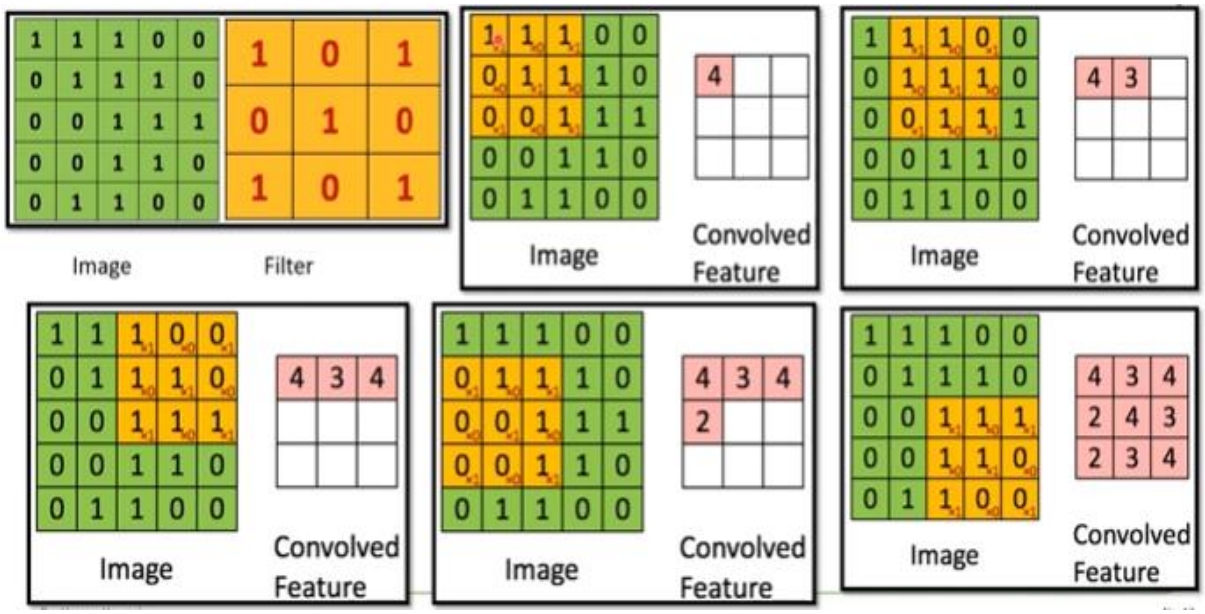

Figure 3.3 Un exemple complet de l'opération d'une convolution

3.5.2 Couches de Pooling

La couche de pooling fournit une opération typique de sous-échantillonnage qui réduit la dimensionnalité dans le plan des cartes de caractéristiques. Une couche de pooling permet de réduire la taille de la matrice d'entrée, d'augmenter la vitesse tout en gardant les informations les plus importantes. Généralement, une couche de pooling est placée entre deux couches de convolution. Pour ce faire, on découpe l'image en cellules régulière, puis on garde au sein de chaque cellule la valeur maximale. Le raisonnement intuitif derrière l'opération de sous-échantillonnage est que la détection des caractéristiques est plus importante que l'emplacement exact des caractéristiques.

Il existe deux types de pooling :

A) Mise en commun maximale (Max-pooling)

La forme la plus populaire d'opération de pooling est la mise en commun maximale, qui extrait les correctifs des cartes de caractéristiques d'entrée, génère la valeur maximale dans chaque correctif et supprime tous les autres. Un max pooling avec un filtre de taille 2 × 2 avec une foulée de 2 est couramment utilisé en pratique. Cela sous-échantillonne la dimension dans le plan des cartes de caractéristiques par un facteur de 2. Contrairement à la hauteur et à la largeur, la dimension de profondeur des cartes de caractéristiques reste inchangée.

B) Mise en commun moyenne (Average pooling)

Une autre opération de pooling à noter est une average pooling [34]. Average pooling effectue un type extrême de sous-échantillonnage, où une carte de caractéristique avec une taille de hauteur × largeur est sous-échantillonnée dans un tableau 1 × 1 en prenant simplement la moyenne de tous les éléments de chaque carte de caractéristique, alors que la profondeur des cartes de caractéristique est retenue. Cette opération est généralement appliquée une seule fois avant les couches entièrement connectées. Les avantages de l'application de la mise en commun moyenne (average pooling) sont :

- Réduit le nombre de paramètres apprenables
- Permet au CNN d'accepter des entrées de taille variable.

La figure 3.4 montre un exemple d'opération de max pooling et Average pooling avec une taille de filtre 2x2 pixels à partir d'une entrée de pixels 4x4.

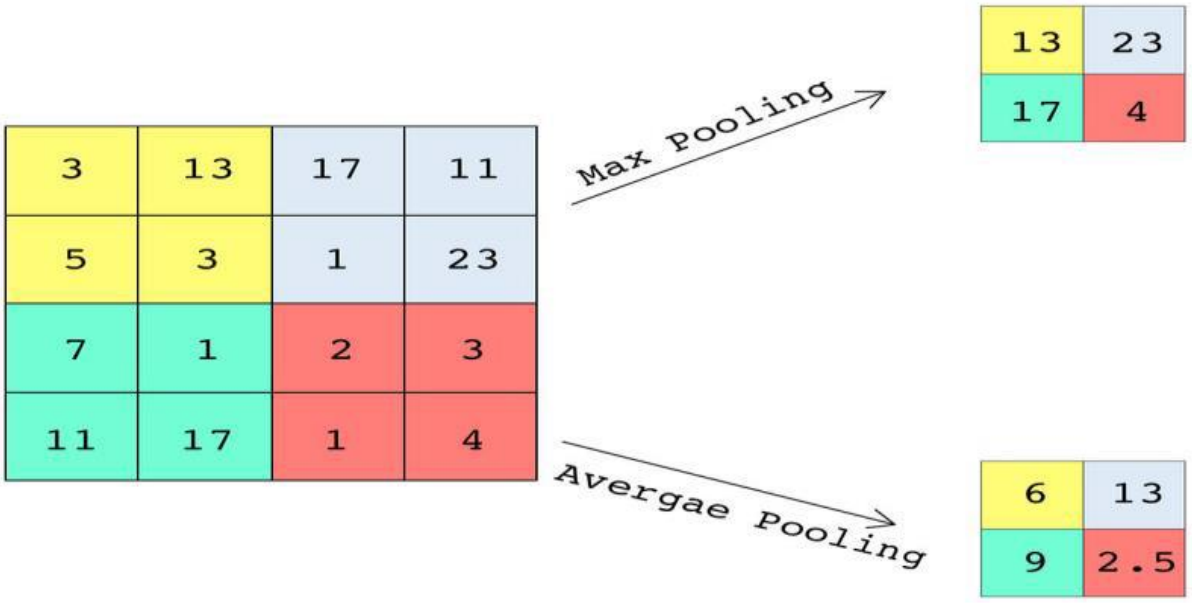

Figure 3.4 : Exemple de fonctionnement de Max pooling et Average pooling [35].

3.5.3 La couche de correction ReLU

Différents types de fonctions d'activation sont utilisés dans l'apprentissage en profond. La fonction d'activation sigmoïde est chargée de négliger les informations d'images et pour cela, la majeure partie du réseau utilise la couche ReLU. La fonction ReLU (abréviation de : Unités

Rectifié linéaires) est une fonction d'activation non linéaire qui est simple à utiliser et s'exécute plus rapidement [36].

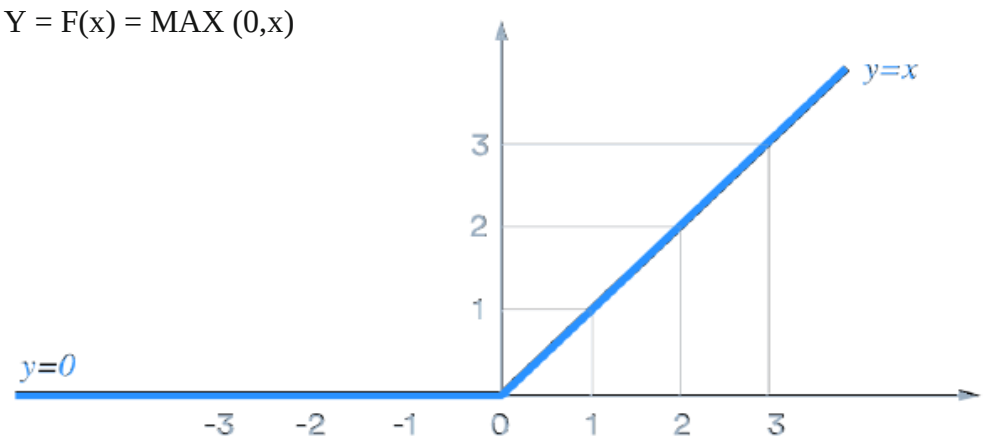

Figure 3.5 : Représentation graphique de la fonction ReLU avec son équation [37]

La couche de correction ReLU remplace donc toutes les valeurs négatives reçues en entrées par des zéros (0). Elle joue le rôle de fonction d'activation.

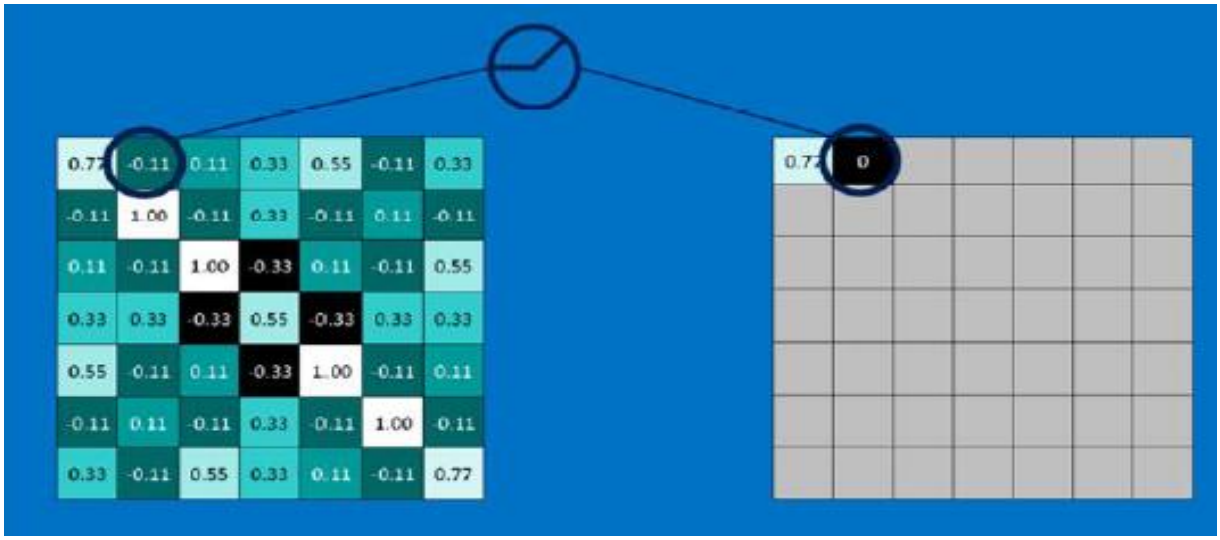

Figure 3.6 : application de fonction Relu

3.5.4 Couches entièrement connectées

La couche entièrement connectée (fully-connected) constitue toujours la dernière couche d'un réseau de neurones. Le terme « entièrement connecté » implique que chaque neurone de la couche précédente est connecté à chaque neurone de la couche suivante. Ce type de couche reçoit un vecteur en entrée et produit un nouveau vecteur en sortie.

La dernière couche entièrement connectée permet de classifier l'image à partir des caractéristiques extraites par la succession de bloc de traitement. Chaque neurone attribue à l'image une valeur de probabilité d'appartenance à la classe i parmi les C classes possibles. Chaque probabilité est calculée à l'aide de la fonction « softmax » dans le cas où les classes sont exclusivement mutuelles.

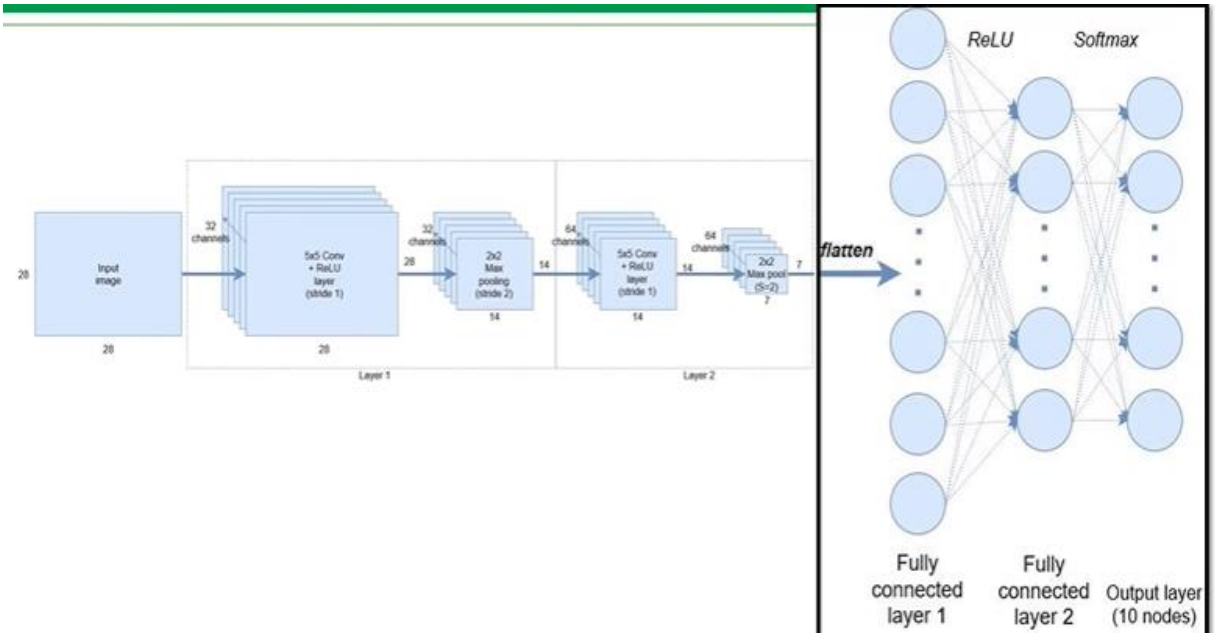

Figure 3.7 Un exemple de la couche entièrement connectées

3.6 L'entrainement d'un réseau de neurone convolutionnelle

L'entraînement d'un CNN consiste à déterminer et à calculer empiriquement la valeur de chacun de ses poids. Le principe est le suivant : le CNN traite une image (de la base de données d'entraînement) et en sortie il fait une prédiction, c'est-à-dire qu'il dit à quelle classe il pense que cette image appartient. Sachant qu'on connaît préalablement la classe de chacune des images d'entraînement, on peut vérifier si ce résultat est correct. En fonction de la véracité de ce résultat, on met à jour tous les poids du CNN selon un algorithme qui s'appelle la rétropropagation du gradient de l'erreur.

Lors de la phase d'entraînement du modèle, le processus expliqué ci-dessus est répété plusieurs fois et avec la totalité des images de la base de données d'entraînement. Le but étant que le modèle classifie au mieux ces données.

Lorsque le modèle a fini de mettre à jour ses poids, on évalue le modèle en lui présentant la base de données de validation.

Il classe toutes ces images (qui sont des images que le modèle n'a jamais vues) et on calcule son taux de bonne classification, c'est ce qu'on appelle la précision du modèle.

3.7 Architecture d'un CNN

De nos jours, les CNN sont considérés comme les algorithmes les plus largement utilisés parmi les inspirés des techniques d'Intelligence Artificielle (IA). L'histoire de CNN commence par le neurobiologique expériences menées par Hubel et Wiesel (1959, 1962) [38]. Leur travail a fourni une plateforme pour de nombreux modèles cognitifs, et CNN a remplacé presque tous ceux-ci. Au fil des décennies, différents efforts ont été menés pour améliorer les performances des CNN.

Il existe de nombreuses architectures CNN réputées. Les architectures CNN les plus populaires sont données dans **Figure 3.8**

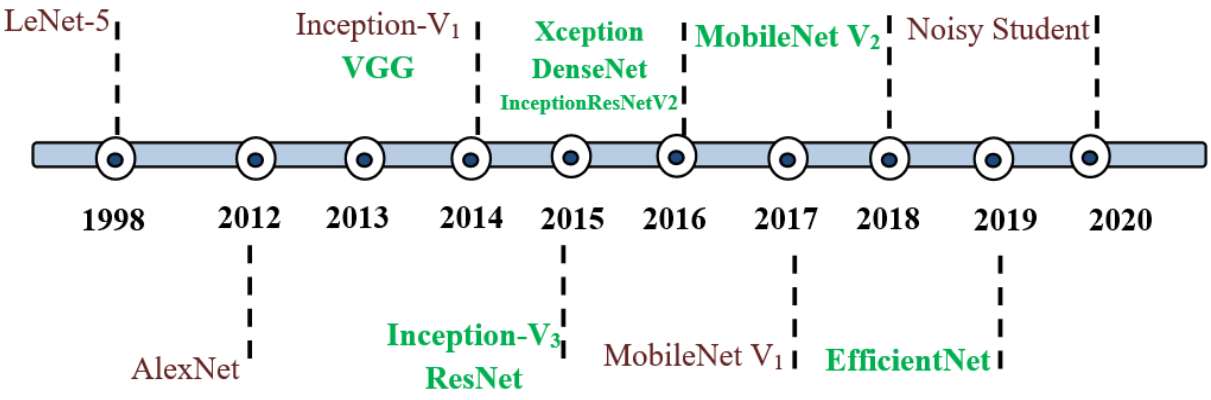

Figure 3.8 Histoire évolutive des CNNs montrant les innovations architecturales [39].

Ci-dessous, nous expliquons les architecturales CNN les plus importants :

3.7.1 LeNet-5

LeNet était le réseau neuronal convolutif le plus archétype développé par Yann Le Cun en 1990 [40] et amélioré plus tard en 1998. L'architecture LeNet la plus connue et la plus efficace pour lire les codes postaux et les chiffres, etc. Cette architecture contient 4 couches convolution et mise en commun (Pooling) alternées, suivies de 3 couches entièrement connectées.

3.7.2 AlexNet

La première architecture CNN célèbre est AlexNet, qui popularise le réseau de neurones convolutifs en vision par ordinateur, développé par Alex Krizhevsky, Ilya Sutskever et Geoff Hinton. Cette architecture contient 5 couches convolutives avec des unités linéaires rectifiées

(ReLU) comme fonctions d'activation, 3 couches Max Pooling et 3 couches entièrement connectées.

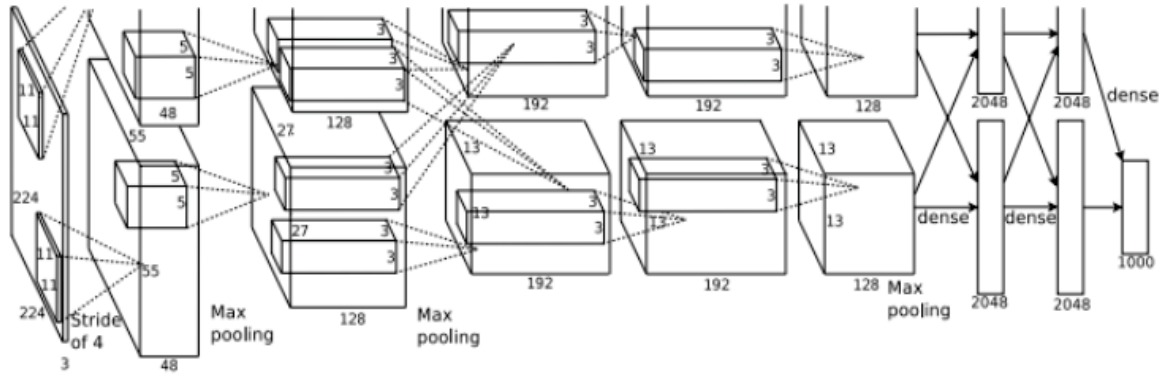

Figure 3.9 Architecture CNN proposée par Alex Krizhevsky et al (2012).

3.7.3 GoogleNet

GoogleNet a été le gagnant du concours 2014-ILSVRC, connu également comme Inception-V1, Il a été développé par une équipe de Google (Christian Szegedy et al) [41].

C'est un type de CNN basé des modules inception, ces blocs encapsulent des filtres de différentes tailles (1x1, 3x3 et 5x5) pour capturer des informations spatiales à différentes échelles, suivi du filtre concat qui permet de concaténer les résultats des filtres. En outre, la densité de la connexion a été réduite en utilisant mise en commun moyenne (Average pooling) à la dernière couche, plutôt qu'une couche entièrement connectée. Ces paramètres ont entraîné une baisse significative du nombre de paramètres, qui est passé de 60 millions à 4 millions.

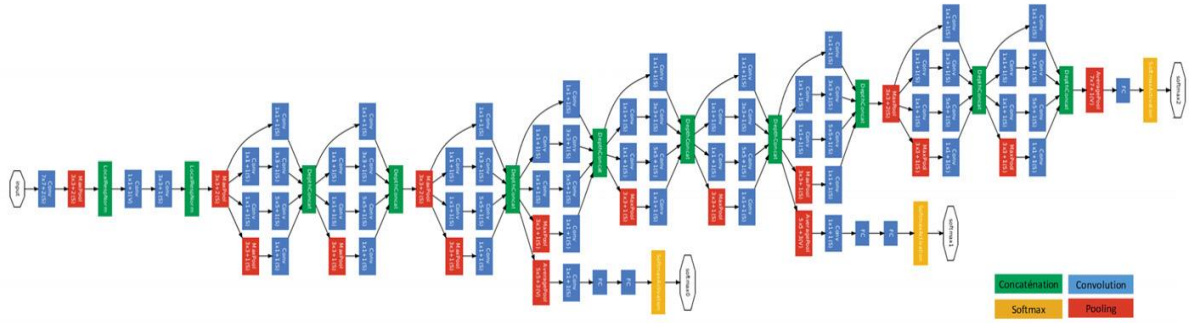

Figure 3.10 GoogleNet proposée par Szegedy *et al.* (2015)

3.7.4 MobileNet-v1

MobileNet est une architecture légère développée par Howard et al. [42] de Google, le modèle MobileNet est conçu pour être utilisé dans des applications mobiles, cette architecture utilise des convolutions séparables en profondeur. Il réduit considérablement le nombre de paramètres par rapport au réseau avec des convolutions régulières avec la même profondeur

dans les filets. Il en résulte des réseaux de neurones profonds légers. Howard et al introduisent deux hyperparamètres globaux simples qui font un compromis efficace entre latence et précision. Ces hyperparamètres permettent au constructeur de modèles de choisir le modèle de la bonne taille pour leur application sur les contraintes du problème.

3.7.5 Noisy Student

Noisy Student a été introduit par l'équipe de recherche de Google (Xie et al) en 2020. [43] Noisy Student Training est une approche d'apprentissage semi-supervisée. Il étend l'idée d'auto-formation et de distillation avec l'utilisation de modèles d'élèves égaux ou plus grands et de bruit ajouté à l'élève pendant l'apprentissage. Noisy Student Training cherche à améliorer l'auto-formation et la distillation de deux manières. Premièrement, cela rend l'élève plus grand ou au moins égal à l'enseignant afin que l'élève puisse mieux apprendre à partir d'un plus grand ensemble de données. Deuxièmement, cela ajoute du bruit à l'élève, de sorte que l'élève bruyant est obligé d'apprendre plus durement à partir des pseudo étiquettes. Pour brouter l'élève, il utilise le bruit d'entrée tel que l'augmentation des données, et le bruit du modèle tel que la profondeur stochastique pendant l'entraînement.

En général. Un réseau de neurones à convolution peut avoir plusieurs étapes de convolution, ReLu et la Mise en commun (Pooling). Une règle à respecter est que la fonction de ReLu doit obligatoirement être appliquée après une étape de convolution pour obtenir une réponse non linéaire, mais la Mise en commun (Pooling) n'est pas obligatoire.

Après avoir passé par toutes les étapes de convolution, ReLu et la Mise en commun (Pooling), on peut passer à la classification des images. La dernière phase est l'envoi de tous les pixels dans un réseau de neurones multi-couches. Étant donné que nous avons pu récupérer les parties les plus importantes d'une image que nous avons condensée, l'étape de classification sera beaucoup plus performante que l'utilisation d'un réseau neuronal artificiel sans convolution.

3.8 Applications de CNN

3.8.1 Reconnaissance de caractères

Trouver des mots-clés dans des morceaux de textes peut être difficile pour les yeux humains, surtout pour ceux qui ont une faible capacité visuelle. L'apparition des bureaux sans papier a rendu nécessaire l'analyse automatisée des documents. Ces deux tâches, ainsi que d'autres qui exigent que les machines reconnaissent les caractères, peuvent être facilement réalisées grâce à le CNN, qui constitue un élément de base. La technique de pointe utilisée par les nombreux chercheurs qui tentent de construire des modules de CNN haute précision avec

un taux d'erreur minimal pour différentes langues et différents types de polices (numériques ou manuscrites, séparées ou attachées) est le CNN pour sa supériorité en matière de reconnaissance d'images.

3.8.2 Reconnaissance vocale

Tandis que le CNN peut être bon dans la reconnaissance d'image comme mentionné précédemment, il peut également donner une performance remarquable dans la reconnaissance de la parole pour sa grande capacité de prédire les phrases et leur signification à partir du contexte de la conversation et aussi la flexibilité que le CNN fournit pour inclure toutes sortes de langues ainsi que les différents accents.

3.8.3 Publicité

Les CNN ont déjà apporté un monde de différence à la publicité avec l'introduction de l'achat programmatique et de la publicité personnalisée basée sur les données.

3.9 Conclusion

Dans ce chapitre nous avons donné une petite définition sur l'apprentissage profond, et le passage aux réseaux de neurones convolutifs. Nous avons aussi donné le principe de ce réseau. Ensuite nous avons présenté en détail les couches de réseaux de neurones convolutifs. Puis nous avons expliqué les architectures CNN les plus populaires. Finalement nous avons donnée quelques exemples d’applications informatiques utilisant le CNN.

Chapitre 04 : Implémentation et Résultats

4.1 Introduction

Les systèmes de reconnaissance des caractères arabes manuscrits font face à plusieurs défis, y compris la variation illimitée de l'écriture humaine et des grandes bases des données publiques.

L'objectif de ce dernier chapitre est de présenter les étapes de l'implémentation des modèles proposés dans le cadre d'un système de reconnaissance des caractères arabes manuscrits, nous avons implémenté les deux techniques de classification CNN et SVM, puis nous avons proposé un modèle hybride entre les deux, les différentes étapes de réalisation ont été présentées, ainsi que la discussion à propos des résultats obtenus.

D'abord, Nous commençons par présenter l'environnement de développement et les ressources utilisés, décrire la création de la base de données qui a servit à l'entrainement et l'évaluation de notre modèle puis les étapes de la réalisation de ce dernier. Nous poursuivons ce chapitre par la présentation des différents résultats expérimentaux obtenus, illustré par quelques captures d'écrans de l'interface de notre application.

4.2 L'environnement de développement utilisé

4.2.1 Environnement matériel

Afin de réaliser et développer notre projet a été effectué dans un ordinateur avec les caractéristiques techniques suivantes :

- Processeur : Intel® Core™ i3-4030U CPU @ 1.90GHz
- Mémoire installé (RAM) : 4,00 Go
- Disque dur : 500 Go

4.2.2 Environnement logiciels et librairies

A) Windows 10 Famille 64bits

B) Python

Python est un excellent langage de programmation de haut niveau interprété (il n'y a pas d'étape de compilation) et orienté objet, il est très sollicité par une large communauté de développement et de programmeurs. Python est un langage simple, facile à apprendre et permet une bonne réduction du cout de la maintenance des codes.

Les bioéthique (package) python encouragent la modularité et la réutilisabilité des codes. Python et ses bibliothèques sont disponible (en source ou binaires) sans charges pour la majorité des plateformes et peuvent être redistribués gratuitement [44].

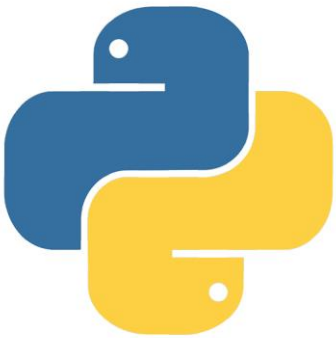

Figure 4.1 : Logo de langage python

C) Jupyter Notebook

L'application Jupyter Notebook est une application web open-source client-serveur qui permet de créer, afficher, modifier et d'exécuter des documents notebooks via un navigateur Web. En plus de ça, dispose d'un tableau de bord (tableau de bord de notebooks), un panneau de configuration affichant les fichiers locaux et permettant d'ouvrir des documents notebooks ou d'arrêter leurs noyaux. Le nom "Jupyter" est un acronyme pour Julia, Python et R. Ces langages de programmation étaient les premiers ciblés par jupyter, qui supporte aujourd'hui une grande variété de langages.

Parmi ses usages : nettoyage et transformation des données, modélisation statistique simulation numérique, apprentissage automatique…etc.

Figure 4.2 : Logo d'application Jupyter Notebook

D) TensorFlow

TensorFlow est une bibliothèque open source développé par l'équipe de Google Brain est basé sur l'apprentissage automatique et l'apprentissage en profondeur, pour le calcul numérique de haute performance, il est aussi fonctionné sur multiplateforme tel que les processeurs et les processeurs graphiques et les processeurs de tenseurs (CPUs, GPUs, TPUs)

Figure 4.3 : Logo de bibliothèque TensorFlow

E) Keras

Keras est une API de réseaux neuronaux de haut niveau, écrite en python et capable de s'exécuter sur TensorFlow, CNTK ou Theano. Il a été développé dans le but de permettre une expérimentation rapide. Être en mesure de passer de l'idée au résultat le plus rapidement possible, est la clé pour faire de la recherche :

- Permet un prototypage facile et rapide, grâce à la convivialité, à la modularité et à l'extensibilité.
- Prend en charge les réseaux (CNN, RNN) ainsi que les combinaisons des deux.
- Fonctionne de manière transparente sur le processeur et le processeur graphique. [45]

Figure 4.4 : Logo de bibliothèque Keras

4.3 Bases de données

Dans ce travail, deux bases de données populaires ont été utilisées : MNIST et AHCD

4.3.1 La base de données MNIST

Le MNIST est un ensemble de données de chiffres manuscrites, L'acronyme de (Modified ou Mixed National Institute of Standards and Technology). C'est un ensemble de données très utilisé en apprentissage automatique.

La base MNIST est devenue un test standard. Se compose d'images manuscrites des 10 chiffres (0 à 9). Elle regroupe 60,000 images d'apprentissage et 10,000 images de test, issues d'une base de données antérieure, appelée simplement NIST. Ce sont des images en noir et blanc, normalisées centrées de 28 pixels de côté. [47]

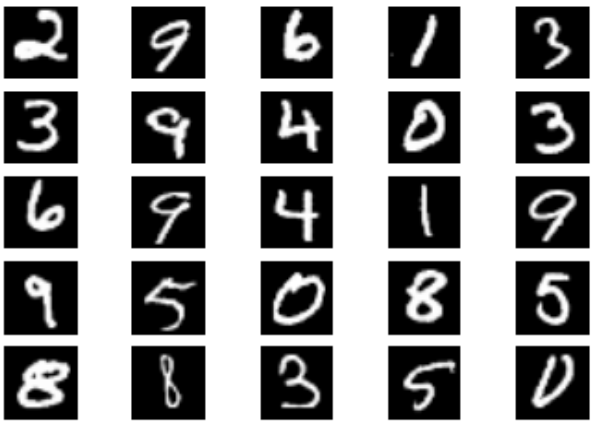

Figure 4.5: Des échantillons de la base de données MNIST.

4.3.2 La base de données AHCD :

L'ensemble de données est composé de 16 800 caractères écrits par 60 participants, la tranche d'âge est se situe entre 19 et 40 ans et 90 % des participants sont droitiers. Chaque participant a écrit chaque caractère (de 'alef' à 'yeh') dix fois sur deux formulaires. Les formulaires ont été scannés à la résolution de 300 dpi. Chaque bloc est segmenté automatiquement à l'aide de Matlab 2016a pour déterminer les coordonnées de chaque bloc.

La base de données est partitionnée en deux ensembles : un ensemble d'apprentissage (13440 caractères pour 480 images par classe) et un ensemble de test (3 360 caractères pour 120 images par classe).

Les rédacteurs de l'ensemble d'entraînement et de l'ensemble de test sont exclusifs. L'ordre d'inclusion des rédacteurs dans l'ensemble de test est aléatoire pour s'assurer que les auteurs de l'ensemble de test ne proviennent pas d'une seule institution (pour assurer la variabilité de l'ensemble de test) [46].

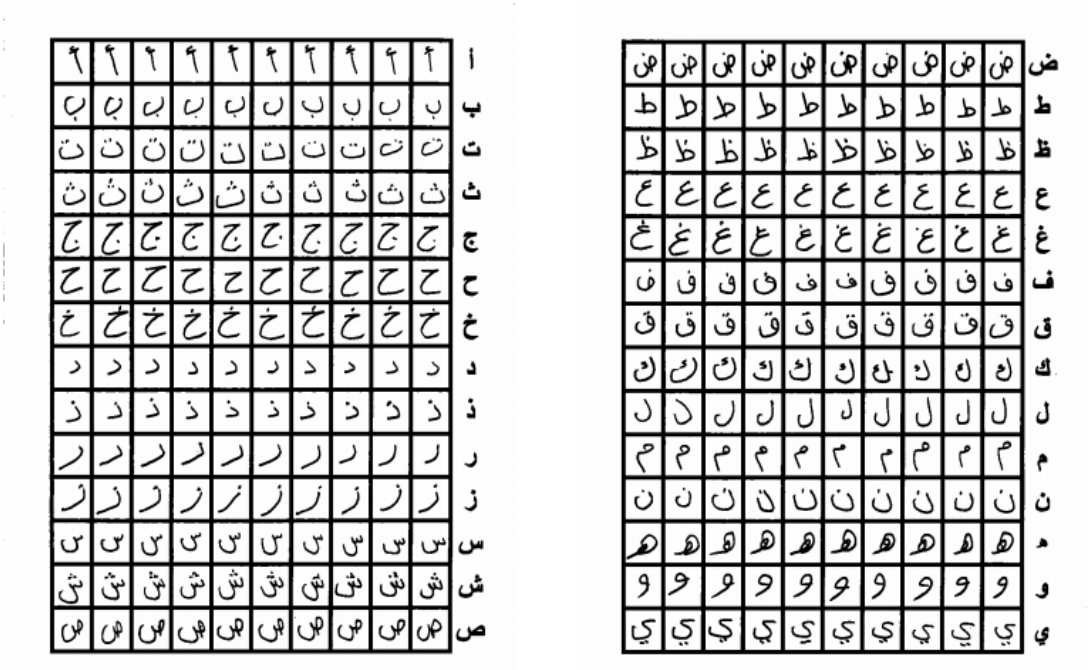

Figure 4.6 : L'ensemble de données AHCD

4.4 Architecture de notre modèle

4.4.1 Le premier modèle (MNIST)

Dans ce modèle nous avons utilisé 4 couches de convolution dont chaque couche utilise un filtre de taille 3*3, et nous proposons d'utiliser 32 filtre dans la première et deuxième couche ,et 64 filtre dans la troisième et quatrième couche, toutes les couches convolutives conv1-4 avec une rectification linéaire Relu ,cette fonction force les neurones à retourner des valeurs positive, une couche de Maxpooling est appliquée après chaque couche de convolution, afin de réduire la taille de l'image et la quantité des paramètres, et une couche de normalisation par lots (batch normalization)après chaque deux couches de convolution pour rendre notre réseau plus rapides et plus stables.

Pour la partie de classification est composée de d'une couche de Flatten et deux couches totalement connectées (fully connected FC), la première avec 100 neurones, et une fonction

d'activation de type 'Relu', la deuxième est une couche de sorite avec 10 neurones et une fonction d'activation de type 'softmax'.

Pour éviter le surapprentissage, on a utilisé la régularisation dropout à la fin de chaque deux couches de convolution et aussi après la première couche caché à taux de 25%, afin de se réfère à les unités ignorées dans le réseau pendant la phase d'entraînement, et lequel ne sont pas considérées pendant une passe avant ou arrière particulière.

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 28, 28, 32)        320
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 14, 14, 32)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 14, 14, 32)        9248
_________________________________________________________________
batch_normalization (BatchNo (None, 14, 14, 32)        128
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 7, 7, 32)          0
_________________________________________________________________
dropout (Dropout)            (None, 7, 7, 32)          0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 7, 7, 64)          18496
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 3, 3, 64)          0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 3, 3, 64)          36928
_________________________________________________________________
batch_normalization_1 (Batch (None, 3, 3, 64)          256
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 1, 1, 64)          0
_________________________________________________________________
dropout_1 (Dropout)          (None, 1, 1, 64)          0
_________________________________________________________________
flatten (Flatten)            (None, 64)                0
_________________________________________________________________
dense (Dense)                (None, 100)               6500
_________________________________________________________________
dropout_2 (Dropout)          (None, 100)               0
_________________________________________________________________
dense_1 (Dense)              (None, 10)                1010
=================================================================
Total params: 72,886
Trainable params: 72,694
Non-trainable params: 192
_________________________________________________________________
```

Figure 4.7 : Configuration du modèle

Pour évaluer notre modèle nous avons utilisé la technique de validation croisée K-fold. Nous allons effectuer une validation croisée 5 fois dans ce modèle. Voir la figure (4.8).

Figure 4.8: Validation croisée 5-fold.

4.4.1.1 Résultats obtenus pour le premier modèle

I. Cas 01

a) La précision

La précision représente la somme des prédictions justes divises sur la somme des observations totales, il est calculé pour l'ensemble d'apprentissage et validation. Plus formellement, il est défini comme le nombre de vrais positifs et de vrais négatifs divisé par le nombre de vrais positifs, de vrais négatifs, de faux positifs et de faux négatifs. {Accuracy = (TP+TN) / (TP+FP+FN+TN)} :

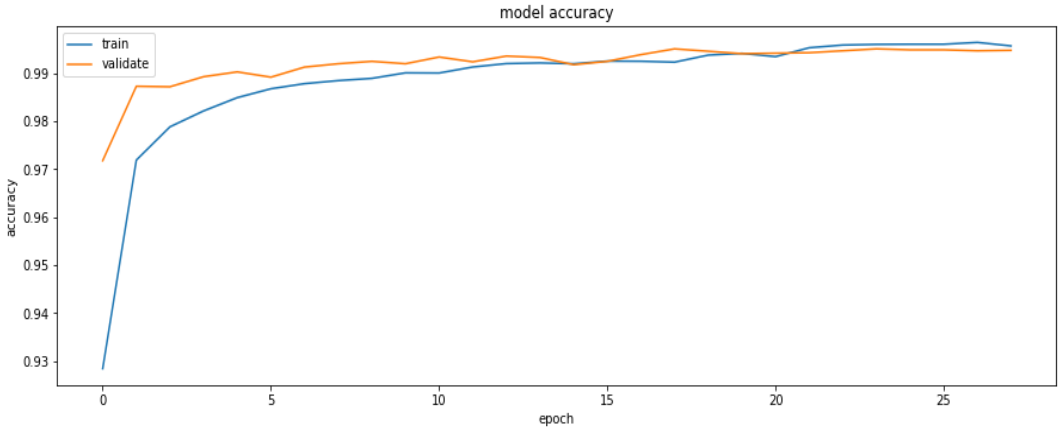

Figure 4.9 : Courbe de la précision pour l'apprentissage et la validation

Dans ce cas la première couche totalement connectée dans notre modèle nous proposons avec 256 neurones, et nous avons utilisé un optimiseur SGD avec taux d'apprentissage (learning rate) (0.01) et momentum (0.9).

On remarque bien que la précision augment avec le nombre d'epochs, (une epoch est une prédiction suivie par une rétro-propagation dans le réseau des neurones avec l'ajustement de poids des neurones). La précision a atteint le seuil de 99.47% au bout des 28 epochs, malgré nous avons utilisé 30 epochs, il arrêté tôt (early stopping), on remarque aussi qu'il ya le surapprentissage (overfitting).

b) La perte

La perte représente la somme des prédictions fausses divisé sur la somme des observations totales, il est calculé pour l'ensemble d'apprentissage et validation :

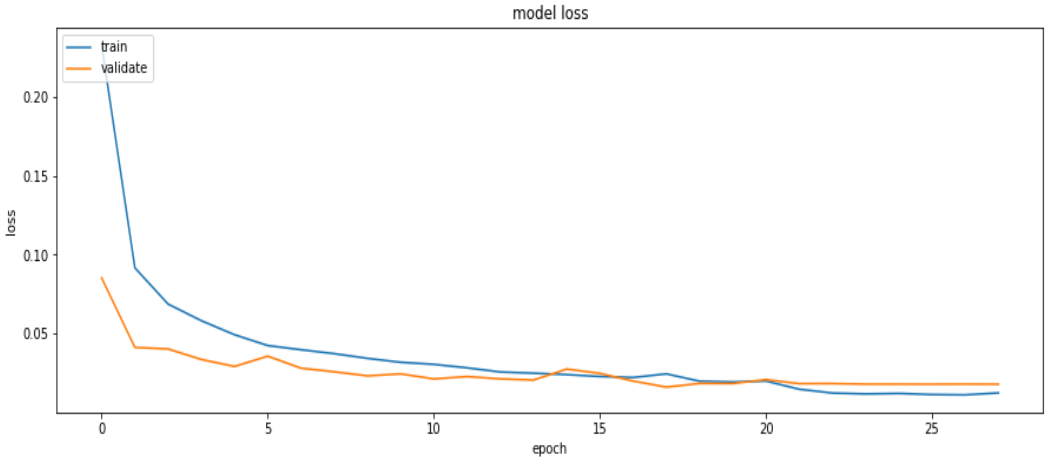

Figure 4.10 : courbe de la perte pour l'apprentissage et la validation

De même on remarque que la perte a atteint de 0.01% au bout de 28 epochs.

II. Cas 02

Dans ce cas nous avons réduit seulement le nombre de neurones à 128 dans la première couche totalement connectée dans notre modèle.

a) La précision :

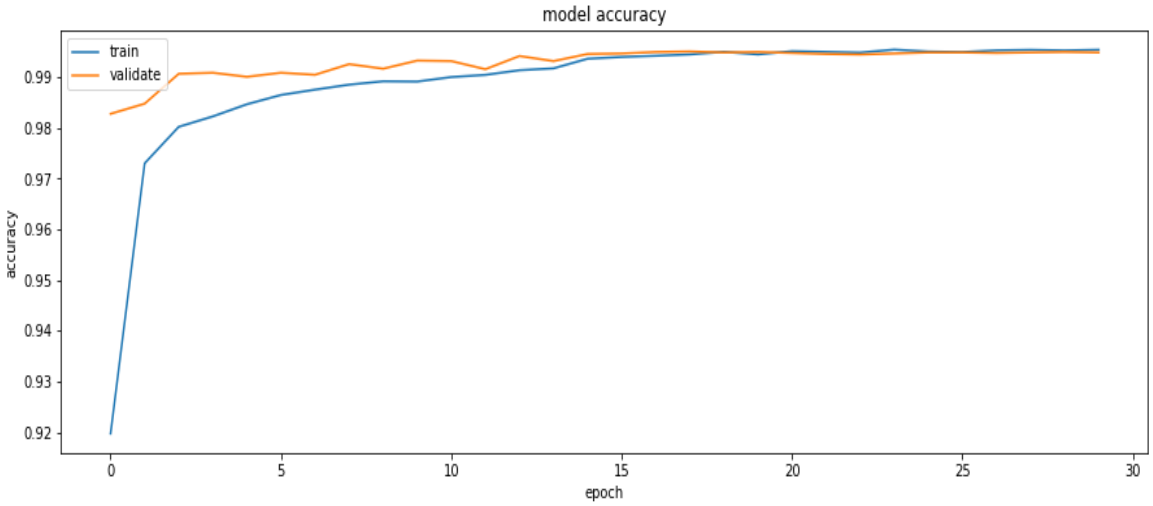

Figure 4.11 : Courbe de la précision pour l'apprentissage et la validation

On remarque que la précision a diminué à 99.44% par rapport au premier cas, mais aussi le surapprentissage a diminué un peu.

b) La perte

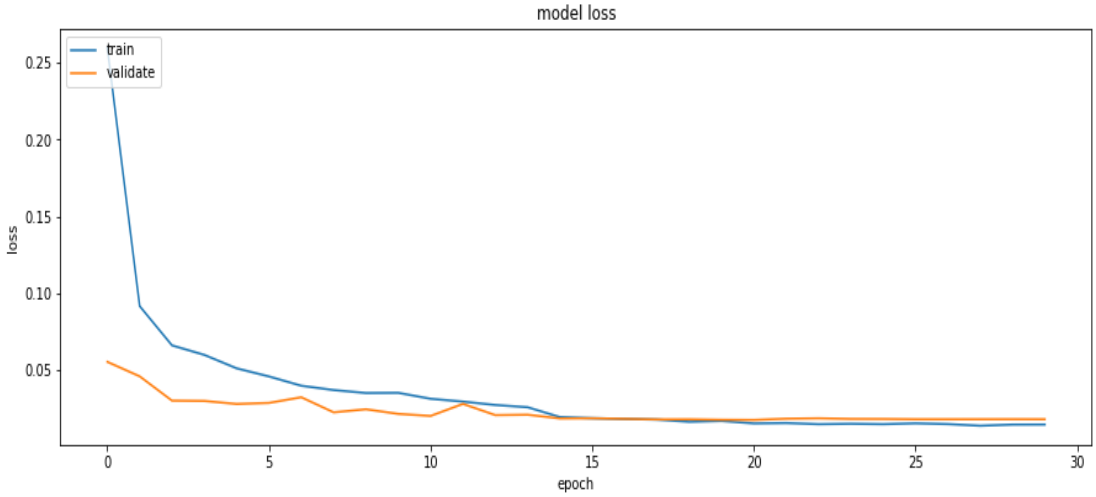

Figure 4.12 : Courbe de la perte pour l'apprentissage et la validation

On remarque que la perte est restée la même par rapport au premier cas, alors que le surapprentissage a diminué un peu.

III. Cas 03

Dans ce cas nous avons réduit le nombre de neurones à 100 dans la première couche totalement connectée.

a) La précision

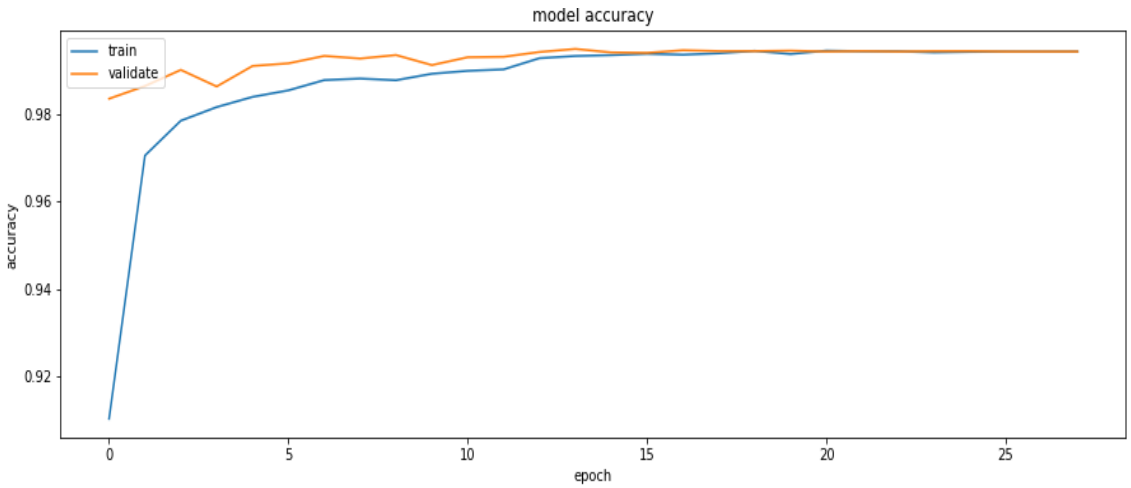

Figure 4.13 : Courbe de la précision pour l'apprentissage et la validation

b) La perte

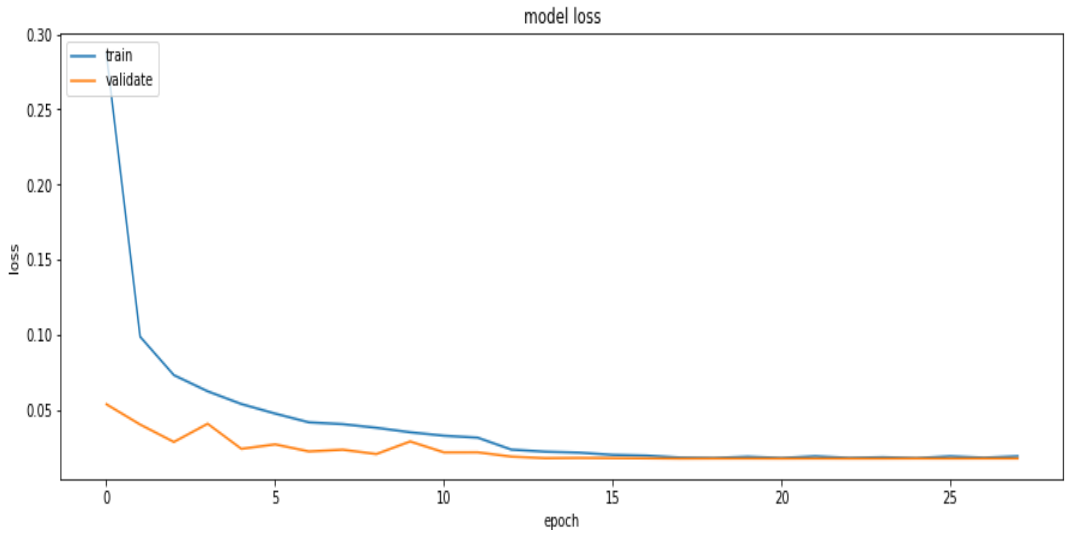

Figure 4.14 : Courbe de la perte pour l'apprentissage et la validation

On remarque que la précision presque est restée la même par rapport à deuxième cas à 99.45%, et le sur apprentissage a légèrement diminué. Il est clair dans la courbe de perte.

IV. Cas 04 :

Dans ce cas, nous avons laissé le nombre de neurones 100 dans la première couche totalement connectée, et modifié les paramètres de l'optimiseur SGD où nous avons supprimé momentum et on laisse le taux d'apprentissage (0.01).

a) La précision :

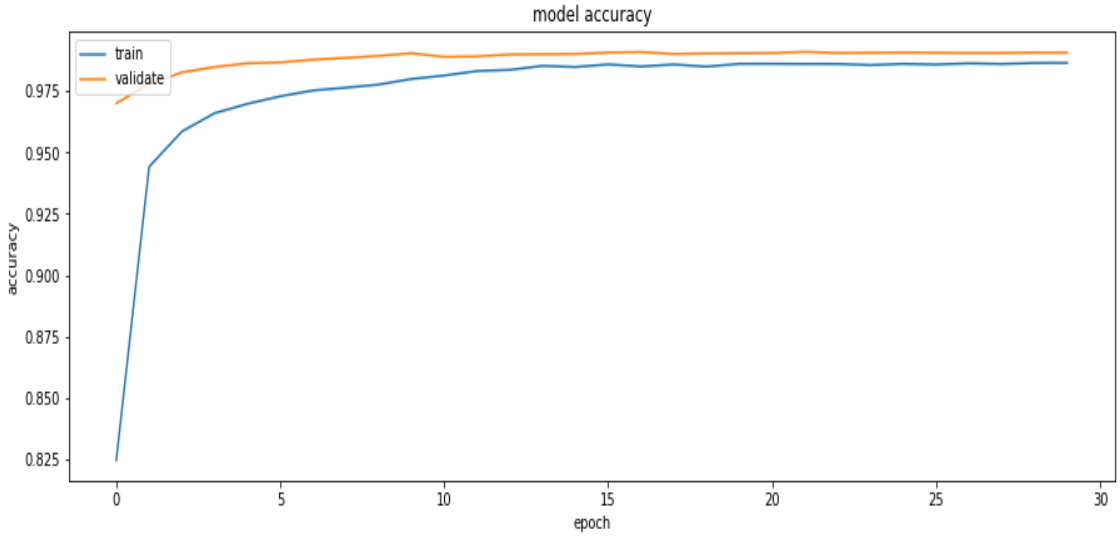

Figure 4.15 : Courbe de la précision pour l'apprentissage et la validation

b) La perte

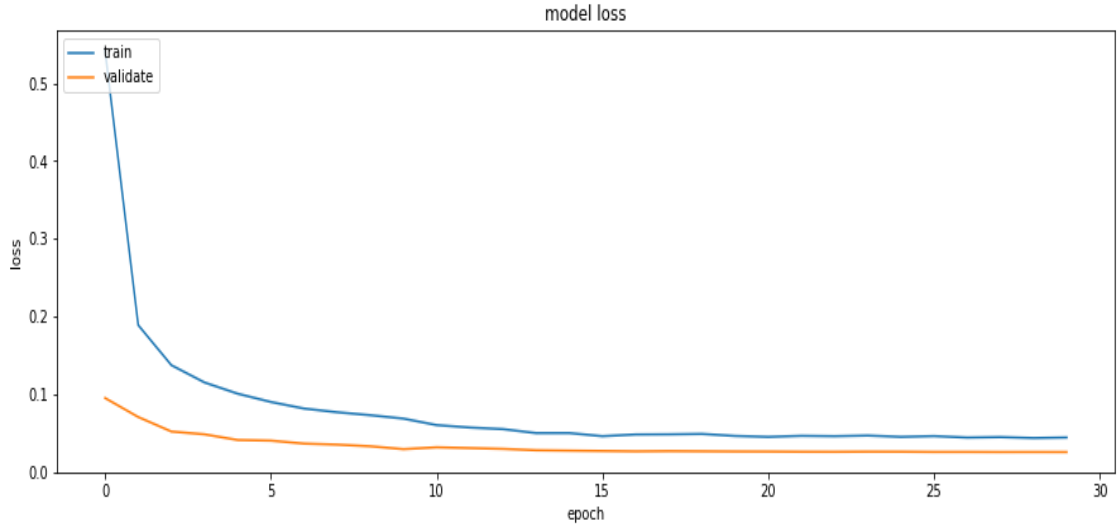

Figure 4.16 : Courbe de la perte pour l'apprentissage et la validation

On remarque dans ce cas une bonne amélioration pour les deux courbes car on s'est évité le surapprentissage, mais la précision a diminué à 99.31% et la perte a augmenté à 0,02%.

Pour comprendre ce score, il est intéressant de regarder les résultats classe par classe grâce à une matrice de confusion. Voir la figure (4.17), et le rapport de classification dans la figure (4.18).

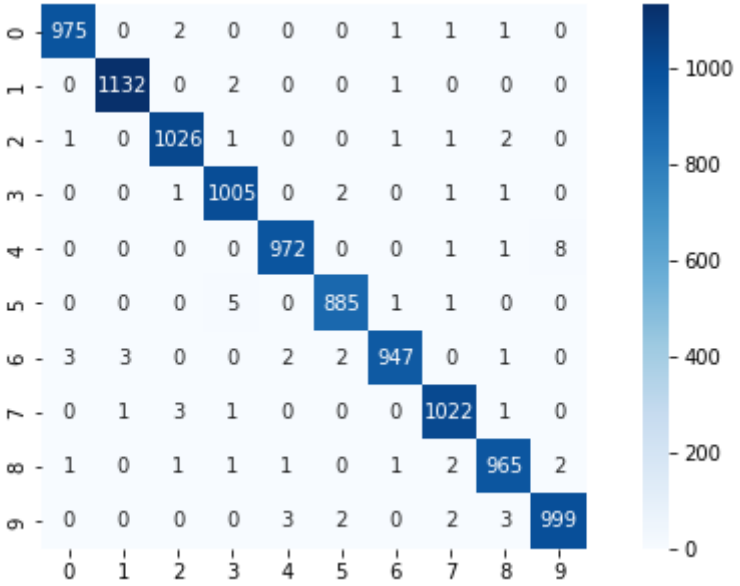

Figure 4.17 : La matrice de confusion

	precision	recall	f1-score	support
0	0.99	0.99	0.99	980
1	1.00	1.00	1.00	1135
2	0.99	0.99	0.99	1032
3	0.99	1.00	0.99	1010
4	0.99	0.99	0.99	982
5	0.99	0.99	0.99	892
6	0.99	0.99	0.99	958
7	0.99	0.99	0.99	1028
8	0.99	0.99	0.99	974
9	0.99	0.99	0.99	1009
accuracy			0.99	10000
macro avg	0.99	0.99	0.99	10000
weighted avg	0.99	0.99	0.99	10000

Figure 4.18 : Le rapport de classification

4.4.2 Le deuxième modèle (AHCD)

4.4.2.1 Le classifieur SVM

Dans notre modèle nous avons utilisé le classifieur SVM, et parmi les difficultés majeures liées à l'utilisation de ce classifieur, la nécessité d'adapter les variables conditionnant le processus d'apprentissage. Ces variables sont appelées hyper paramètres.la méthode des SVM implique la sélection de plusieurs paramètres : le type de noyau, le ou les paramètres du noyau et le paramètre de régularisation C.

Afin de trouver le meilleur ensemble des valeurs d'hyper paramètres pour notre SVM, il est nécessaire de spécifier une méthode permettant de sélectionner un modèle parmi plusieurs possibles, cette méthode que nous avons utilisée s'appelle la fonction GridSearchCV, lequel automatise la recherche d'un optimum parmi les hyper paramètres, elle utilise notamment la validation croisée.

Nous avons utilisé la fonction GridSearchCV sur notre base de données AHCD, sur l'ensemble d'hyper paramètres suivant :

params= {'kernel': ['linear','rbf','poly','sigmoid'],

'C': [0.01, 0.1, 1.0, 10.0, 100.0],

'Gamma':(0.01, 0.02, 0.03, 0.1, 1.0, 10.0, 100.0)}

Cette recherche a pris environ 8 heures pour obtenir les meilleures valeurs des hyper paramètres, qui sont : {'C': 100.0, 'gamma': 0.03, 'kernel': 'rbf'} et score égal 73.24%.

Alors, on remarque que SVM ne donne pas de bons résultats s'il est appliqué seul sur la base AHCD.

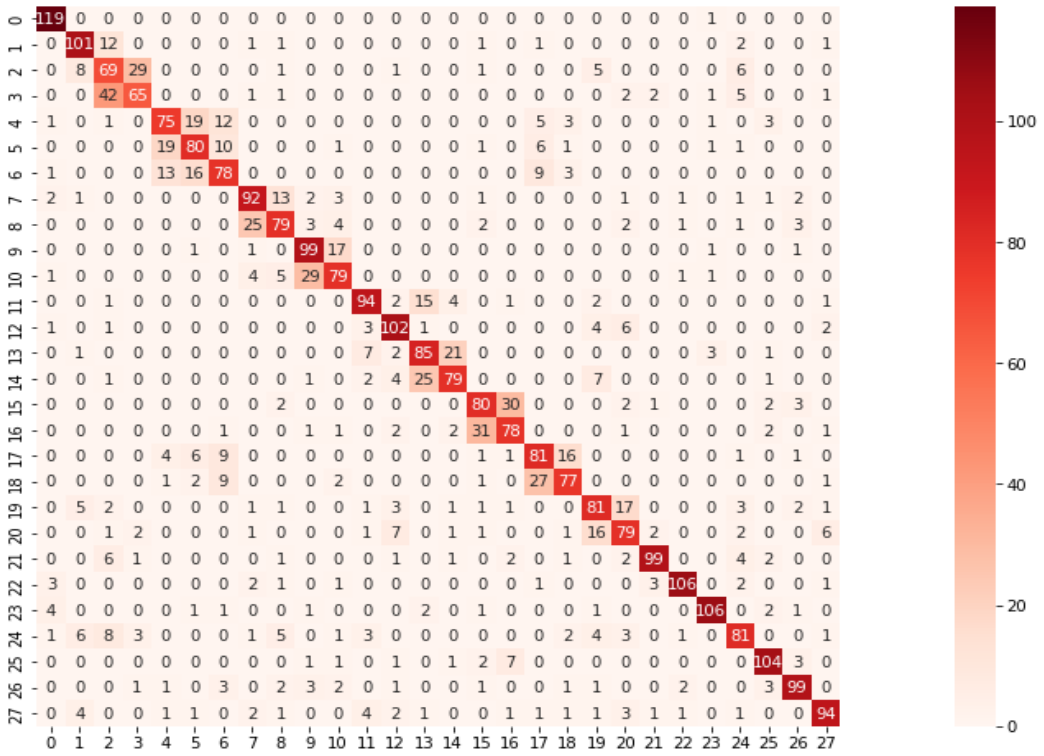

Figure 4.19 : La matrice de confusion (SVM).

	precision	recall	f1-score	support
0	0.99	0.89	0.94	133
1	0.84	0.80	0.82	126
2	0.57	0.48	0.52	144
3	0.54	0.64	0.59	101
4	0.62	0.66	0.64	114
5	0.67	0.63	0.65	126
6	0.65	0.63	0.64	123
7	0.77	0.70	0.73	131
8	0.66	0.70	0.68	113
9	0.82	0.71	0.76	140
10	0.66	0.71	0.68	112
11	0.78	0.82	0.80	115
12	0.85	0.80	0.82	128
13	0.71	0.66	0.68	129
14	0.66	0.72	0.69	110
15	0.67	0.64	0.65	125
16	0.65	0.64	0.65	121
17	0.68	0.62	0.65	131
18	0.64	0.73	0.68	106
19	0.68	0.66	0.67	122
20	0.66	0.67	0.66	118
21	0.82	0.92	0.87	108
22	0.88	0.94	0.91	113
23	0.88	0.92	0.90	115
24	0.68	0.74	0.70	110
25	0.87	0.86	0.86	121
26	0.82	0.86	0.84	115
27	0.78	0.85	0.82	110
accuracy			0.73	3360
macro avg	0.73	0.74	0.73	3360
weighted avg	0.73	0.73	0.73	3360

Figure 4.20 : Le rapport de classification (SVM).

4.4.2.2 Le classifieur CNN

Dans ce modèle nous avons utilisé quatre couches de convolution, nous proposons d'utiliser 32 filtre dans la première et deuxième couche, et 64 filtre dans la troisième et quatrième couche, dont chaque couche utilise un filtre de taille 3*3, toutes les couches de convolution avec une fonction de rectification linéaire « Relu ».

Après chaque couche de convolution on applique une couche de Maxpooling, pour réduire la taille de l'image et la quantité des paramètres, et une couche de normalisation par lots (batch normalisation) après chaque deux couche de convolution pour rendre notre réseau plus rapides et plus stables.

Pour la partie de classification est composée de d'une couche de Flatten et deux couches totalement connectée s, la première avec 512 neurones, et une fonction d'activation de type 'Relu', la deuxième est une couche de sorite avec 28 neurones et une fonction d'activation de type 'softmax'.

Pour éviter le surapprentissage, on a utilisé la régularisation dropout à la fin de chaque deux couche de convolution et après la première couche totalement connectée à taux de 25%. Nous avons utilisé aussi la technique d'augmentation des données, pour augmenter la précision et améliorer la performance de notre modèle, il est préférable d'entraînés le réseau avec beaucoup d'échantillons d'image c'est pour ça on a augmenté la base de données. Pour augmenter notre base de données on a fait quelque prétraitement sur les images (normalisation, redimensionnement, zoom, rotation).

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 32, 32, 32)        320
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 16, 16, 32)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 16, 16, 32)        9248
_________________________________________________________________
batch_normalization (BatchNo (None, 16, 16, 32)        128
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 8, 8, 32)          0
_________________________________________________________________
dropout (Dropout)            (None, 8, 8, 32)          0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 8, 8, 64)          18496
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 4, 4, 64)          0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 4, 4, 64)          36928
_________________________________________________________________
batch_normalization_1 (Batch (None, 4, 4, 64)          256
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 2, 2, 64)          0
_________________________________________________________________
dropout_1 (Dropout)          (None, 2, 2, 64)          0
_________________________________________________________________
flatten (Flatten)            (None, 256)               0
_________________________________________________________________
dense (Dense)                (None, 512)               131584
_________________________________________________________________
dropout_2 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_1 (Dense)              (None, 28)                14364
=================================================================
Total params: 211,324
Trainable params: 211,132
Non-trainable params: 192
_________________________________________________________________
```

Figure 4.21 : Configuration du modèle.

Dans ce cas nous avons utilisé un optimiseur Adam avec taux d'apprentissage (learning rate)=(0.001), beta_1= (0.99), beta_2= (0.9999), avec 50 epochs. Après de nombreuses tentatives, nous avons obtenu les meilleurs résultats suivants :

a) La précision

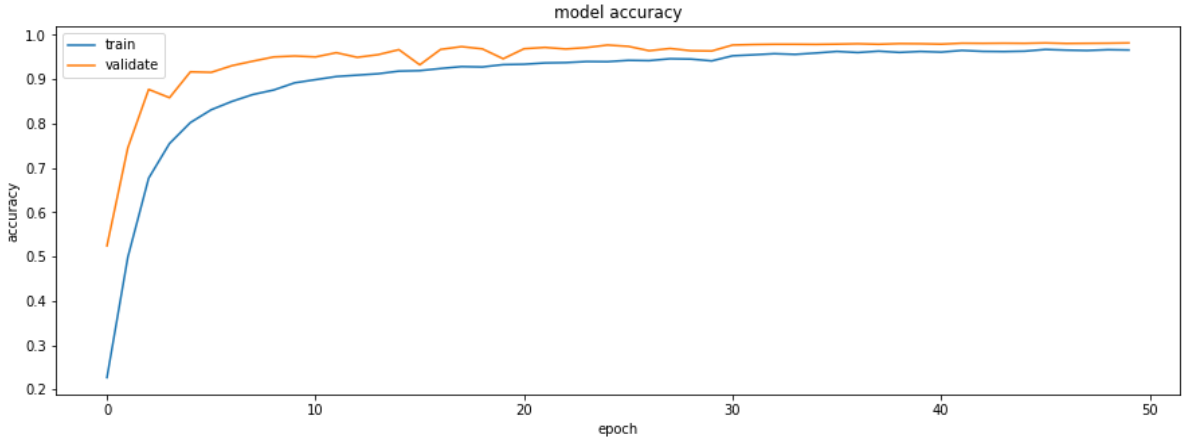

Figure 4.22 : Courbe de la précision pour l'apprentissage et la validation(CNN).

b) La perte

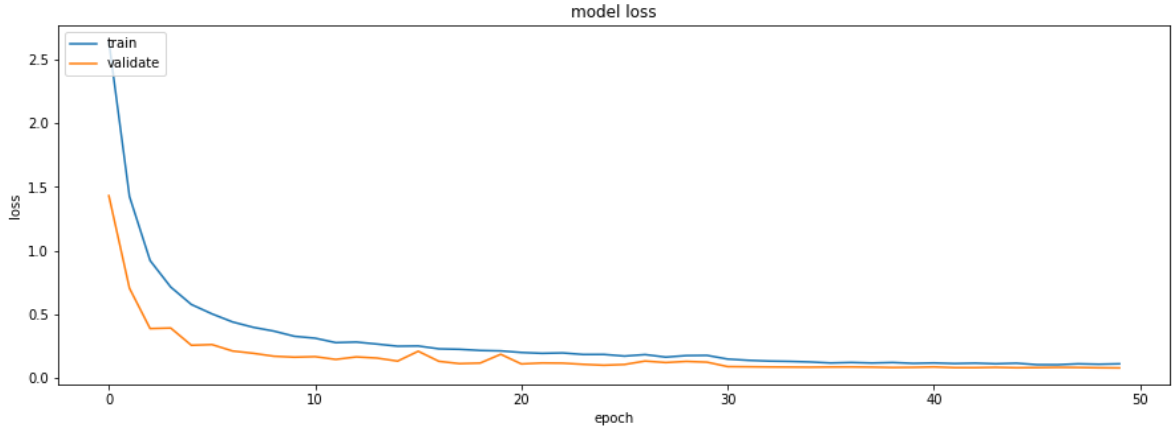

Figure 4.23 : Courbe de la perte pour l'apprentissage et la validation (CNN).

On remarque que le classifieur CNN a donné des bons résultats malgré il est appliqué seul sur la base AHCD, où la précision a atteint 98,15% et la perte à 0,07% au bout des 50 epochs, nous remarquons aussi une chose important qui attire l'attention qu'il n'y a pas de surapprentissage (overfitting), cela est dû à l'utilisation de dropout et l'augmentation des données.

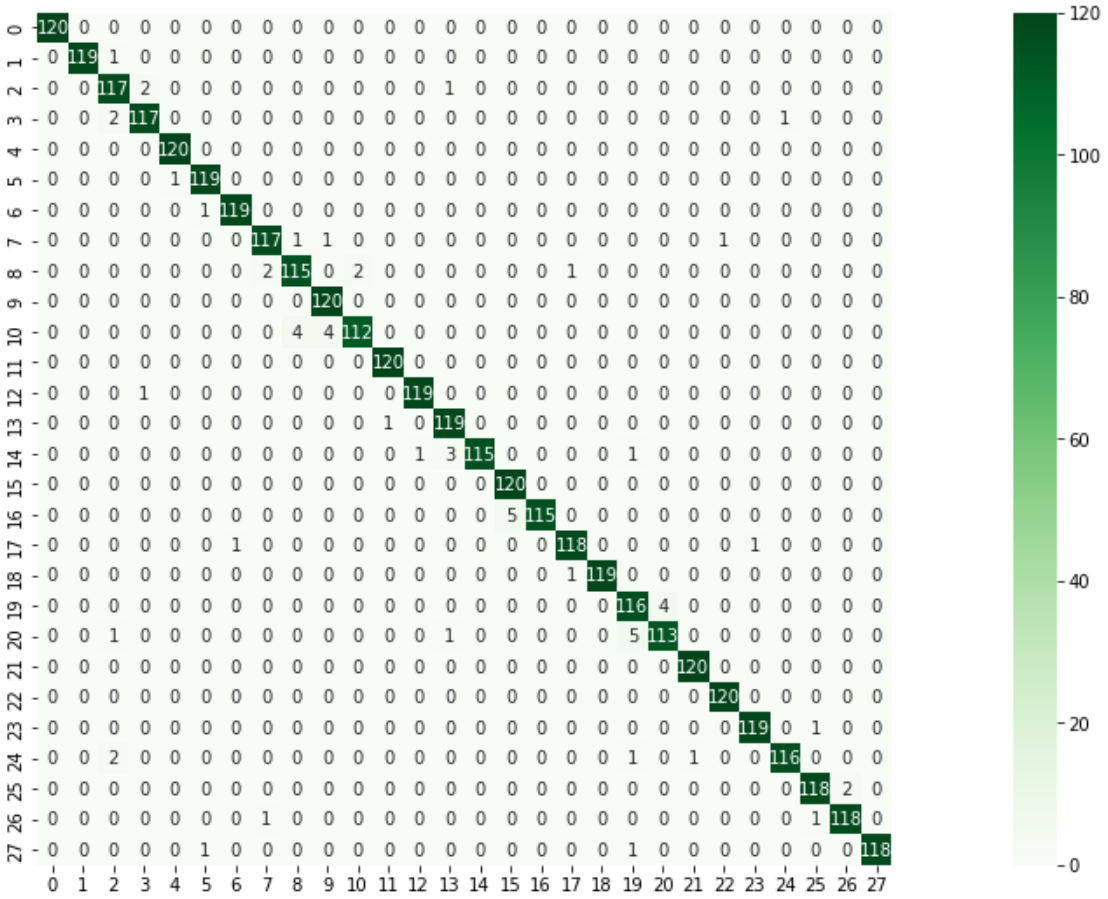

Figure 4.24 : La matrice de confusion (CNN).

	precision	recall	f1-score	support
0	1.00	1.00	1.00	120
1	1.00	0.99	1.00	120
2	0.95	0.97	0.96	120
3	0.97	0.97	0.97	120
4	0.99	1.00	1.00	120
5	0.98	0.99	0.99	120
6	0.99	0.99	0.99	120
7	0.97	0.97	0.97	120
8	0.96	0.96	0.96	120
9	0.96	1.00	0.98	120
10	0.98	0.93	0.96	120
11	0.99	1.00	1.00	120
12	0.99	0.99	0.99	120
13	0.96	0.99	0.98	120
14	1.00	0.96	0.98	120
15	0.96	1.00	0.98	120
16	1.00	0.96	0.98	120
17	0.98	0.98	0.98	120
18	1.00	0.99	1.00	120
19	0.94	0.97	0.95	120
20	0.97	0.94	0.95	120
21	0.99	1.00	1.00	120
22	0.99	1.00	1.00	120
23	0.99	0.99	0.99	120
24	0.99	0.97	0.98	120
25	0.98	0.98	0.98	120
26	0.98	0.98	0.98	120
27	1.00	0.98	0.99	120
accuracy			0.98	3360
macro avg	0.98	0.98	0.98	3360
weighted avg	0.98	0.98	0.98	3360

Figure 4.25 : Le rapport de classification (CNN).

4.4.2.3 Combinaison entre CNN et SVM

Dans cette partie, nous présentons l'architecture de notre système AHCD basé sur CNN et SVM, où CNN est considéré comme un algorithme d'apprentissage en profondeur, sur lequel la technique de "dropout" a été appliquée pendant l'entraînement. Notre système proposé a été adapté en modifiant le classificateur entraînable du CNN avec un classificateur SVM. Notre objectif est de mélanger les capacités respectives du CNN et du SVM pour obtenir un nouveau système de reconnaissance efficace inspiré des deux formalismes. Comme le montre l'exemple de la figure. (4.26).

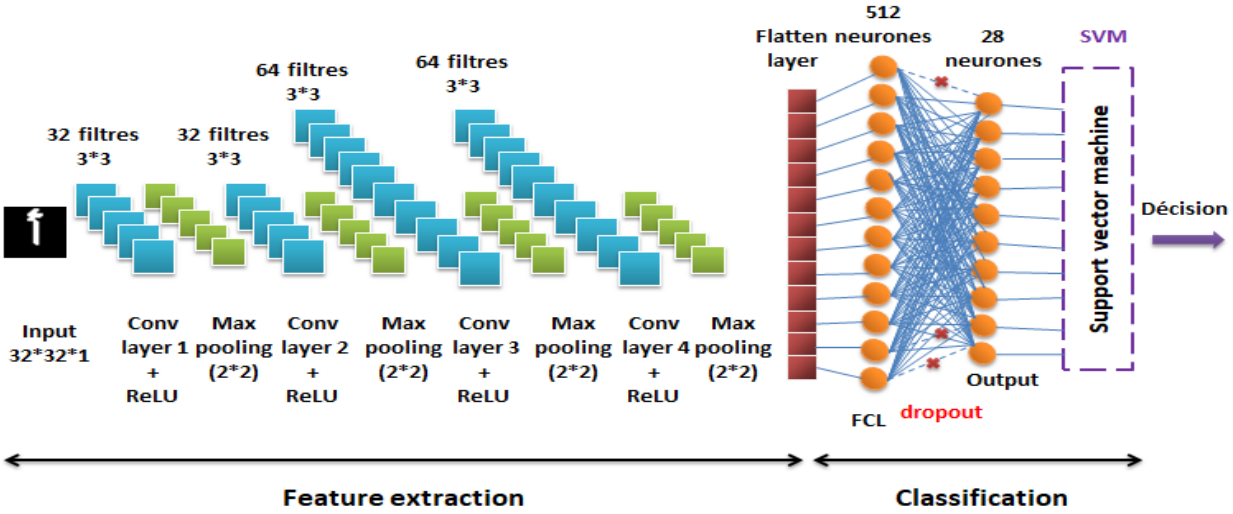

Figure 4.26 : l'architecture du modèle combinaison CNN-SVM.

L'idée générale est que le CNN a été appliqué pour extraire les caractéristiques des images de caractères, puis SVM a été utilisé pour la classification.

La couche finale du CNN a été remplacée par SVM avec un noyau RBF pour la classification. Les sorties des unités totalement connectée s sont prises par le SVM comme vecteur de caractéristiques pour le processus d'entrainement. Après cela, la phase de l'entrainement se poursuit jusqu'à la réalisation d'un bon entrainement. Enfin, la classification sur l'ensemble de test a été effectuée par le classifieur SVM avec ces caractéristiques automatiquement extraites.

En raison de l'utilisation d'un grand nombre de données et de paramètres, un surapprentissage peut se produire. Donc, pour prévenir notre réseau de ce problème et pour l'améliorer, le "dropout" est appliqué.

Les résultats obtenus pour ce modèle

Dans le 1er cas, on extrait les sorties du couche Flatten et les donnons en entrée au classificateur SVM., on obtient des résultats suivants :

Avec SVC (C= 100, gamma= 0.03, kernel='rbf'), le score égal 77.5%.

Tandis qu'avec SVC ( ) par défaut où (C=1, kernel="rbf", gamma="scale"), le score s'est bien amélioré à 98.12%.

Dans le 2ème nous extrayons les sorties de la première couche FC de notre CNN et les donnons en entrée au classificateur SVM, on obtient :

Avec SVC (C= 100, gamma= 0.03, kernel='rbf'), le score a augmenté à 98.15%, et avec SVC (), le score a atteint à 98.21% c'est le meilleur résultat que nous pouvons obtenir .on remarque que lorsque on utilise le classifieur SVM avec le classifieur CNN, cela donne de très bons résultats.

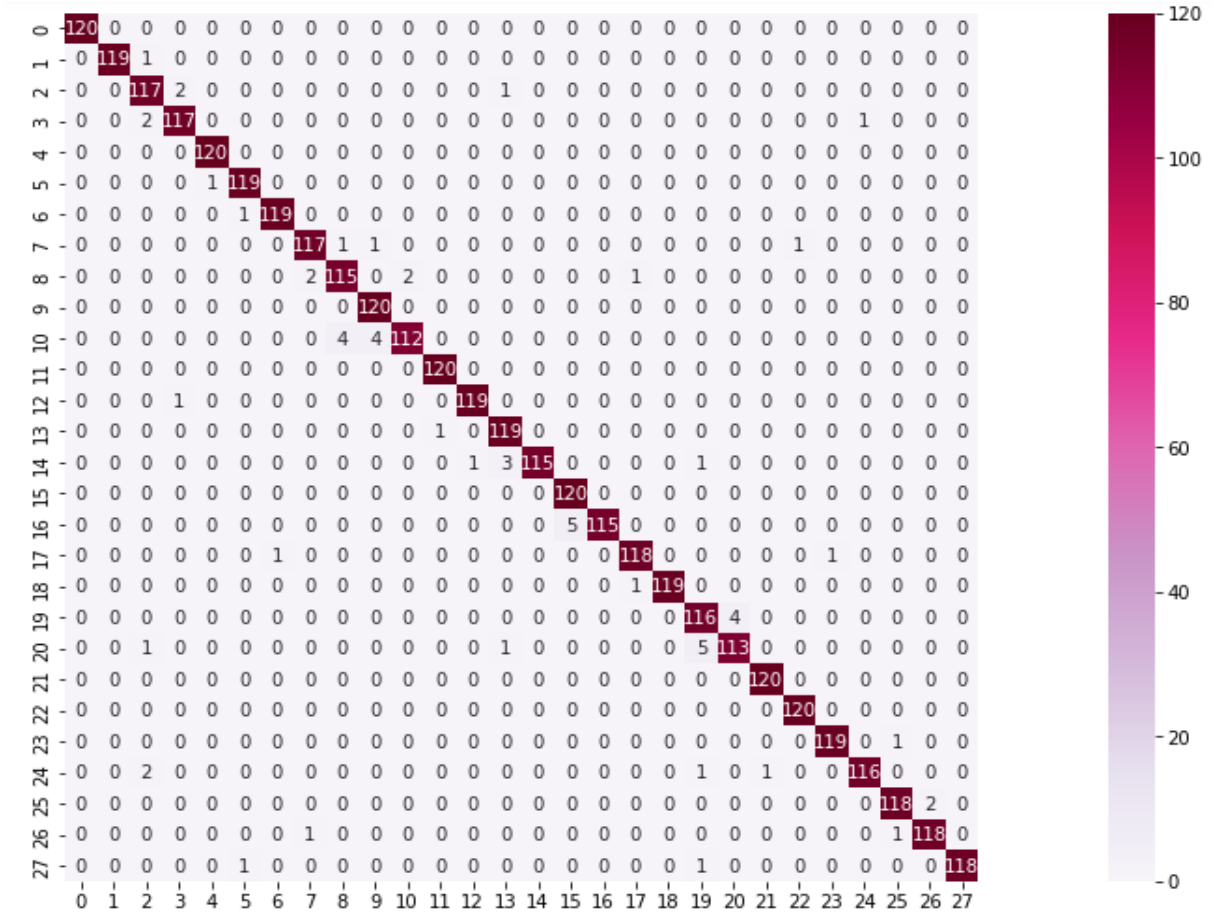

Figure 4.27 : La matrice de confusion (combinaison CNN-SVM).

	precision	recall	f1-score	support
0	1.00	1.00	1.00	120
1	1.00	0.99	1.00	120
2	0.95	0.97	0.96	120
3	0.97	0.97	0.97	120
4	0.99	1.00	1.00	120
5	0.98	0.99	0.99	120
6	0.99	0.99	0.99	120
7	0.97	0.97	0.97	120
8	0.96	0.96	0.96	120
9	0.96	1.00	0.98	120
10	0.98	0.93	0.96	120
11	0.99	1.00	1.00	120
12	0.99	0.99	0.99	120
13	0.96	0.99	0.98	120
14	1.00	0.96	0.98	120
15	0.96	1.00	0.98	120
16	1.00	0.96	0.98	120
17	0.98	0.98	0.98	120
18	1.00	0.99	1.00	120
19	0.94	0.97	0.95	120
20	0.97	0.94	0.95	120
21	0.99	1.00	1.00	120
22	0.99	1.00	1.00	120
23	0.99	0.99	0.99	120
24	0.99	0.97	0.98	120
25	0.98	0.98	0.98	120
26	0.98	0.98	0.98	120
27	1.00	0.98	0.99	120
accuracy			0.98	3360
macro avg	0.98	0.98	0.98	3360
weighted avg	0.98	0.98	0.98	3360

Figure 4.28: Le rapport de classification (combinaison CNN-SVM).

4.5 Conclusion

Nous avons présenté dans ce chapitre une approche de classification pour la reconnaissance des caractères arabes manuscrits, basée sur les réseaux de neurones convolutionnels (CNN) et les machines à vecteurs supports (SVM), pour cela, on a proposé plusieurs modèles et on a tester avec la base de données MNIST et AHCD.

Puis, nous avons montré les différents résultats obtenus en termes de précision, d'erreur, et en termes d'évaluation : la matrice de confusion et le rapport de classification. La comparaison des résultats trouvés a montré que le nombre d'époques, la profondeur de réseaux, les paramètres d'optimiseur, sont des facteurs importants pour l'obtention de meilleurs résultats.

Conclusion général

Actuellement, aucune solution optimale au problème de la reconnaissance de l'écriture manuscrite n'est encore connue. Bien que, des progrès importants dans ce domaine aient permis le développement de quelques systèmes pratiques, atteindre un taux de reconnaissance comparable aux performances humaines. Pour cette raison, ce domaine de recherche constitue encore un terrain fertile pour de futurs travaux.

L'objectif principal de toute recherche est d'aboutir à des résultats satisfaisants et de mettre en œuvre des systèmes performants en se focalisant sur le choix de la méthode qui accroît au mieux la précision. Dans ce mémoire, Nous avons présenté trois systèmes des reconnaissances des caractères arabes manuscrits, appliqué sur la base d'images AHCD qui contient 16 800 caractères écrits par 60 participants :

- Le premier modèle basé sur les machines à vecteurs supports (SVM), le système ne donné pas de bons résultats, une précision de 73.24%.
- Le deuxième modèle basé sur un réseau de neurones convolutif (CNN). Le système a donné des bons résultats, une précision de 98.15%.
- Un modèle de combinaison CNN-SVM a été proposé ce système a donné des très bons résultats, une précision de 98.21%, c'est le meilleur résultat que nous pouvons obtenir. Ce qui démontre l'efficacité et la performance du modèle proposé.

En conclusion, ce projet nous a permis d'acquérir de nouvelles connaissances. Nous avons pu découvrir au cours de ce travail de nouvelles notions telles que la reconnaissance de caractères arabes manuscrits, le Depp Learning, le CNN et le SVM. On a pu palper d'une façon concrète les différentes difficultés relatives à l'implémentation d'un modèle d'apprentissage profond.

Les références :

[01] S. Mori, H. Nishida, & H. Yamada, Optical character recognition (New York, NY: John Wiley & Sons Inc.,1999).

[02] P.M. Lallican, C. Viarp-Gaudin, S. Knerr : From o_-line to on-line handwriting recognition. Proc. 7th workshop on frontiers in handwriting recognition, pp. 303-312, Amsterdam 2000.

[03] N. Arica & F. Yarman-Vural, Optical character recognition for cursive handwriting, IEEE Trans. Pattern Analysis and Machine Intelligence, 24(6), 2002, 801-813.

[04] Abdelaziz BELAID – LORIA. "Reconnaissance de l'écriture en ligne par les réseaux de neurones dynamiques " thèse de magister en informatique université sciences et technologie Oran Mohamed Boudiaf 21/01/2015.

[05] N. Benamara : « Utilisation des modèles de Markov cachés planaires en reconnaissance de l'écriture arabe imprimée ». Thèse de doctorat, spécialité Génie Electrique, Université des sciences, des Techniques et de médecine de Tunis II, 1999.

[06] J. Trenkle, A. Gillies, S.Schlosser : _ An o_-line Arabic recognition system for machine printed documents _. Proc. Of the symposium on document image understanding technology (SDIUT'97), pp. 155-161 1997.

[07] A. Amin, H.B. Al-Sadoun , S. Fisher : _ Handprinted A rabic character recognition system using an arti_cial network _. Pattern recognition, vol. 29, No 4, pp. 663-675,
1997.

[08] Pritpal Singh and Sumit Budhiraja, "Feature extraction and classification techniques in OCR systems for handwritten Gurmukhi Script a survey," International Journal of Engineering Research and Applications (IJERA), vol. 1, no. 4, pp. 1736--1739, 2011.

[09] BEKHELIFI. O. Support Vector Machines. Rapport technique. Université MOHAMMED BOUDIAF, Oran.

[10] Zennaki M ; Mamouni, E. M ; Sadouni, K : « SVM Model Selection Using PSO for Learning Handwritten Arabic Characters ». Université des Sciences et de la Technologie d'Oran Mohamed Boudiaf USTO-MB, BP 1505 El M'naoeur, 311000, Oran Algérie.

[11] V. Blanz, B. Scholkopf, H.H. Bulthoff, C. Burges, V. Vapnik, and T. Vetter. « Comparison of view-based object recognition algorithms using realistic 3d models ». In ICANN, pages 251–256, 1996.

[12] Bernhard E. Boser, Isabelle M. Guyon, Vladimir N. Vapnik, "A training algorithm for optimal margin classifiers", COLT '92 : Proceedings of the fifth annual workshop on Computational learning theory July 1992 Pages 144–152

[13] Stephan R. Sain, " The Nature of Statistical Learning Theory", « Southern Methodist University », Technometrics.

[14] Cortes C, Vapnik V, "Support-vector networks". Machine Learning, 20, (3) :273-297,1995.

[15] Ayat N., "sélection de modèle automatique des machines à vecteurs de support : application à la reconnaissance d'images de chiffres manuscrits" Thèse de doctorat Montréal, le 20 janvier 2004.

[16] Mohamadally H., Fomani B. "SVM : Machines à Vecteurs de Support ou Séparateurs à Vastes Marges" Versailles St Quentin, France 16 janvier 2006.

[17] Milgram J., "Contribution à l'intégration des machines à vecteurs de support au sein des systèmes de reconnaissance des formes : Application à la lecture automatique de L'écriture manuscrite". Thèse ÉTS Montréal, le 29 juin 2007.

[18] Steve R. G, "Support Vector Machines for Classification and Regression" Universityof Southampton, 10 May 1998.

[19] Vapnik, V. "The Nature of Statistical Learning Theory". Springer-Verlag, 1998.

[20] Antoine C., Laurent M., "Apprentissage artificiel : concepts et algorithmes", Préface de Tom Mitchell. Édition EYROLLES deuxième tirage 2003.

[21] L. Hamel. "Knowledge discovery with support vector machines". Wiley Edition, 2009.

[22] Chang Y. "Feature ranking using linear svm. Causation and Prediction Challenge" Challenges in Machine Learning, 2:47, 2008.
[23] Joachims T. "Text categorization with support vector machines : Learning with any relevant features". Springer, 1998.

[24] Munteanu D. "A quick survey of text categorization algorithms". Annals of University Dunarea de Jos 1/ of Galati, 1, 2007.

[25] Robert H., Trevor, Tibshirani. "Stanford statistical learning online course SVM lecture. Example and comparison with logistic regression". Web Video, Nov 2013. Accesed 15Sep2014.

[26] Gilles Lebrun, « Sélection de modèles pour la classification supervisée avec des SVM (Séparateurs à VasteMarge). Application en traitement et analyse d'images ». Doctorat de l'universite de Caen/basse-Normandie, soutenue le 24 novembre 2006.

[27] Russakovsky O, Deng J, Su H et al (2015) ImageNet Large Scale Visual Recognition Challenge. Int J Comput Vis 115 :211–252.

[28] Krizhevsky A, Sutskever I, Hinton GE (2012) ImageNet classification with deep convolutional neural networks. Adv Neural Inf. Process Syst 25. Available online at: https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neuralnetworks.pdf. Accessed 22 Jan 2018

[29] M. HARGRAVE, « Deep Learning Definition, » 06 April 2021. [En ligne]. Available : https://www.investopedia.com/terms/d/deep-learning.asp. [Accès le 23 April 2021].

[30] Hubel DH, Wiesel TN (1968) Receptive fields and functional architecture of monkey striate cortex. J Physiol 195:215–243

[31] Goodfellow I, Bengio Y, Courville A (2016) Deep learning. MIT Press. http://www.deeplearningbook.org

[32] Y. LeCun, Y. Bengio, and G. Hinton, "Deep Learning", Nature, vol. 521, pp. 436-444, May 2015.

[33] Rikiya Yamashita, Mizuho Nishio, Richard Kinh Gian Do & Kaori Togashi, "Convolutional neural networks: an overview and application in radiology", Insights Imaging, Springer, vol. 9, pp. 611–629, June 2018. DOI: 10.1007/s13244-018-0639-9

[34] LinM, Chen Q, Yan S (2013) Network in network. arXiv. Available online at: https://arxiv.org/pdf/1312.4400.pdf. Accessed 22 Jan 2018.

[35] Sohaib, Asif et K.Amjad, «Automatic COVID-19 Detection from chest radiographic images using,» medRxiv preprint doi: https://doi.org/10.1101/2020.11.08.20228080, 2020.

[36] Dong-yuan Ge, Xi-fan Yao, Wen-jiang Xiang, Xue-jun Wen, En-chen Liu, "Design of High Accuracy Detector for MNIST Handwritten Digit Recognition Based on Convolutional Neural Network", International Conference on Intelligent Computation Technology and Automation (ICICTA), IEEE, pp. 658-662, March 2020. DOI : 10.1109/ICICTA49267.2019.00145.

[37] Danqing Liu, "A Practical Guide to ReLU", Medilum, November 2017. Available at: https://medium.com/@danqing/a-practical-guide-torelu-b83ca804f1f7.

[38] Wiesel et Hubel, «Receptive fields of single neurones in the cat's striate cortex,» The Journal of Physiology, vol. 148, n° %13, p. 574–591, 1959.

[39] Imane.Nedjar, « Medical images indexation and annotation [Thèse de doctorat, Université de Tlemcen] » Tlemcen, 2015.

[40] Y. LeCun et al, «Handwritten digit recognition with a backpropagation network,» in Advances in neural information processing systems, pp. 396-404, 1990.

[41] C.Szegedy, W.Liu, Y.Jia, P.Sermane, S.Reed, D.Anguelov, D.Erhan, V.Vanhoucke et A.Rabinovich, «Going deeper with convolutions,» in Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1-9, 2015.

[42] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto et H. Adam, « Mobilenets: Efficient convolutional neural networks for mobile vision applications,» arXiv preprint arXiv:1704.04861, 2017.

[43] XIE, Qizhe, LUONG, Minh-Thang, HOVY, Eduard et al, « Self-training with noisy student improves imagenet classification, » arXiv preprint arXiv :1911.04252, 2019.

[44] Daoud fouad et louli farouk « La reconnaissance des caractères arabes manuscrits par les reseaux convolutifs » Thèse magistère en informatique université Blida 2007/2008

[45] OULMI Mehdi et KALOUNE Salim Titre « Classification d'objets avec le Deep Learning », Master en informatique Université Bouira 2017/2018.

[46] A. El-Sawy, M. Loey and H. El-Bakry, "Arabic Handwritten Characters Recognition Using Convolutional Neural Network," WSEAS Transactions on Computer Research, vol. 5, pp. 11-19, 2017.

[47] LeCun, Y., Bottou, L., Bengio, Y. and Haffner, P. (1998) “Gradient-based learning applied to document recognition.” In : Proceedings of the IEEE 86(11):2278-2324.