RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE
MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA RECHERCHE SCIENTIFIQUE

ÉCOLE NATIONALE POLYTECHNIQUE

المدرسة الوطنية المتعددة التقنيات
Ecole Nationale Polytechnique

Département d'Électronique

Mémoire de projet de fin d'études

Pour l'obtention du diplôme d'ingénieur d'état en Électronique

---

Intégrale Floue avec Optimisation par Essaim de Particules pour la Classification des Maladies des Feuilles de Plantes

---

Melissa MESSAOUDI

Sous la direction de **Dr. BOUADJENEK Nesrine** ENP

Présenté et soutenu publiquement le 01/07/2024 auprès des membres du jury :

Présidente	Mme. Rachida	TOUHAMI	Prof.	ENP, Alger
Promotrice	Mme. Nesrine	BOUADJENEK	Dr.	ENP, Alger
Examinateur	M. Sid-Ahmed	BERRANI	Prof.	ENP, Alger

ENP 2024

Électronique / ENP, 10 Rue des Frères OUDEK, El Harrach 16200 Alger Algérie.

RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE
MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA RECHERCHE SCIENTIFIQUE

ÉCOLE NATIONALE POLYTECHNIQUE

المدرسة الوطنية المتعددة التقنيات
Ecole Nationale Polytechnique

Département d'Électronique

Mémoire de projet de fin d'études

Pour l'obtention du diplôme d'ingénieur d'état en Électronique

---

Intégrale Floue avec Optimisation par Essaim de Particules pour la Classification des Maladies des Feuilles de Plantes

---

Melissa MESSAOUDI

Sous la direction de **Dr. BOUADJENEK Nesrine** ENP

Présenté et soutenu publiquement le 01/07/2024 auprès des membres du jury :

Présidente	Mme. Rachida	TOUHAMI	Prof.	ENP, Alger
Promotrice	Mme. Nesrine	BOUADJENEK	Dr.	ENP, Alger
Examinateur	M. Sid-Ahmed	BERRANI	Prof.	ENP, Alger

ENP 2024

Électronique / ENP, 10 Rue des Frères OUDEK, El Harrach 16200 Alger Algérie.

Dédicace

À mes parents, pour tout ce qu'ils ont fait pour moi.

À mes chers frères Yanis et Ghiles

À tous ceux qui me sont chers.

À vous tous.

Merci.

- Melissa

ملخص

الزراعة ضرورية لبقاء الإنسانية، لكنها تواجه تحديات كبيرة تتعلق بأمراض النباتات، التي يمكن أن تؤدي إلى خسائر كبيرة في المحاصيل. يعتبر الكشف المبكر ومنع انتشار الأمراض أمراً أساسياً. للتغلب على هذه التحديات، يستغل العديد من الباحثين تقنيات الرؤية الحاسوبية لتحليل صور أوراق النباتات وتحديد الأمراض، حيث تظهر غالبية الأعراض على الأوراق. يندرج عملنا في هذا النهج من خلال تطوير وتقييم أربعة نماذج متقدمة: نموذج شبكة عصبية تلافيفية (CNN) وثلاثة نماذج هجينة تجمع بين CNN كمستخرج للميزات مع مصنفات مختلفة (SVM، KNN، والغابة العشوائية) لتحديد 19 فئة من النباتات مع 15 مرضًا. لتحسين دقة هذه النماذج، تم استخدام التكاملات الضبابية (Sugeno وChoquet)، مع تحسين محدد للمقياس الضبابي بفضل خوارزمية تحسين سرب الجسيمات .(PSO)

الكلمات المفتاحية: تشخيص أمراض النباتات، الرؤية الحاسوبية، CNN، مصنفات، التكاملات الضبابية، تكامل Choquet، تكامل Sugeno، المقاييس الضبابية،PSO .

Abstract

Agriculture is crucial for the survival of humanity, but it faces major challenges related to plant diseases, which can lead to significant crop losses. Early detection and prevention of disease spread are essential. To overcome these challenges, many researchers are exploiting computer vision techniques to analyze images of plant leaves and identify diseases, as the majority of symptoms appear on the leaves. Our work is part of this approach by developing and evaluating four advanced models: a convolutional neural network (CNN) model and three hybrid models combining a CNN as a feature extractor with different classifiers (SVM, KNN, and Random Forest) to identify 19 plant classes with 15 diseases. To improve the accuracy of these models, fuzzy integrals (Sugeno and Choquet) were used, with a specific optimization of the fuzzy measure using the particle swarm optimization (PSO) algorithm.

Keywords: Plant Disease Diagnosis, Computer Vision, CNN, Classifiers, Fuzzy Integrals, Choquet Integral, Sugeno Integral, Fuzzy Measures, PSO.

Résumé

L'agriculture est cruciale pour la survie de l'humanité, mais elle fait face à des défis majeurs liés aux maladies des plantes, qui peuvent entraîner des pertes de récoltes significatives. La détection précoce et la prévention de la propagation des maladies sont essentielles. Pour surmonter ces défis, de nombreux chercheurs exploitent les techniques de vision par ordinateur pour analyser les images des feuilles des plantes et identifier les maladies, car la majorité des symptômes apparaissent sur les feuilles. Notre travail s'inscrit dans cette démarche en développant et en évaluant quatre modèles avancés : un modèle de réseau de neurones convolutionnel (CNN) et trois modèles hybrides combinant un CNN comme extracteur de caractéristiques avec différents classifieurs (SVM, KNN et forêt aléatoire) pour identifier 19 classes de plantes avec 15 maladies.Afin d'améliorer la précision de ces modèles, des intégrales floues (Sugeno et Choquet) ont été utilisées, avec une optimisation spécifique de la mesure floue grâce à l'algorithme d'optimisation par essaim de particules (PSO).

Mots clés : Diagnostic des maladies des plantes, Vision par ordinateur, CNN, Classifieurs, Intégrales Floues, Intégrale de Choquet, Intégrale de Sugeno, Mesures Floues, PSO.

Remerciements

Tout d'abord, je tiens à remercier ALLAH le Tout-Puissant de m'avoir donné la santé, la volonté et le courage de réaliser ce travail.

En guise de reconnaissance, je tiens à remercier très sincèrement ma promotrice, Dr. Nesrine BOUADJENEK, pour son aide précieuse, son encadrement de qualité et ses directives qui m'ont été d'une grande utilité afin d'accomplir ce travail.

J'adresse également mes remerciements aux membres du jury qui ont accepté de consacrer leur temps à examiner ce travail : Madame Rachida TOUHAMI, Professeure à l'École Nationale Polytechnique et Monsieur Sid-Ahmed BERRANI, Professeure à l'École Nationale Polytechnique.

Pour conclure, je tiens à remercier tous ceux qui m'ont aidé de près ou de loin tout au long de mon cursus académique.

Melissa.

TABLES DES MATIÈRES

LISTE DES TABLEAUX

TABLE DES FIGURES

LISTE DES ACRONYMES

- **ANN** : Artificial Neural Network
- **CNN** : Convolutional Neural Network
- **CPU** : Central Processing Unit
- **DNN** : Deep Neural Network
- **EC** : Entièrement connecté
- **GPU** : Graphic Processor Unit
- **IA** : Intelligence Artificielle
- **KNN** : K-Nearest Neighbors
- **MLP** : Multi-layer Perceptron
- **ReLU** : Rectified Linear Unit
- **SVM** : Support Vector Machine
- **PSO** : Particle Swarm Optimization
- **FM** : Fuzzy Measures
- **RF** : Random forest
- **SGD** : Stochastic Gradient Descent
- **LR** : Logistic Regression
- **LDA** : Linear Discriminant Analysis
- **NB** : Naive Bayes

INTRODUCTION GÉNÉRALE

"**L'agriculture est la base et la force de la prospérité du pays**"

Napoléon Bonaparte

L'agriculture est l'un des domaines clés dont dépend la survie de l'humanité. Avec l'augmentation de la population, les besoins alimentaires ont considérablement augmenté par rapport aux époques précédentes[1]. Cependant, elle est confrontée à un défi majeur qui est la protection contre les maladies qui peuvent affecter le rendement des cultures. Aux États-Unis seulement, il y a une perte de récolte estimée à environ 21 milliards de dollars par an [2]. Nous ne pouvons faire face à ce problème que si nous sommes capables de prévenir les maladies des plantes dès les premiers stades et de prendre des mesures préventives pour stopper la propagation des maladies entre les plantes [3].

Il y a de nombreux problèmes à le faire manuellement : les années d'expérience nécessaires, l'identification exacte de la maladie et ensuite la suggestion d'un diagnostic approprié. Nous pouvons résoudre ce problème en introduisant des techniques automatisées. La détection automatisée des maladies des plantes est un sujet de recherche qui aspire la communauté de l'intelligence artificielle et de la vision par ordinateur. Elle vise à développer des systèmes intelligents permettant une détection automatique des maladies des feuilles des plantes . Les feuilles contiennent des textures et des caractéristiques visuelles qui permettent d'identifier le type de maladie en utilisant des algorithmes d'apprentissage. Ces dernières années, les techniques d'apprentissage profond ont été largement utilisées car elles permettent de développer des systèmes de détection automatisés [4]. Cependant, les scores de détection doivent être élevés pour minimiser l'erreur. Le problème qui confronte la plupart des modèles est que lorsque le nombre de classes est très élevé, l'exactitude diminue. C'est ce problème que nous allons résoudre dans notre projet.

Ce travail décrira notre apport dans la détection des maladies des plantes. Lors de cette étude, nous avons développé quatre modèles : un modèle de réseau de neurones convolutionnel (CNN) et trois modèles hybrides combinant un CNN comme extracteur de caractéristiques avec différents classifieurs (SVM, KNN et forêt aléatoire) pour identifier 19 classes de plantes avec 15 maladies. Dans le but d'amélioration des performances de notre système, nous avons intégré des outils mathématiques comme les intégrales floues. Dans notre étude, nous avons utilisé deux types d'intégrales floues : l'intégrale de Sugeno et l'intégrale de Choquet. La particularité de ces intégrales est qu'elles disposent d'un paramètre qui influence beaucoup leurs résultats appelé la mesure floue. Dans la majorité des modèles développés par les chercheurs, ce paramètre est choisi aléatoirement. Dans notre étude, nous l'avons calculé de manière exacte

en utilisant l'algorithme d'optimisation par essaim de particules pour attribuer la valeur la plus optimale à ces mesures floues.

Ce mémoire est structuré en 5 chapitres de la manière suivante :

Chapitre 1 : Ce chapitre présente les étapes constituant un système de classification des maladies des plantes, ainsi qu'une description des travaux les plus récents effectués dans ce domaine.

Chapitre 2 : Ce chapitre fournit une étude approfondie sur les réseaux de neurones convolutifs, ainsi que sur les différents classifieurs que nous avons utilisés dans notre étude.

Chapitre 3 : Dans ce chapitre, nous détaillerons l'intégrale floue et l'optimisation par essaim de particules, ainsi que la manière dont nous les avons combinées pour travailler ensemble dans notre étude.

Chapitre 4 : Dans ce chapitre, nous parlerons plus en détail du travail effectué dans notre projet, à travers la description de la méthodologie suivie, ainsi que l'analyse des performances des différents systèmes mis en œuvre pour la classification des maladies des plantes.

Chapitre 5: Ce dernier chapitre sera consacré à la description des méthodes utilisées pour améliorer les performances des différents systèmes mis en œuvre pour la classification des maladies des plantes.

Enfin, nous terminerons ce mémoire par les principales conclusions et quelques perspectives.

CHAPTER 1

ÉTAT DE L'ART SUR LES SYSTÈMES DE CLASSIFICATION DES MALADIES DES PLANTES

1.1 Introduction

L'agriculture n'est pas seulement le pilier de l'économie nationale, mais aussi une source essentielle de subsistance pour l'humanité. Dans ce contexte, la préservation de la santé des plantes revêt une importance cruciale. La détection précoce des infections végétales est une étape vitale pour protéger les plantes saines, garantissant ainsi la préservation des rendements et la sécurité alimentaire [5]. Les feuilles des plantes constituent des indicateurs essentiels des infections, car la majorité des symptômes de maladies se manifestent principalement sur elles. L'état de santé d'une plante se révèle à travers la couleur, la texture, les bordures et la taille de ses feuilles [4]. Comme illustré dans la Figure 1.1.

En utilisant des images de feuilles pour l'identification des maladies, les progrès substantiels en matière d'apprentissage automatique et d'apprentissage profond ont ouvert des opportunités pour perfectionner la précision et l'efficacité des systèmes d'identification des maladies des feuilles de plantes. En exploitant des réseaux de neurones convolutifs (CNN) et des classifieurs d'apprentissage automatique, ces systèmes peuvent analyser des images de feuilles avec une grande précision, détectant des signes de maladies souvent imperceptibles à l'œil nu.Cette technologie offre des diagnostics rapides et précis, réduisant les ressources nécessaires pour gérer les infections des plantes et améliorant la productivité agricole[6].

Figure 1.1: Exemples des images des feuilles de l'ensemble de données PlantVillage[7]

A travers Ce chapitre nous allons présente les étapes constituant un système de classification des maladies des plantes, ainsi qu'une description des travaux les plus récents effectués dans ce domaine

1.2 Maladies des plantes

Les maladies des plantes surviennent en raison d'une anomalie dans la forme, la physiologie ou le comportement des plantes[8]. Elles peuvent être causées par des agents biotiques (pathogènes tels que les champignons, les bactéries) ou des agents abiotiques (les facteurs physiologiques tels que les coups de soleil, les carences minérales), comme montre dans la Figure 1.2.

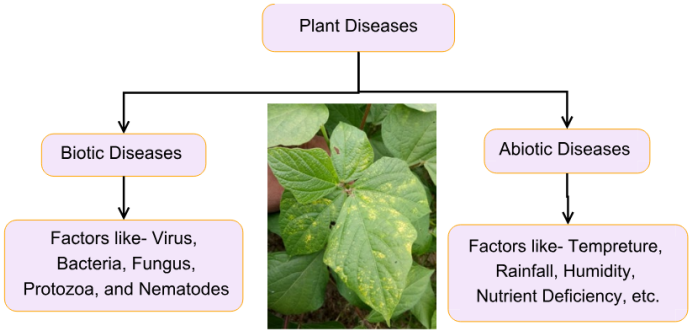

Figure 1.2: Classification des maladies des plantes et facteurs responsables[8]

Les maladies biotiques sont infectieuses. En revanche, les maladies abiotiques sont causées par la présence d'agents non infectieux dans les plantes. Par conséquent, les maladies biotiques sont les plus dangereuses et causent les plus grands dommages à nos récoltes agricoles. Ce type des maladies sont divisées en trois catégories principales, comme illustré dans la Figure 1.3.

Maladies fongiques:
Les infections fongiques sont plus graves que d'autres maladies des plantes, affectant les cultures et leur qualité tout en raccourcissant la croissance et la productivité des plantes. Les champignons phytopathogènes (parasites et semi-parasites) sont responsables des maladies fongiques des plantes. Les maladies fongiques les plus courantes et dangereuses pour les plantes sont l'oïdium, la rouille, le flétrissement fusarien, la tache noire[9] .

Maladies bactériennes :
Les infections bactériennes peuvent causer des dommages aux tiges, feuilles et racines des plantes, parfois même sans présenter de symptômes externes. Elles se propagent rapidement d'une plante à l'autre. Les maladies bactériennes, résultant de ces infections, sont principalement causées par des bactéries. Parmi les plus communes, on trouve la pourriture noire, la brûlure bactérienne, la brûlure des feuilles et la tache bactérienne. Son impact destructeur s'intensifie notamment par temps chaud et humide, ce qui en fait une menace sérieuse pour ces cultures. [9]

Maladies virales:
Contrairement aux champignons, les infections virales ne produisent ni spores ni corps fructifères. Contrairement aux maladies bactériennes, elles ne créent pas non plus d'exsudat. Ils ne peuvent même pas être cultivés en laboratoire car ils ne se développent que dans des

cellules vivantes. Les maladies virales sont causées par le pathogène virus. Le jaunissement des feuilles, la courbure des feuilles, la mouche blanche, la brûlure des bourgeons et le virus de la mosaïque sont les maladies virales les plus courantes. [9]

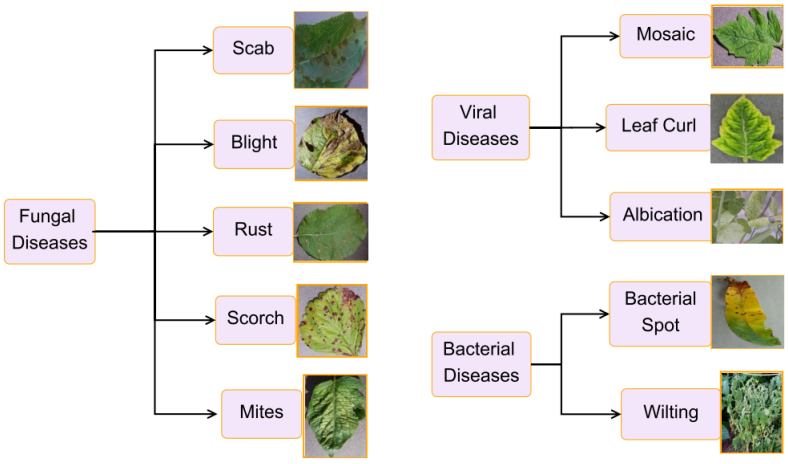

Figure 1.3: Infections biotiques des plantes dans diverses cultures[8]

1.3 Systèmes de détection des maladies des plantes

Un système de diagnostic des maladies des plantes comme montre dans la figure 1.4 est un outil technologique sophistiqué conçu pour identifier et évaluer les infections des plantes en analysant les symptômes visibles, souvent à travers des images des feuilles et d'autres parties de la plante. Selon [10], un tel système est utilisé pour prévoir la survenue ou l'aggravation des maladies chez les plantes cultivées. À l'échelle d'une parcelle, ces systèmes assistent les producteurs dans la prise de décisions stratégiques concernant les traitements phytosanitaires. Ils intègrent les conditions météorologiques actuelles et prévues pour formuler des recommandations précises sur la nécessité d'un traitement. La mission principale d'un système de diagnostic des maladies végétales est de détecter les maladies suffisamment à l'avance pour permettre aux cultivateurs de prendre des décisions éclairées, afin de minimiser les pertes économiques.

Figure 1.4: Systèmes de détection des maladies des plantes [11]

1.3.1 Objectifs des systèmes de détection des maladies des plantes

Les systèmes de détection des maladies des plantes sont développés dans le but de répondre à plusieurs objectifs spécifiques :

- Simplifier la gestion des épidémies.

- Minimiser les pertes financières engendrées par la propagation des parasites.
- Aider les agriculteurs dans leurs décisions concernant l'usage des pesticides et fongicides.
- Améliorer la qualité des récoltes.

1.3.2 Caractéristiques des système de détection des maladies des plantes

Selon [10] un système de détection des maladies efficace repose sur plusieurs caractéristiques clés comme :

- **Fiabilité :** Utilisation de données biologiques et environnementales solides.
- **Simplicité :** Plus le système est simple, plus il sera appliqué et utilisé par les producteurs.
- **Importance :** La maladie doit être économiquement importante pour la culture, mais suffisamment sporadique pour que le besoin de traitement ne soit pas systématique.
- **Utilité :** Le système de prévision doit être bénéfique, c'est-à-dire qu'il doit alléger les producteurs de nombreuses tâches de surveillance des cultures en fournissant les recommandations nécessaires pour l'application des produits chimiques et la gestion des maladies au moment opportun.
- **Disponibilité :** Les données concernant les éléments d'interaction, telles que le type de plante et les données climatiques, doivent être disponibles en temps réel.
- **Rentabilité :** Le système de prévision doit être économique en termes de coût par rapport aux techniques de gestion des maladies existantes.

l'ensemble des ces caractéristique ils sont représente dans la figure 1.5 .

Figure 1.5: Caractéristiques des système de détection des maladies des plantes [11]

1.4 Architecture des systèmes de classification des maladies des plantes.

L'architecture d'un système de classification des maladies des plantes se compose principalement de trois blocs principaux, comme montre dans la Figure 1.6 :

Bloc de pré-traitement: Ce premier bloc reçoit les images brutes et les soumet à une série de processus visant à améliorer leur qualité. Ces processus incluent le débruitage, la normalisation, la détection de contours et la segmentation des images.

Bloc d'extraction des caractéristiques : Ce bloc joue un rôle central dans la qualité globale du système. Il analyse les images prétraitées afin d'extraire les caractéristiques fondamentales utilisées dans leur classification. Ces caractéristiques sont obtenues grâce à l'application de méthodes statistiques conventionnelles ou de techniques d'apprentissage profond, assurant ainsi une représentation précise des données pour la classification ultérieure.

Bloc de classification : Ce bloc reçoit en entrée le vecteur de caractéristiques généré par le bloc précédent,Il se charge ensuite de la classification, que ce soit à travers des algorithmes classiques ou bien par de l'apprentissage, fournissant ainsi en sortie la classe de l'image d'entrée.

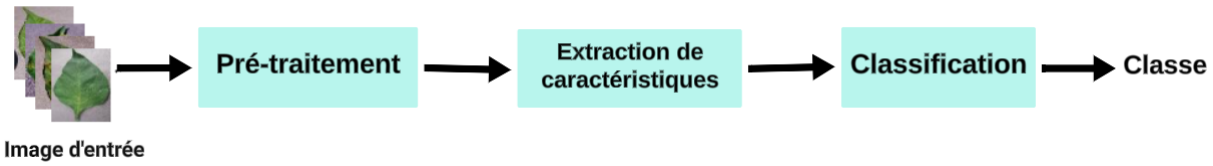

Figure 1.6: Architecture d'un système de classification des maladies des plantes

1.4.1 Pré-traitement des images

Le prétraitement des images est une étape cruciale en vision par ordinateur et en traitement d'images, car la qualité des données détermine la qualité du modèle final. Cette étape implique des opérations de bas niveau, où l'entrée et la sortie sont des images[12]. L'objectif principal du prétraitement est d'améliorer la qualité des images pour faciliter leur analyse. En supprimant les distorsions indésirables et en améliorant certaines propriétés des images, on prépare ainsi les données pour une meilleure performance du modèle.

Par exemple, à l'entrée des premières couches d'un module, il est essentiel que toutes les images soient de la même taille. Le prétraitement permet de standardiser les dimensions des images, réduisant ainsi le temps d'entraînement du modèle et accélérant l'inférence[13]. On distingue plusieurs opérations de prétraitement, incluant des opérations comme le redimensionnement, le recadrage, l'élimination de l'arrière-plan, le lissage et la normalisation comme ci illustrée dans la Figure 1.7.

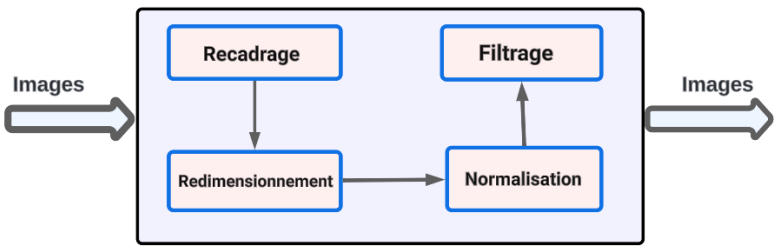

Figure 1.7: Principe de Prétraitement d'image

Dans la plus part des cas , pour effectuer le prétraitement des images de plantes, on utilise ce genre d'opérations de prétraitement.

1.4.1.1 Normalisation des images

Nous standardisons nos images afin d'optimiser les performances de la plupart des modèles d'apprentissage automatique et d'apprentissage profond. Idéalement, ces modèles fonctionnent de manière plus efficace lorsque les valeurs d'entrée sont normalisées dans une plage de -1 à 1, ce qui leur permet de mieux appréhender les nuances et les variations des données.

1.4.1.2 Débruitage des images

L'image acquise peut contenir différents types de bruit provenant de l'environnement, des dispositifs de capture d'image, qui sont éliminés par diverses techniques de suppression du bruit. La méthode du filtre moyen est utilisée pour éliminer le bruit gaussien. Le filtrage médian est une technique de filtrage numérique non linéaire, utilisée pour éliminer le bruit de sel et poivre et pour accentuer les contours. Pour la suppression du bruit et le lissage des images tout en préservant les contours, un filtre bilatéral est utilisé.[14]
La Figure 1.8 illustre les résultats obtenu après l'application d'un filtre de lissage gaussien

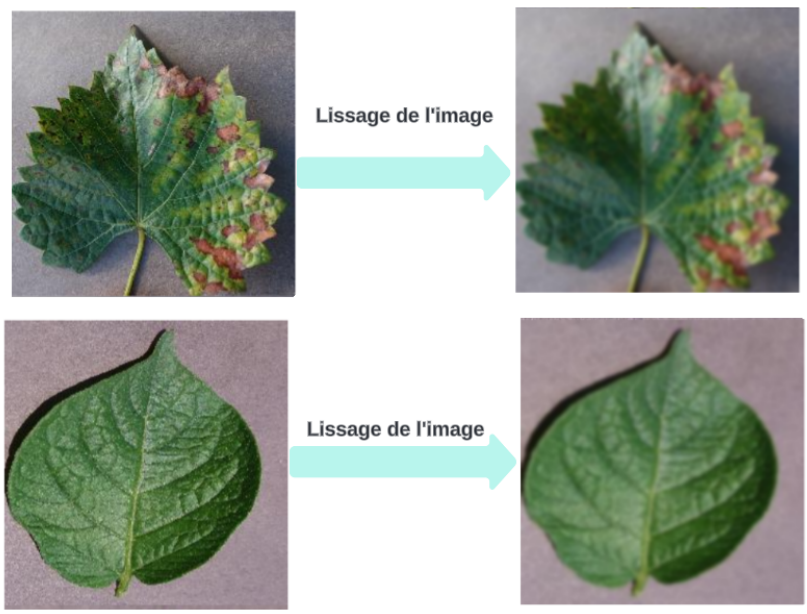

Figure 1.8: Lissage des images avec un filtre gaussien

1.4.1.3 Suppression de l'arrière-plan

La suppression de l'arrière-plan est une technique utilisée pour séparer une image en deux classes distinctes : le premier plan (feuille) et l'arrière-plan (sol). Comme ci illustré dans la figure 1.9. En éliminant l'arrière-plan indésirable, on s'assure que les caractéristiques extraites se concentrent uniquement sur la partie de l'image qui nous intéresse, à savoir la feuille[15]. Cela améliore la qualité des caractéristiques extraites en éliminant les distractions et les interférences de l'arrière-plan, ce qui permet une analyse plus précise et fiable de la feuille.

Figure 1.9: Exemples d'images après suppression de l'arrière-plan[16]

1.4.1.4 Segmentation des images

La segmentation d'image est une technique utilisée pour diviser une image en régions distinctes, dans le but de détecter des anomalies ou des zones d'intérêt spécifiques, telles que des régions malades comme ci illustré dans la Figure 1.10. Cette méthode simplifie la représentation de l'image, en permettant une analyse plus ciblée et une meilleure distinction entre les différentes zones. En impactant l'étape suivante, qui est l'extraction des caractéristiques, la segmentation contribue à améliorer la précision et la fiabilité de l'analyse, en fournissant des informations pertinentes sur les régions identifiées. Ainsi, la segmentation d'image joue un rôle crucial dans de nombreuses applications de traitement d'image et de vision par ordinateur.[8]

Figure 1.10: Images de feuilles malades avec segmentation des régions malades[16]

1.4.2 Extraction des caractéristiques

L'étape d'extraction des caractéristiques est cruciale dans la construction du modèle de classification et vise à extraire les attributs pertinents caractérisant chaque classe [17]. Dans la littérature, les attributs de couleur, de texture et de forme de la zone malade sont utilisés pour la classification des maladies des plantes. En fonction de ces attributs, diverses techniques d'extraction de caractéristiques appropriées ont été employées dans plusieurs recherches de classification des maladies des plantes.

La performance du système de classification des maladies dépend de la technique d'extraction des caractéristiques employée. Par conséquent, le choix des caractéristiques et des descripteurs de caractéristiques est un enjeu crucial dans la classification des maladies en raison de la

similarité des caractéristiques des maladies [18]. La matrice de co-occurrence des niveaux de gris (GLCM), la transformée en ondelettes, la texture de Haralick et la transformée de Gabor, l'histogramme des gradients orientés (HOG) et les techniques des motifs binaires locaux sont généralement utilisées comme descripteurs/extracteurs de caractéristiques dans diverses études [19]; [20]; [21]; [22].

La sélection de caractéristiques non pertinentes provoque des problèmes de performance de modèle telle qu'il peut entraînent un coût computationnel plus élevé et aussi peuvent causer un surapprentissage du classifieur.Ainsi, dans les applications d'apprentissage automatique et de vision par ordinateur, des techniques de sélection de caractéristiques sont employées pour trouver les caractéristiques les plus pertinentes du vecteur de caractéristiques.
À cette fin, le score de l'analyse en composantes principales (ACP), l'entropie et la covariance des caractéristiques extraites ont été utilisés dans une étude [22]. Des études ont également utilisé l'ACP, le gain d'information (IG), et l'évaluateur de caractéristiques Relief-F, les courbes floues et la surface floue, pour sélectionner les caractéristiques les plus adaptées [23];[24]; [25]. De plus, l'algorithme génétique et l'optimisation par essaim de particules améliorés avec la méthode de divergence de skew ont également été utilisés dans le but de sélectionner des caractéristiques de couleur et de texture adaptées[26].

1.4.3 Classification des maladies

La classification représente l'étape primordiale de la classification des maladies des plantes. Son rôle crucial dans ce processus dépend étroitement des étapes précédentes. En effet, cette phase utilise le vecteur de caractéristiques extrait des descriptions préalables. Dans des nombreux travaux récents, cette partie est traitée avec une diversité de classifieurs, comme nous l'explorerons dans la section de l'état de l'art. On retrouve souvent l'utilisation de classifieurs par apprentissage automatique tels que les SVM, Random Forest, KNN ou encore les réseaux de neurones. Chaque algorithme possède ses propres avantages et limites.

1.5 État de l'art

Cette section présente une analyse des recherches récentes sur l'application de l'apprentissage profond à la classification des maladies des plantes. Elle se concentre particulièrement sur les études utilisant l'ensemble de données **PlantVillage** [27], dont les résultats sont résumés dans le tableau 1.1.

Les auteurs de [28] ont proposé un système hybride pour la classification des maladies des feuilles de tomates, utilisant un ensemble de données composé de 16 012 images représentant 10 classes différentes. Le système hybride développé utilise le **(CNN)** pour l'extraction des caractéristiques et le **(SVM)** pour la classification. Leurs résultats montrent que le système hybride proposé atteint une précision de **98,25 %**, ce qui est supérieur à plusieurs autres systèmes de pointe.

Les auteurs de [29] ont utilisé l'apprentissage par transfert sur 15 modèles de CNN pré-entraînés pour classifier 10 types de maladies des feuilles de riz, en utilisant un ensemble de données de 10080 images. Les résultats ont montré que le modèle InceptionV3 se démarquait avec une précision moyenne de 99,64%, tandis que le modèle AlexNet avait une performance inférieure avec une précision moyenne de 97,35%. Les autres modèles évalués incluaient InceptionResNetV2, Xception, ResNet50, DenseNet201, EfficientNetB0, MobileNetV2, ResNet101, GoogleNet, Nas-

NetMobile, ShuffleNet, DarkNet53, SqueezeNet, et VGG16, obtenant des précisions allant de 97,28% à 99,61%.

Les auteurs de[30]ont proposé une approche hybride utilisant des modèles CNN, VGG19, ResNet15V2, DenseNet et Xception pour l'extraction des caractéristiques, tandis que le SVM a été employé pour la classification. L'ensemble de données utilisé dans cette étude comprenait 31 397 images de feuilles saines et malades de divers types de plantes, réparties en 25 classes. Les résultats ont montré que la précision obtenue variait de 82,05 % à 91,96 %, démontrant ainsi l'efficacité du modèle hybride pour améliorer la détection des maladies des plantes.

Les auteurs de l'article [31] ont élaboré un modèle pour diagnostiquer automatiquement quatre maladies spécifiques du riz en suivant une méthodologie rigoureuse. Le processus débute par le prétraitement des images, incluant leur redimensionnement et leur conversion en espaces colorimétriques RVB et HSV, suivi d'une étape de segmentation. Ensuite, les caractéristiques globales telles que les moments de Hu, Haralick et l'histogramme de couleur sont extraites et regroupées. L'analyse utilise plusieurs classificateurs, parmi lesquels la régression logistique, les arbres de décision, les forêts aléatoires, KNN, LDA, SVM et le classificateur bayésien naïf gaussien. Les résultats de cette étude démontrent que la forêt aléatoire se distingue comme le meilleur classificateur, avec une précision de 97,62 %. Une validation croisée à 10 volets est effectuée pour évaluer la performance des algorithmes, en utilisant la matrice de confusion ainsi que la précision, le rappel, le score F1 et le support comme critères d'évaluation.

Les auteurs de [32] ont mené une étude comparant l'efficacité de l'apprentissage profond (DL) et de l'apprentissage machine (ML) dans la classification des maladies des feuilles des plantes. Le jeu de données pour les maladies des feuilles d'agrumes a été utilisé pour effectuer la classification basée sur le DML. Les captures d'écran ont été prises personnellement par les auteurs à l'aide d'un appareil photo reflex numérique et ont une résolution de 7 DPI et une taille de pixel de 256x256. Il y a un total de 609 images. En ce qui concerne l'identification des maladies des plantes d'agrumes, les approches basées sur le ML (SVM, RF, SGD) et sur le DL (InceptionV3, VGG16, VGG19) sont toutes deux efficaces. Le taux de reconnaissance des maladies (CA) obtenu lors des tests est assez bon, les approches DL surpassant les méthodes ML dans les tâches de reconnaissance de maladies suivantes : RF (76,8%), SGD (86,5%), SVM (87%), VGG19 (87,4%), InceptionV3 (89%), et VGG16 (89,5%). D'après les résultats, RF a obtenu la précision la plus basse tandis que VGG16 a donné les meilleurs résultats.

D'autres travaux de recherche et expériences ont été menés pour classifier les maladies des plantes. Les performances de ces travaux seront résumées dans le tableau 1.1 :

Auteurs	Ensemble de données			Modèle	Exactitude	Année
	Type de plante	Nombre de classes	Nombre d'images			
[28]	Tomato	10	16012	Réseau CNN basé sur les images RGB comme extracteur de caractéristiques, SVM comme classifieur	98.25%	2022
[33]	Tomato, Potato, Rice, Apple, Pepper Bell, Sorghum	26	24156	CNN , AlexNet, MobileNet	84.24%, 91.19%, 97.33%	2024
[34]	Tomato	3	1800	CNN , InceptionV3	94.72%, 96.11%	2020
[35]	Rice,Potato	7 (4 **Rice** , 3 **Potato**)	**Rice** : 5932 **Potato** :1500	CNN ,SVM , KNN , Decision Tree, Random Forest	**Rice** :99.58%, 93.76%, 68.95 %, 90%,96.83%, **Potato** : 97.66%, 92%, 70%, 66.66%,85%,	2021
[29]	Rice	10	10080	15 modèles de CNN pré-entraînés (InceptionResNetV2, Xception, ResNet50, InceptionV3, DenseNet201, EfficientNetB0, MobileNetV2, ResNet101, GoogleNet,NasNetMobile , ShuffleNet , DarkNet53, SqueezeNet, VGG16, AlexNet)	99.64% ,99.61% ,99.59% , 99.59% ,99.58% ,99.51% , 99.47% ,99.47% ,99.35% , 99.27% ,99.19% ,99.18% , 98.98% ,98.21% ,97.28%	2023
[30]	Tomato, Potato, Corn(maiza), Grape, Apple	25	31397	CNN et VGG19 et ResNet15V2 et DenseNet et Xception comme extracteur de caracteristique et SVM comm classifieur	89.79%,90.71 %, 84.89%,91.96%, 82.05 %	2023
[36]	Tomato, Grape, Corn(maiza), Apple	21	10949	CNN , KNN	90.92%, 67.85%	2023
[37]	Apple, Grape, Peach, Cherry Pepper Bell, Strawberry	16	36958	CNN , Random Forest SVM, KNN	97.89%, 87.43% , 78.61% , 76.96%	2021
[38]	Potato	3	2850	CNN, Apprentissage par Transfert ResNet50, Hybride CNN(ResNet50) avec SVM	91.20%, 94.80%, 95.6%	2023
[39]	14 crop species	38	24305	AlexNet, ResNet50 , VGG16 , Inception V3 , MobileNet V2 , EfficientNet	87.1% , 89.7% ,91.5% , 93.8% ,93.5% ,97.5% ,	2024
[31]	Paddy(riz paddy)	4	560	LR , LDA , KNN , Decision Tree , RF, Gaussien NB ,SVM	94.05% , 76.79% ,81.55% , 94.05% , 97.62% , 66.07% , 96.43%	2023
[32]	Citrus	5	609	**ML :** RF, SGD, SVM **DL :** VGG-19, Inception-V3 , VGG-16	**ML :** 76.8% , 86.5 % , 87% **DL :**87.4% , 89% , 89.5%	2021

Table 1.1: Performances atteintes dans l'état de l'art de la détection des maladies des plantes

1.6 Conclusion

A travers ce chapitre, nous avons abordé les généralités sur les maladies des plantes, les systèmes de détection de ces maladies et leurs caractéristiques. Nous avons également décrit en détail les différents blocs constituant un système de classification des maladies des plantes. Un résumé de l'état de l'art a été présenté, mettant en évidence l'utilisation massive de l'apprentissage automatique et de l'apprentissage profond dans ce secteur. Pour bien maîtriser les outils que nous allons utiliser dans le développement de notre propre système de classification des maladies des plantes, le chapitre suivant présentera les réseaux de neurones convolutifs ainsi que les différents algorithmes de classification que nous utiliserons dans notre projet.

CHAPTER 2

RÉSEAUX DE NEURONES CONVOLUTIFS ET ALGORITHMES DE CLASSIFICATION

2.1 Introduction

L'agriculture moderne fait face au défi crucial de la détection précoce des maladies des plantes, essentielle pour assurer des rendements optimaux et la sécurité alimentaire. L'intelligence artificielle **(IA)** et les techniques d'apprentissage automatique offrent des solutions prometteuses pour automatiser et améliorer ce processus.

Pour explorer ces solutions, ce chapitre commence par une présentation globale de l'IA. Nous abordons ensuite les réseaux de neurones, en mettant l'accent sur les réseaux de neurones convolutifs (CNN), particulièrement efficaces pour l'analyse d'images et la reconnaissance des symptômes des maladies sur les feuilles des plantes. Enfin, nous conclurons par une étude détaillée des différents algorithmes de classification utilisés dans le développement de notre système de classification des maladies des plantes, tels que les machines à vecteurs de support (SVM), les forêts aléatoires (Random Forest), et les k-plus proches voisins (KNN).

2.2 L'intelligence artificielle

L'intelligence artificielle **(IA)** c'est un programme multidisciplinaire, qui se situe à l'interaction de plusieurs disciplines , notamment l'informatique, les mathématiques et les sciences cognitives .Cette discipline scientifique inventée en 1955 par deux mathématiciens, John MacCathy et Marvin Lee Minsky[40] dans le but de recréer un équivalent technologique à l'intelligence humaine.

Au cours des dernières années, les avancées dans le domaine de l'intelligence artificielle (IA) et de l'apprentissage automatique ont été remarquables, comme le démontre clairement la Figure 2.1. De nouvelles approches d'apprentissage, telles que les réseaux de neurones profonds, ont considérablement rehaussé les performances de nombreux systèmes d'IA. Ces réseaux de neurones profonds sont capables d'acquérir des représentations de données de plus en plus abstraites, leur permettant ainsi de résoudre des problèmes très complexes .

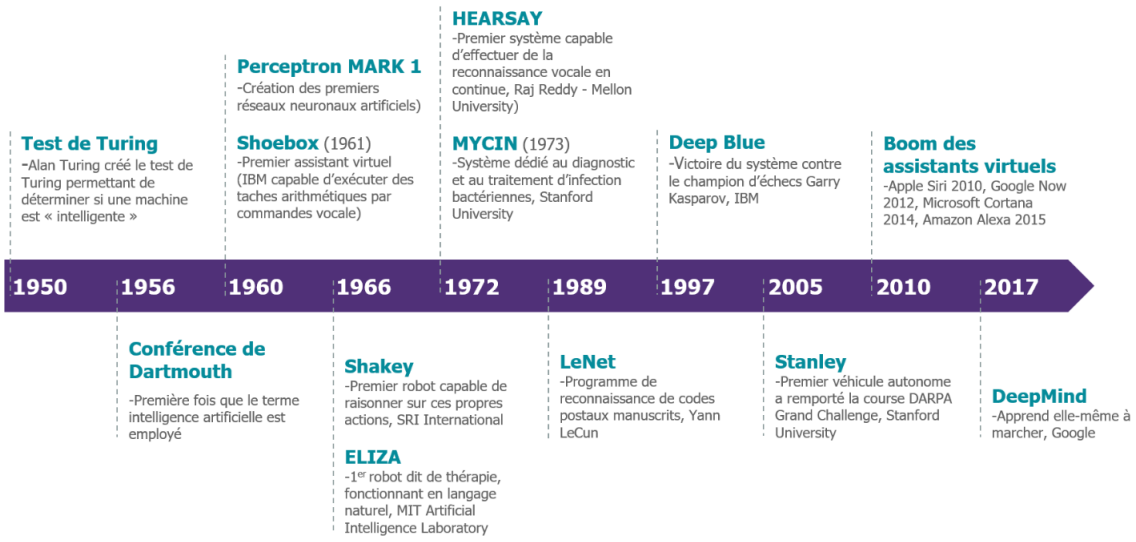

Figure 2.1: Chronologie de l'intelligence artificielle [41]

2.3 Réseaux de neurones

Pour bien comprendre le fonctionnement des réseaux de neurones artificielles , on revient à l'origine de leur histoire.Les premiers réseaux de neurones ont été inventés en 1943 par deux mathématiciens neuroscientifiques : Warren McCulloch et Walter Pitts. Il ont pu programmer des neurones artificiels en s'inspirant du finctionnement des neurones biologiques.
En biologie , les neurones sont des cellules excitables connectées les unes aux autres. Et ayant pour rôle la transmission des informations dans notre système nerveux .Chauqe neurones est composé de plusieurs dendrites , d'un corps cellulaire et d'un axone , comme ci illustré dans la Figure 2.2.Les dendrites sont les portes d'entrée d'un neurone.C'est au niveau de la synapse, que le neurone reçoit des signaux lui provenant des neurones qui le procèdent.Ces signaux peuvent être de type excitateur ou l'inverse inhibiteur. Le signal électrique produit circule le long de l'axone en direction des terminaisons pour être envoyé à son tour vers d'autres neurones de notre système nerveux.

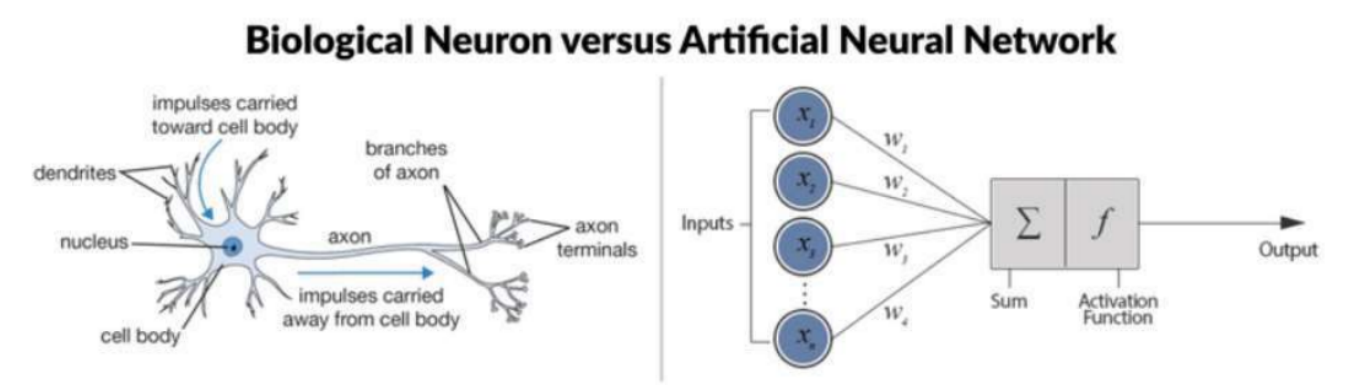

Figure 2.2: Neurone biologique versus réseau neuronal artificiel

Ainsi, les réseaux de neurones sont des modèles d'apprentissage automatique inspirés du fonctionnement du cerveau humain .Ils sont capables d'apprendre à partir de données en ajustant

les poids des connexions entre les neurones , ce qui leur permet de modéliser des relations complexes entre les entrées et les sorties .ils sont largement utilisés dans diverses applications telles que la vision par ordinateur comme le cas de notre prjet .

2.3.1 Perceptron

En 1957 ,le psychologue américain Franck ROSENBLATT a developpé le premier algorithme d'apprentissage de l'histoire de Deep Learning : **Le Perceptron**
Le perceptron est le réseau de neurones le plus simple et le plus ancien, développé pour la classification binaire. Composé de plusieurs entrées, d'une unité de traitement et d'une sortie, comme c'est illustrée dans La figure 2.3 .

Les entrées $\mathbf{x_i}$ **(de 1 à** P**)** jouent le rôle des dendrites et apportent un signal.
Chaque entrée est multipliée par un poids $\mathbf{w_i}$, remplaçant les synapses du neurone biologique.
La somme de ces entrées, multipliées par leurs poids respectifs, est injectée dans le noyau, dont le résultat est calculé à l'aide de la fonction d'activation **h**.[42]
La sortie y, se calcule comme une combinaison linéaire des poids et des entrées, plus un biais $\mathbf{w_0}$:

$$y = h\left(w_0 + \sum_{i=1}^{p} w_i x_i\right) \tag{2.1}$$

Cette fonction d'activation **h** permet de prendre **une décision** , telle que si i la combinaison linéaire était supérieure ou égale à 0, alors la sortie **y** valait 1, sinon la sortie **y** valait 0 .[42]

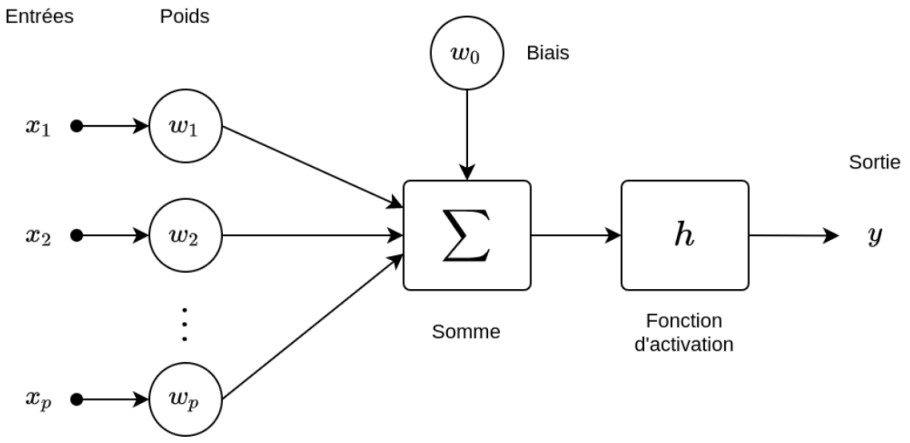

Figure 2.3: Architecture d'un Perceptron[42]

Le modèle de ce perpetron, c'est qu'il ne sait faire que **des séparations linéaires** : il n'est pas capable de construire des courbes pour séparer des points dans l'espace. C'est pour cela que nous avons besoin de créer des couches de perceptrons.

2.3.2 Perceptron multicouches (MLP)

Le perceptron multicouche a été développé par Geoffrey HNTON. Le perceptron multicouche souvent abrégé en **MLP**, est une extension du perceptron de base, comprenant au minimum trois couches de neurones comme ci illustré dans le Figure 2.4 : une couche d'entrée qui reçoit les données à traiter, une ou plusieurs couches cachées (également appelées couches intermédiaires)

qui effectuent des transformations non linéaires des entrées, et une couche de sortie qui fournit la réponse du modèle. Le MLP peut accomplir des tâches de classification et de régression non linéaires grâce à une fonction d'activation non linéaire appliquée à chaque couche.

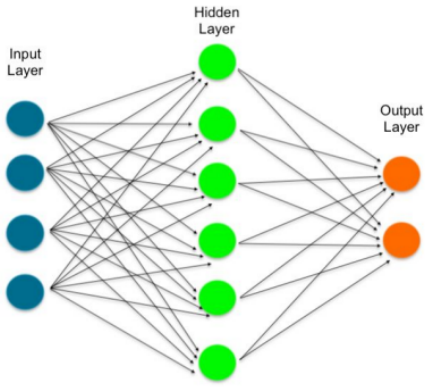

Figure 2.4: Réseau de neurones à une couche cachée[43]

2.4 Réseaux de neurones convolutifs

Un réseau neuronal convolutionnel(CNN ou ConvNet) est une catégorie de réseaux neuronaux spécialisée dans le traitement de données ayant une topologie en grille, comme une image.CNN sont utilisés pour extraire automatiquement des caracteristiques pertinentes à partir des images d'entrée, lesquelles sont ensuite transmises aux couches entiérement connectées qui agissant comme un classificateur comme ci illustré dans la Figure 2.5.
Les CNN ont gagné en popularité après les performances exceptionnelles d'AlexNet en 2012[44], mais leur développement a en fait débuté bien plus tôt. Les fondations des réseaux neuronaux convolutionnels remontent à la découverte de Hubel et Wiesel en 1959[45], qui ont montré que les cellules du cortex visuel des animaux reconnaissent la lumière dans de petits champs récepteurs. Inspiré par ce travail, Kunihiko Fukushima a proposé en 1980 le néocognitron[46], considéré comme le premier modèle théorique pour les CNN. Plus tard, en 1990, Yann LeCun et al ont développé le cadre moderne des CNN avec LeNet-5, utilisé pour reconnaître les chiffres manuscrits. Le succès d'AlexNet en 2012, avec ses améliorations significatives, a ouvert la voie à l'invention de différents modèles CNN[47] et à leur application dans divers domaines.

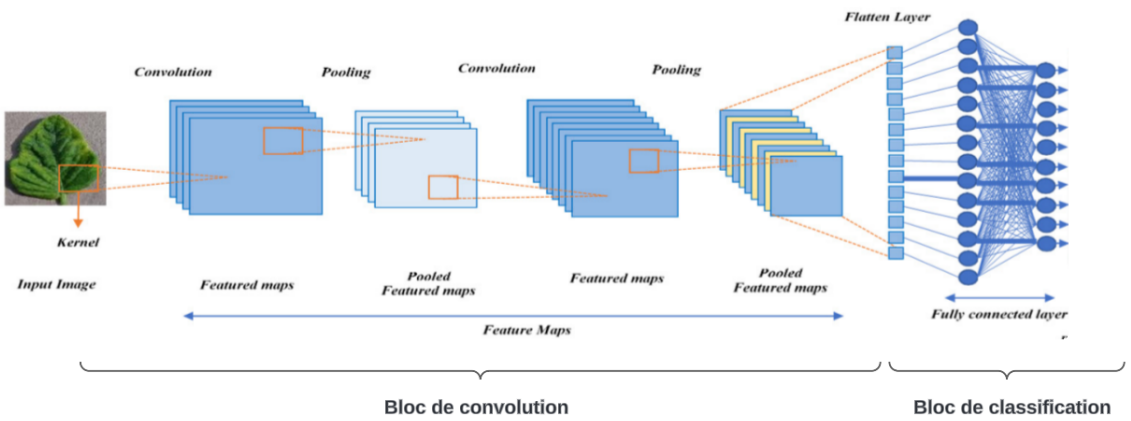

Figure 2.5: Architecture standard d'un CNN

2.4.1 Bloc des couches d'extraction des caractéristiques

Ce premier bloc distingue ce réseau en agissant comme un extracteur de caractéristiques. La première couche de ce bloc applique plusieurs noyaux de convolution à l'image, générant des caractéristiques qui sont ensuite normalisées ou redimensionnées via une fonction d'activation, formant une carte de caractéristiques. Ce processus peut être répété plusieurs fois. Les cartes de caractéristiques obtenues sont filtrées par de nouveaux noyaux à chaque étape, produisant ainsi de nouvelles caractéristiques à normaliser et filtrer de nouveau, et ainsi de suite. Enfin, les valeurs des dernières caractéristiques sont concaténées et aplaties en un vecteur colonne. Ce vecteur constitue la sortie du premier bloc et l'entrée du second.[48]
Ce Bloc du cnn comme c'est illustrée dans la Figure 2.6,il se compose principalement de deux types de couches : les couches convolutionnelles et les couches de Pooling.

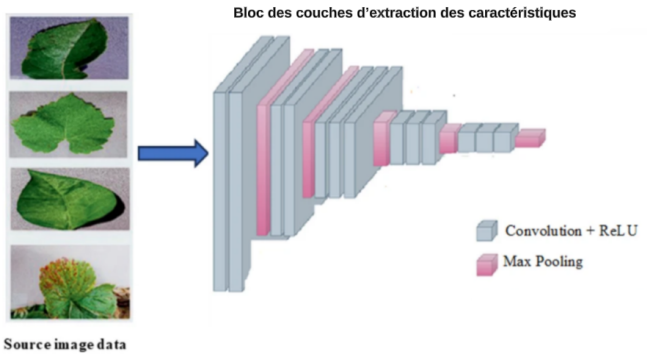

Figure 2.6: Bloc des couches d'extraction des caractéristiques[48]

2.4.1.1 Couches convolutionnelles

Les couches de convolution, en tant que première étape du traitement d'image dans un réseau de neurones, apprennent à extraire les caractéristiques de l'image d'entrée en utilisant des filtres. Chaque couche de convolution prend deux entrées : une matrice d'image de dimension h × w × d (où h est la hauteur de la matrice d'image, w est sa largeur et d est le nombre de canaux) et un filtre ou noyau de dimension fh × fw × d (où fh est la hauteur du filtre, fw est sa largeur et d est le nombre de canaux). Typiquement, l'entrée de la couche convolutive est une image couleur RGB, représentée par 3 canaux. Ces filtres analysent l'image en glissant sur celle-ci, créant des cartes de caractéristiques qui représentent les éléments significatifs de l'image. Au fur et à mesure que l'on empile plusieurs de ces couches, le réseau peut progressivement extraire des caractéristiques de plus haut niveau. Les résultats de la convolution sont ensuite passés à une fonction d'activation non linéaire pour obtenir les sorties finales de la couche. [49].
Le filtre se déplace sur l'image de gauche à droite et de haut en bas, avec un nombre défini de pas, appelés strides, pour effectuer le même calcul. Cela est illustré dans la Figure 2.7

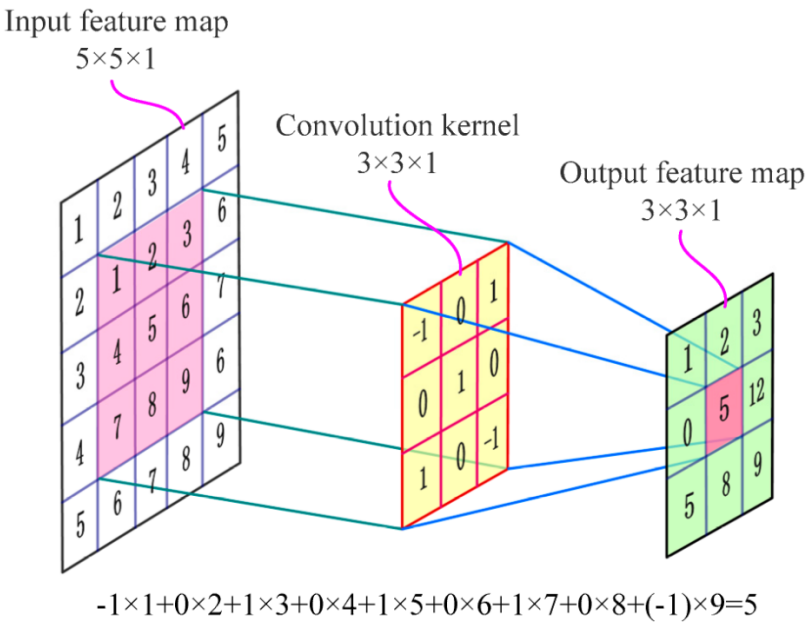

Figure 2.7: Couche de convolution

2.4.1.2 Couches de pooling

Les couches de Pooling interviennent entre les couches de convolution consécutives dans une architecture CNN. Leur fonction est de réduire la taille des cartes de caractéristiques, ce qui prévient le sur-apprentissage et améliore la vitesse de traitement. Le Pooling maintient l'invariance par translation et rotation de la convolution. Les méthodes courantes de Pooling incluent le **Max Pooling**, qui sélectionne la valeur maximale dans un champ récepteur local, et le **average Pooling**, qui calcule la moyenne des valeurs dans ce champ. Ces couches permettent de réduire la complexité du modèle en comprimant les informations non essentielles, tout en améliorant la robustesse et la vitesse de traitement. En pratique, le max-Pooling est souvent préféré au average-Pooling pour ses meilleurs résultats. [49]
La Figure 2.8 illustre un exemple des opérations de Max Pooling et d'Average Pooling.

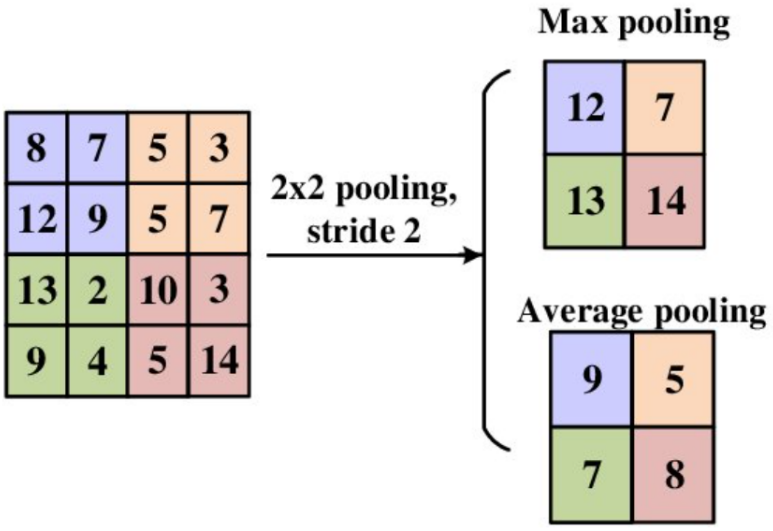

Figure 2.8: Exemple de l'Opération de Pooling (avec Max Pooling et Average Pooling)

2.4.1.3 Couche d'aplatissement : Flatten layer

La couche d'aplatissement conclut le bloc convolutif en finalisant le processus d'extraction des caractéristiques. Elle convertit les cartes caractéristiques en un vecteur unidimensionnel, comme illustré dans La Figure 2.9. Ce vecteur est ensuite envoyé au bloc de classification pour traitement par le réseau neuronal.

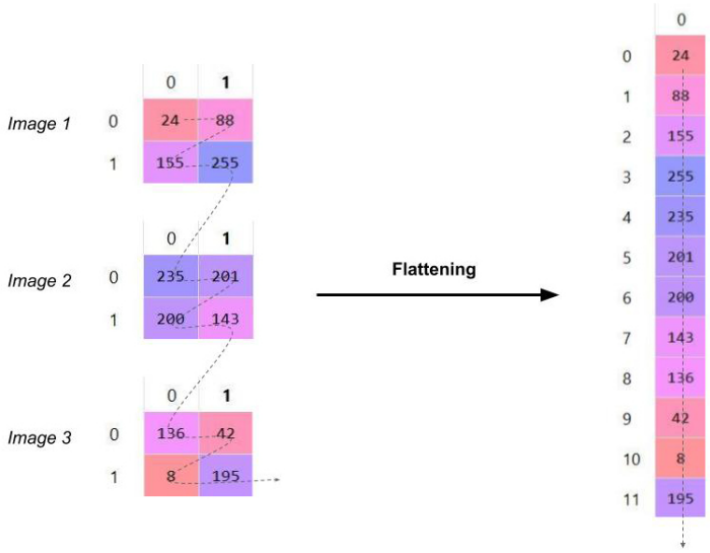

Figure 2.9: Mécanisme d'aplatissement

2.4.2 Bloc des couches de classification

Le bloc des couches de classification d'un CNN utilise des couches entièrement connectées (EC) pour classifier les images. En partant d'un vecteur aplati, les couches EC apprennent les paramètres nécessaires pour attribuer une image à une classe. Plusieurs couches EC sont empilées pour améliorer l'apprentissage et la prise de décision. La dernière couche EC produit un vecteur dont la taille correspond au nombre de classes, avec des neurones utilisant la fonction d'activation **softmax** pour fournir les probabilités d'appartenance à chaque classe comme ci illustré dans la Figure 2.10. Ce bloc assure la section de classification du CNN.[50]

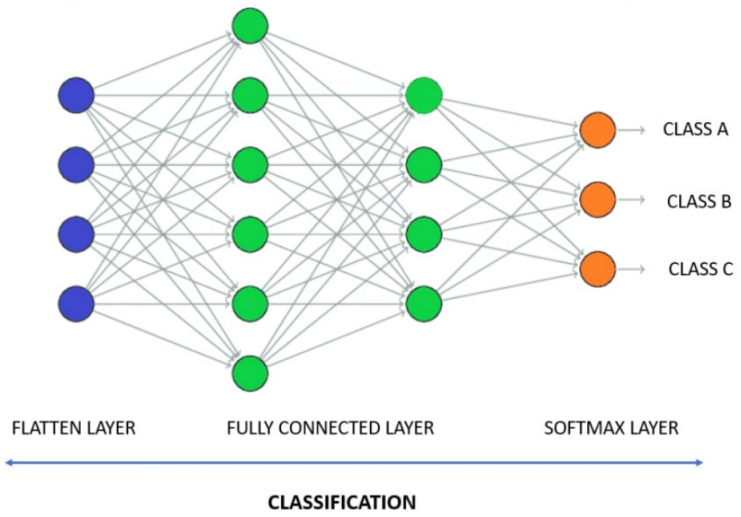

Figure 2.10: Bloc des couches de classification dans un CNN[50]

2.4.3 Paramètres d'un CNN

En complément des paramètres du MLP, dont le taux d'apprentissage, le nombre d'époques, ainsi que le nombre de couches cachées et de neurones, plusieurs autres paramètres sont requis pour superviser le bloc de convolution. Cette section examine brièvement ces paramètres.

2.4.3.1 Filtres

La technique de convolution implique de déplacer une série de filtres ou de noyaux de convolution sur l'image d'origine. La sélection des filtres, de leur taille et de leur nombre (également appelé profondeur) est une démarche expérimentale. La **"profondeur de la couche de convolution"** fait référence au nombre de noyaux utilisés, ce qui justifie le terme **"Deep Learning"**. En utilisant un nombre q de filtres, l'image subira q convolutions. Ce processus génère q matrices filtrées connues sous le nom de **"cartes de caractéristiques"**. Un exemple de cette opération est présenté dans la Figure 2.11. Dans la plupart des architectures de réseaux de neurones convolutifs prédéfinies, il existe généralement une sélection par défaut spécifique à l'application pour laquelle le modèle a été développé. [51]

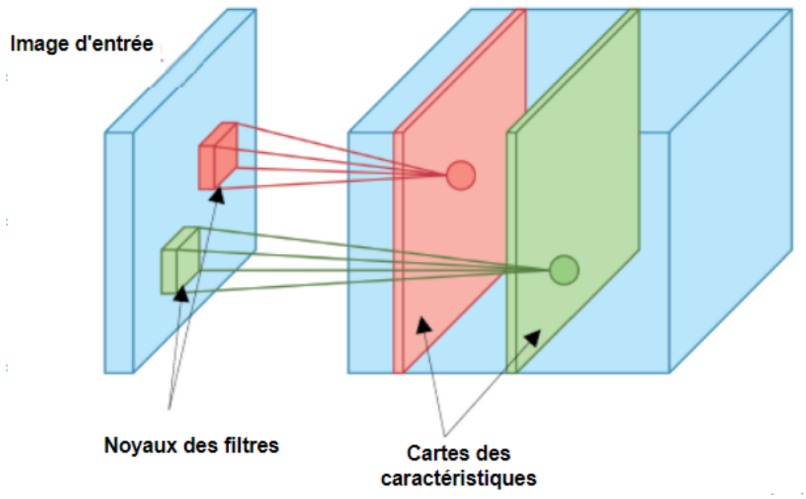

Figure 2.11: Cartes de caractéristiques

2.4.3.2 Fonctions d'activation

Ce sont des fonctions mathématiques spécifiques appliquées à la sortie du filtre dans un réseau de neurones.Elle permet au réseau de modéliser des relations complexes et non linéaires entre les données, facilitant ainsi des prises de décision plus précises et efficaces.[52] Voici quelques types de fonctions d'activation utilisées dans les réseaux de neurones :

La fonction Rectified Linear Unit (ReLU) : c'est la fonction seuil qui transforme les entrées en 0 si elles sont négatives, sinon elle les laisse inchangées comme ci illustré dans la Figure 2.12.Elle est largement utilisée dans les réseaux de neurones profonds pour sa simplicité et sa capacité à accélérer l'entraînement.
Elle est décrite par l'équation suivante :

$$G(E) = max(0, E) = \begin{cases} E & si \quad E \geq 0 \\ 0 & sinon \end{cases} \tag{2.2}$$

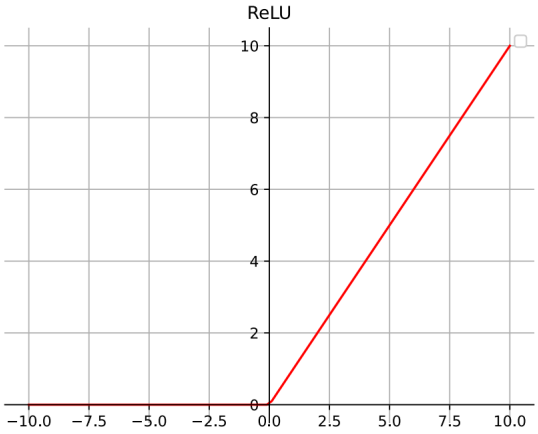

Figure 2.12: Fonction d'activation ReLU

La fonction sigmoïde : Elle représente la fonction de répartition de la loi logistique.Cett fonction transforme les entrées en une plage de valeurs entre 0 et 1 comme ci illustré dans la Figure 2.13.Elle est couramment utilisée dans les réseaux de neurones pour la classification binaire. Elle est décrite par la fonction suivante :

$$\alpha(x) = \frac{1}{1+e^{-x}} \quad \forall x \in \mathbb{R} \tag{2.3}$$

La fonction tangente hyperbolique (tanh) : La fonction tangente hyperbolique est similaire à la fonction sigmoïde mais elle transforme les entrées en une plage de valeurs entre -1 et 1 ,comme ci illustré dans la Figure 2.13.Elle est défini par l'équation suivante :

$$tanh(x) = \frac{sinh(x)}{cosh(x)} = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{2.4}$$

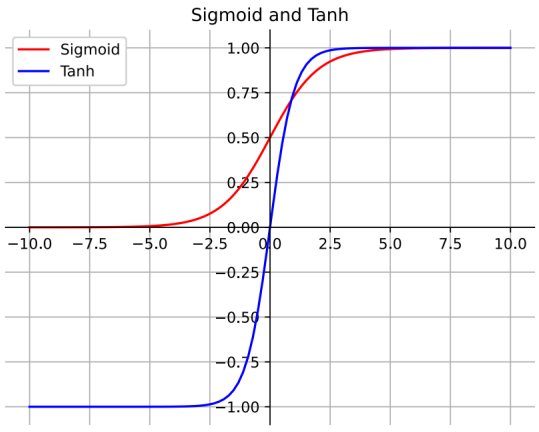

Figure 2.13: Fonction d'activation tanh et sigmoïde

La fonction softmax : c'est la fonction qui transforme les entrées en une distribution de probabilité comme ci illustré dans la Figure 2.14. Elle est souvent utilisée pour la classification multiclasse dans les réseaux de neurones.
Elle est défini par l'équation suivante :

$$G(e_j) = \frac{e^{e_j}}{\sum_i e^{e_i}} \tag{2.5}$$

Où e_j est l'élément considéré j du vecteur d'entrée. La classe ayant la plus grande probabilité sera prise comme sortie du réseau de neurones.

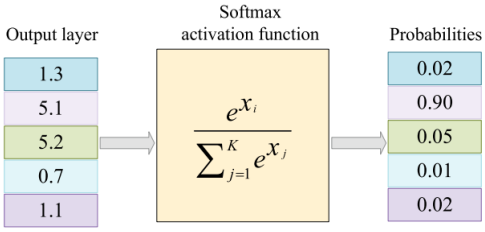

Figure 2.14: Schéma illustratif du principe de la fonction Softmax[52]

La figure 2.15 représente le graphe de la fonction d'activation softmax.

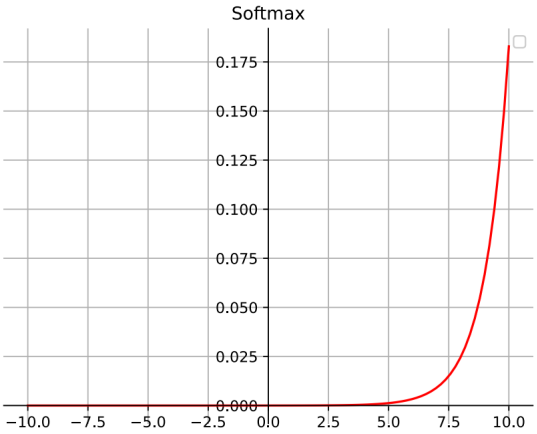

Figure 2.15: Fonction d'activation Softmax

2.4.3.3 Stride

Le Stride [53]correspond au nombre de décalages effectués lors du processus de convolution dans la matrice d'image. Si la valeur est de 1, alors les filtres de convolution se déplacent d'un pixel. Si la valeur est de 2, alors les filtres de convolution se déplacent de 2 pixels, et ainsi de suite, comme présenté dans la Figure 2.16 .

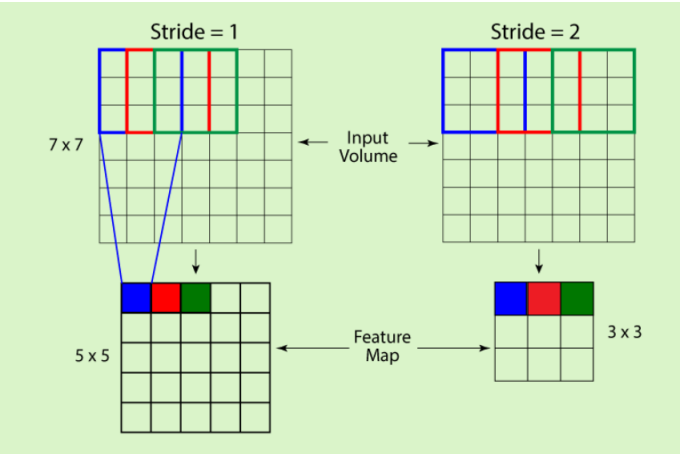

Figure 2.16: Principe du stride

2.4.3.4 Zero-padding

Après chaque opération de convolution, la dimension de l'image résultante est réduite par rapport à celle de l'image initiale, car les pixels de bordure ne peuvent pas être pleinement traités par le filtre. L'utilisation de plusieurs couches convolutionnelles entraîne une diminution significative de la taille de l'image de sortie par rapport à celle de l'image d'entrée. Afin de préserver la taille originale de l'image d'entrée, la technique du zero-padding consiste à ajouter des zéros autour des bords de l'image, en tenant compte de la taille du filtre. Ainsi, quelle que soit la profondeur des couches convolutionnelles, l'image de sortie conserve la même dimension que l'image d'entrée.La Figure 2.17 illustre un exemple de zero-padding

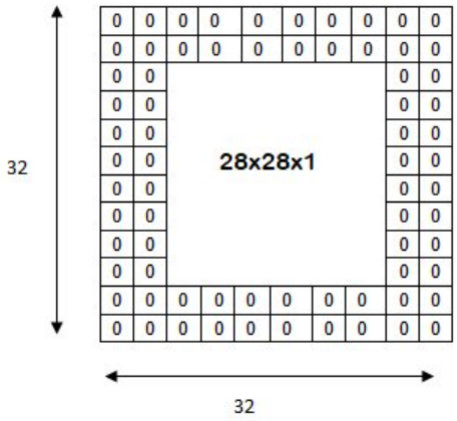

Figure 2.17: Principe du zero-padding

2.4.3.5 Drop-out

Le Dropout est une technique de régularisation utilisée lors de la phase d'apprentissage des réseaux de neurones pour réduire l'erreur de test et éviter le sur-apprentissage lié aux couches entièrement connectées. Il fonctionne en désactivant aléatoirement certains neurones pendant l'entraînement, créant ainsi des sous-réseaux simplifiés. Cette méthode permet au réseau de mieux répartir les informations et d'apprendre plus efficacement. En phase de test, tous les neurones sont activés avec des poids ajustés pour compenser les désactivations effectuées lors de l'entraînement[52].La Figure 2.18 illustre un exemple de la technique de dropout.

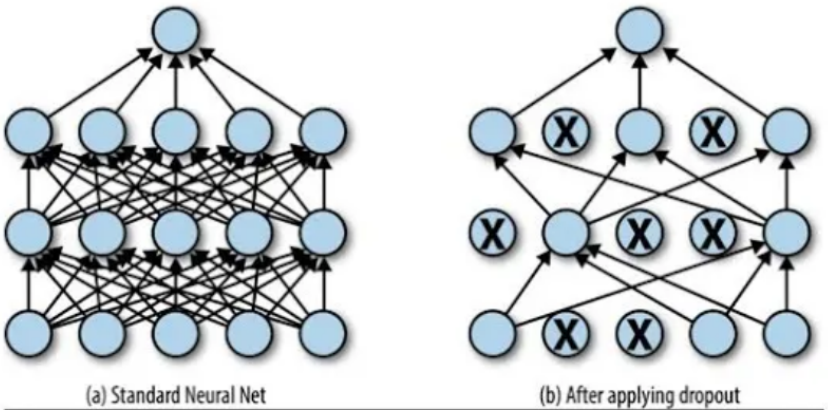

Figure 2.18: Matrice de confusion

2.5 Différents types de classifieurs utilisés

Dans le domaine de l'apprentissage automatique, il existe une variété de classifieurs adaptés à différentes tâches et types de données. Dans cette section, nous allons explorer les classifieurs populaires que nous avons utilisés dans notre étude de classification des maladies des plantes, en examinant en détail les classifieurs SVM, KNN et Random Forest.

2.5.1 Machine à vecteurs de supports (SVM)

Les machines à vecteurs de support (SVM) sont des modèles d'apprentissage supervisé utilisés pour la classification et la régression. Développées par Cortes et Vapnik en 1995, le principe des SVM consiste à séparer les données en plusieurs classes en trouvant un hyperplan qui maximise la distance entre la frontière de séparation et les points de données. Les SVM ont été rapidement adoptées en raison de leur capacité à travailler avec des données de grandes dimensions, de leurs garanties théoriques et de leurs bons résultats en pratique.Requérant un faible nombre de paramètres.[54]

Les performances et la capacité de généralisation des SVM peuvent être fortement influencées par les paramètres utilisés. Voici quelques paramètres clés à considérer :

Marge maximale : SVM vise à identifier l'hyperplan présentant la plus grande marge, définie comme la distance entre cet hyperplan et les points de données les plus proches de chaque classe. Cette marge favorise une meilleure généralisation du modèle et diminue le risque de surajustement.[55]

Vecteurs de support : Les vecteurs de support sont les points de données qui se trouvent le plus près de l'hyperplan et influencent de manière significative la position et l'orientation de l'hyperplan. Ce sont les points critiques qui déterminent la marge optimale et la frontière de décision. Les SVM sont particulièrement efficaces car ils ne se basent que sur ces vecteurs de support pour faire des prédictions, ce qui les rend robustes aux valeurs aberrantes.[55]

Fonctions de noyau : Les SVM peuvent gérer à la fois les relations linéaires et non linéaires entre les caractéristiques d'entrée et les étiquettes de sortie grâce à l'utilisation de fonctions de noyau. Les fonctions de noyau transforment les données d'entrée en un espace de dimension supérieure, ce qui facilite la recherche d'un hyperplan séparant les classes.[55]

- **Noyau linéaire** : adapté aux données linéairement séparables.
- **Noyau polynomial** :utile pour capturer les relations non linéaires.
- **Noyau à fonction de base radiale (RBF)** : adapté aux relations complexes et non linéaires. C'est l'un des noyaux les plus couramment utilisés.

Paramètre de régularisation : SVM comprennent un paramètre de régularisation, souvent noté C, qui est également appelé contrainte de la boîte.Il opère de sorte à maximiser la marge de l'hyperplan et à minimiser l'erreur de classification.Une valeur faible de C permet d'obtenir une marge plus large entre les classes, mais peut conduire à davantage d'erreurs de classification sur les données d'entraînement. À l'inverse, une valeur élevée de C vise à classifier correctement tous les exemples d'entraînement, même au prix d'une marge plus étroite entre les classes. En ajustant la valeur de C, on peut contrôler le compromis entre la complexité du modèle et sa capacité à généraliser sur de nouvelles données.[54]

Influence de Gamma : Le paramètre gamma contrôle l'influence d'un seul exemple d'entraînement sur la construction de la frontière de décision dans un modèle de machine learning. Une valeur faible de gamma indique que la frontière de décision est principalement influencée par les points de données les plus éloignés, ce qui tend à produire une frontière plus lisse et moins complexe. En revanche, une valeur élevée de gamma donne plus de poids aux points de données proches, entraînant une frontière de décision plus complexe et flexible, ce qui peut potentiellement conduire à un surapprentissage si elle est trop élevée.[54]

Forme de la fonction de décision : est un paramètre utilisé dans les SVM. Il détermine la manière dont la fonction de décision est calculée lorsqu'il y a plus de deux classes à prédire.On distingue deux configurations

- **OvO (One-vs-One)** :Dans cette configuration, le classifieur SVM crée un classifieur binaire pour chaque paire de classes possibles. Par exemple, si vous avez trois classes (A, B, C), cela générera des classifieurs pour les paires (A contre B), (A contre C) et (B contre C). Ensuite, lors de la prédiction, chaque classifieur "vote" pour la classe qu'il prédit, et la classe avec le plus de votes est choisie.
- **OvR (One-vs-Rest)** :Dans cette configuration, un classifieur est créé pour chaque classe, qui est entraîné à distinguer cette classe de toutes les autres classes. Par exemple, dans un problème à trois classes (A, B, C), vous auriez un classifieur pour prédire A contre non-A, un autre pour prédire B contre non-B, et ainsi de suite. Lors de la prédiction, chaque classifieur prédit simplement s'il pense que l'instance appartient à sa classe ou non, et la classe avec le score le plus élevé est choisie comme prédiction finale.

Le choix entre **OvO** et **OvR** peut affecter les performances de classification dans différents scénarios. **ovo** est généralement plus coûteux en termes de temps de calcul car il nécessite l'entraînement et la prédiction de plusieurs classifieurs binaires, mais peut être plus précis dans certains cas, notamment lorsque les classes sont déséquilibrées. **ovr**, en revanche, est plus simple et plus rapide, mais peut être moins précis dans certaines situations. Le choix dépend souvent de la nature du problème et des données.

La figure 2.19.montre un schéma explicatif des machines à vecteurs de support (SVM)

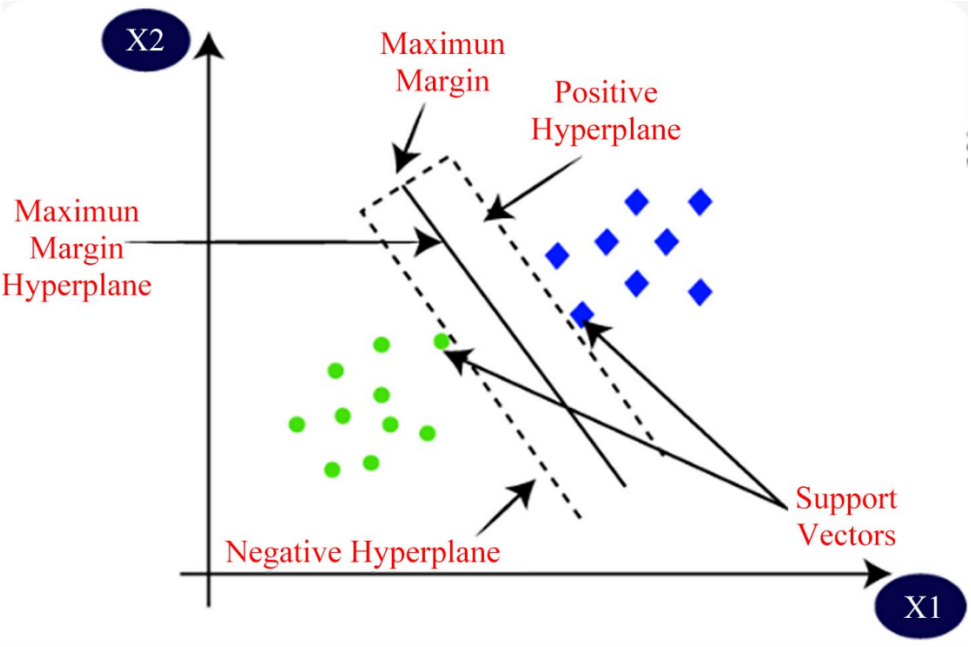

Figure 2.19: L'algorithme des machines à vecteurs de support (SVM) en 2-dimensions[54].

2.5.2 K-plus proches voisins (KNN)

L'algorithme des K plus proches voisins est un classifieur d'apprentissage supervisé non paramétrique, ce qui signifie qu'il ne fait aucune hypothèse sous-jacente sur la distribution des données[56]. Le principe de fonctionnement de KNN est le suivant : tout d'abord, il utilise l'ensemble des données d'entraînement. Ensuite, un nombre K de voisins sont sélectionnés. L'algorithme calcule ensuite les distances entre la nouvelle donnée (de test) et les données d'entraînement disponibles. Il sélectionne ensuite les K plus proches voisins et effectue un vote. La classe la plus représentée parmi les K voisins sera la classe attribuée. Cela est clairement illustré dans l'exemple présenté dans la Figure 2.20. Dans cet exemple, l'élément en rouge représente l'élément que l'on souhaite classer. Lorsque K = 3, la classe majoritaire est la classe B, tandis que lorsque K = 6, la classe majoritaire est la classe A. Cela implique que le choix du paramètre K est crucial.

Ce classifieur repose principalement sur deux paramètres :

k (nombre de voisins les plus proches) : Le nombre de voisins les plus proches à considérer pour la classification d'une nouvelle observation. Une valeur plus élevée de k peut rendre les prédictions plus stables mais peut inclure des voisins moins pertinents.[56]

Mesures de distance : Afin de déterminer quels points de données sont les plus proches d'un point de requête donné, il vous faudra calculer la distance entre le point de requête et les autres points de données. Ces mesures de distance aident à former des limites de décision, qui partitionnent les points de requête en différentes régions [57]
On distingue plusieurs méthode de calcul de la distance :

- **Distance Euclidienne :** C'est la mesure de distance la plus utilisée. Elle se calcule en prenant la racine carrée de la somme des carrés des différences entre les coordonnées de deux points.[56]

$$D_e(x, y) = \sqrt{\sum_{j=1}^{n}(x_j - y_j)^2} \tag{2.6}$$

- **Distance de Manhattan :** On l'obtient en calculant la somme des valeurs absolues des différences entre les coordonnées de deux points. [56]

$$D_{m_h}(x, y) = \sum_{i=1}^{k} |x_i - y_i| \tag{2.7}$$

- **Distance de Hamming :**la distance entre deux points données est la différence maximale entre leurs coordonnées sur une dimension. [56]

$$D_h(x, y) = \sum_{i=1}^{k} |x_i - y_i| \tag{2.8}$$

où :

$$\begin{cases} x = y & \Longrightarrow D = 0 \\ x \neq y & \Longrightarrow D = 1 \end{cases}$$

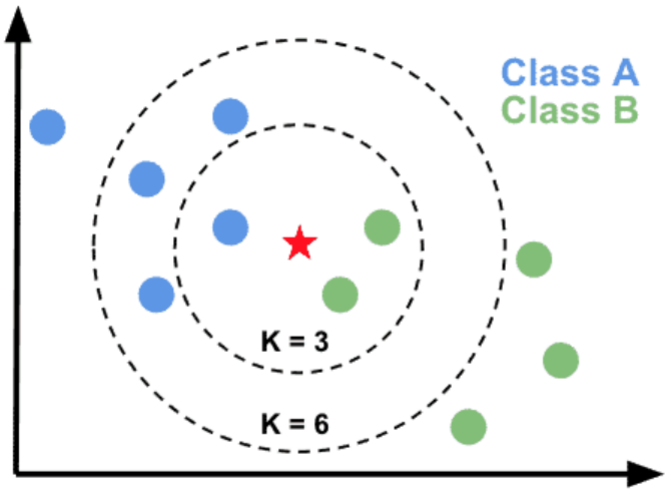

Figure 2.20: Principe du KNN.

2.5.3 Forêt aléatoire (Random Forest)

algorithme de la forêt aléatoire est une technique d'apprentissage par arbres puissante en apprentissage automatique. Il fonctionne en créant un certain nombre d'arbres de décision pendant la phase d'entraînement. Chaque arbre est construit en utilisant un sous-ensemble aléatoire du jeu de données et en mesurant un sous-ensemble aléatoire de caractéristiques à chaque partition. La sortie de la forêt aléatoire est la classe sélectionnée par la plupart des arbres,comme elle est illustrée dans la Figure 2.21. Ce processus décisionnel collaboratif, soutenu par plusieurs arbres avec leurs perspectives, fournit des résultats stables et précis.
Les forêts aléatoires sont largement utilisées pour les tâches de classification, connues pour leur capacité à gérer des données complexes, à réduire le surapprentissage et à fournir des prévisions fiables dans différents environnements[58].

Les performances et la capacité de généralisation de la forêt aléatoire sont fortement influencées par les paramètres utilisés, tels que la définition des critères d'arrêt pour limiter la croissance d'un arbre avant qu'il n'atteigne un nombre excessif de niveaux, afin d'éviter le surapprentissage. Parmi les paramètres clés à considérer, on trouve :

La profondeur maximale des arbres : Ce paramètre détermine la profondeur maximale de chaque arbre dans la forêt. Une profondeur plus importante peut entraîner un surajustement (overfitting). La plupart du temps, ce paramètre est laissé à sa valeur par défaut (None), permettant ainsi aux arbres de se développer jusqu'à ce que toutes les feuilles soient atteintes.

Nombre minimum d'échantillons par feuille: C'est le paramètre qui spécifie le nombre minimum d'échantillons requis pour qu'un noeud de l'arbre soit considérè comme une feuille .Un nombre plus élevé peut éviter que les arbres s'adaptent trop précisément aux données d'entraînement (surajustement).

Nombre minimum d'échantillons requis pour effectuer une division : Ce paramètre spécifie le nombre minimum d'échantillons nécessaires pour diviser un nœud interne de l'arbre.Un nombre plus élevé peut aider à éviter le surajustement en forçant les arbres à être plus généralisés.

Le critère de division : désigne la fonction utilisée pour mesurer la qualité d'une division à chaque nœud de l'arbre de décision. Les critères couramment utilisés sont l'indice de Gini et l'entropie. Le choix du critère influence le développement des arbres, affectant ainsi la performance globale de la forêt. Une bonne sélection du critère peut améliorer la précision et l'efficacité du modèle, en adaptant la façon dont les arbres se scindent et se construisent en fonction des données.[54]

Nombre d'arbres : c'est le nombre d'arbres de décision dans la forêt aléatoire. Un plus grand nombre d'arbres réduit la variance du modèle, améliorant ainsi la stabilité et la précision des prédictions. Toutefois, cela augmente le temps de calcul et les ressources nécessaires. Il est donc essentiel de trouver un équilibre optimal pour maximiser les performances de généralisation tout en gérant efficacement les ressources.

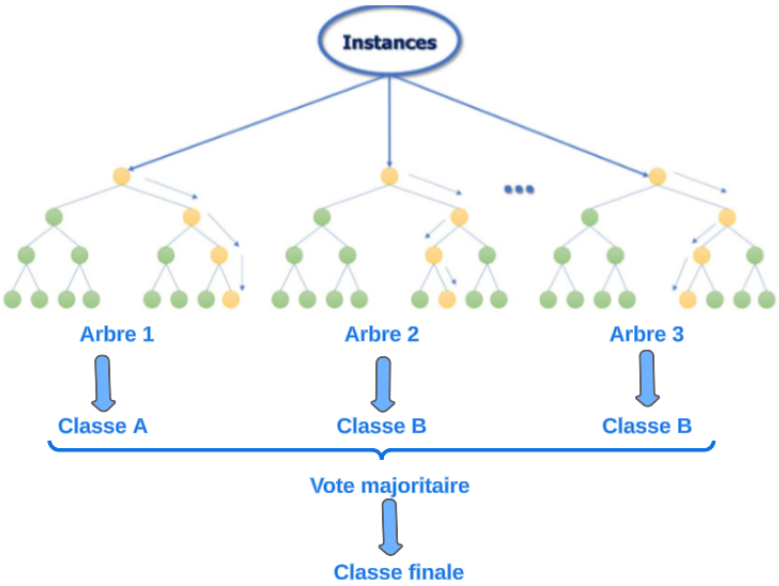

Figure 2.21: Principe d'une forêt aléatoire.

2.6 Conclusion

Ce chapitre nous a permis de retracer l'histoire des réseaux de neurones artificiels, en mettant en lumière leur évolution et leur importance . Nous avons ensuite détaillé les réseaux de neurones convolutifs (CNN), en expliquant leurs principaux paramètres . En terminant, par la présentation des différents classifieurs utilisés dans notre étude, notamment les machines à vecteurs de support (SVM), les forêts aléatoires (Random Forest), et les k-plus proches voisins (KNN). Chacun de ces classifieurs a été décrit en détail, en mettant en évidence leurs paramètres spécifiques et leurs contributions à l'amélioration des performances de notre système .
Afin d'améliorer les performances de notre système, nous proposons d'intégrer des outils mathématiques avancés. Ainsi, le prochain chapitre sera consacré à l'étude de l'intégrale floue et de l'optimisation par essaim de particules. Nous détaillerons comment ces techniques peuvent être combinées pour apporter des améliorations significatives à notre système de classification des maladies des plantes, enrichissant ainsi les approches explorées dans ce chapitre.

CHAPTER 3

INTÉGRALE FLOUE AVEC OPTIMISATION PAR ESSAIM DE PARTICULES

3.1 Introduction

Dans le domaine de la classification des maladies des plantes, la précision et la fiabilité des systèmes sont cruciales pour garantir des résultats efficaces. Afin d'atteindre cet objectif, notre étude a intégré des outils mathématiques avancés, en particulier les intégrales floues. Ces intégrales dépendent d'un paramètre appelé mesure floue, qui exerce une influence significative sur leurs résultats. Contrairement aux approches antérieures qui choisissaient ce paramètre de manière aléatoire, nous avons utilisé l'algorithme d'optimisation par essaim de particules pour déterminer ces mesures de manière précise et optimale.

Ce chapitre est dédié à une exploration approfondie de l'intégrale floue et de l'optimisation par essaim de particules. Nous examinerons en détail ces techniques, ainsi que la manière dont nous les avons combinées dans notre étude. Notre objectif à travers ce chapitre est de démontrer comment cette combinaison innovante peut contribuer de manière significative à l'amélioration de notre système de classification des maladies des plantes.

3.2 Intégrale floue

L'intégrale Floue (FI) est un concept mathématique proposé par sugeno en 1974 et represente en quelque sorte une extension de l'intégrale de Lesbsegue[59].L'intégrale floue (FI) est un opérateur d'agrégation non linéaire dont le comportement est défini par la mesure floue (FM)[60].Parmi les types d'intégrales floues les plus courants on retrouve l'intégrale de sugeno et celui de choquet , chacun offre des méthodes distinctes pour l'agrégation de l'information.
Les opérateurs d'intégrale floue (IF) combinent les décisions des classificateurs en fonction de la connaissance a priori de leur efficacité, exprimée en termes de mesures floues. Les décisions des classificateurs sont présentées sous forme des scores d'appartenance pour toutes les classes d'intérêt. Ensuite, l'IF est calculée séparément pour chaque classe en prenant en compte toutes les scores d'appartenance avec leurs mesures floues comme ci illustré dans la Figure 3.1. Ces opérations produisent des scores d'appartenance de l'IF, et l'exemple de test que l'on souhaite classifier sera assigné à la classe ayant le score le plus élevé parmi ceux prédits par cette intégrale

floue.[61]

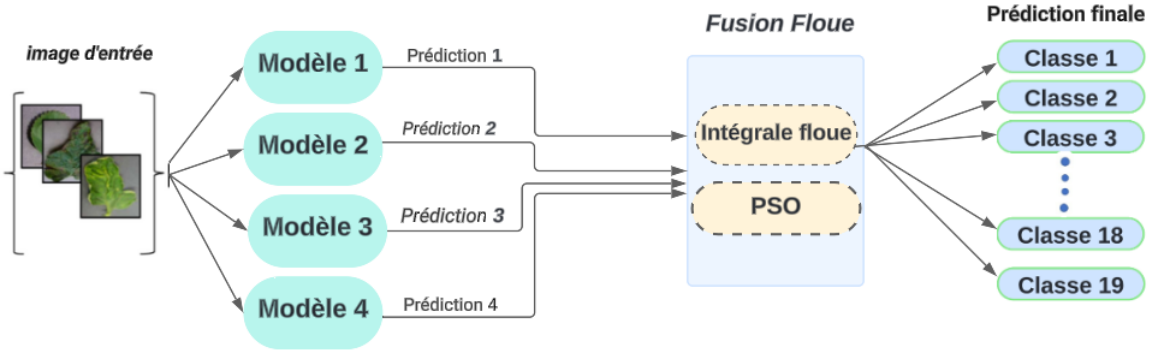

Figure 3.1: fusion floue

Pour comprendre toutes ces étapes de calcul, on va détaillé tous ces étapes dans les sections prochaines.

3.3 Mesures floues

Le concept de mesure est très important en mathématiques, en particulier pour la théorie des intégrales. Ces mesures classiques sont supposées respecter l'additivité. Bien que cette propriété soit intéressante dans certains cas, il apparaît que dans des applications de la théorie de la décision, la théorie des jeux et l'intelligence artificielle, il devient indispensable de définir des mesures non-additives. Un exemple criant de ce genre de situation est le travail d'un groupe de personnes. Si l'on représente l'efficacité d'une personne par une mesure, il est évident que l'efficacité globale du groupe ne sera pas l'addition des différentes mesures mais dépendra bien des interactions entre les personnes.[62]
Sugeno [62] propose donc de remplacer cette notion d'additivité par de la monotonie, et introduit ainsi les mesures floues, qui sont en fait des mesures non additives monotones.
Soit $Z = \{Z_i \mid i = 1, \ldots, N\}$ l'ensemble de N classificateurs tandis que $g(Z_i)$ désigne leurs performances. g est appelée une mesure floue si elle vérifie les propriétés suivantes [61] :

- $g(\emptyset) = 0$.
- $g(Z) = 1$.
- $g(Z_i) \leq g(Z_j)$ si $Z_i \subseteq Z_j$.

3.3.1 La mesure floue λ (lambda)

Selon la nature des mesures floues, Sugeno a montré que la mesure floue pour l'union de deux classificateurs ne correspond pas à la somme des mesures floues individuelles [61]. Comme solution, il a proposé la mesure floue λ qui exprime le degré d'interaction entre deux classifieurs Z_i et Z_j comme suit :

$$g(Z_i \cup Z_j) = g(Z_i) + g(Z_j) + \lambda g(Z_i) g(Z_j) \tag{3.1}$$

Pour chaque classe, λ est l'unique racine non nulle d'une équation de degré $N-1$ qui appartient à l'intervalle $[-1, \ldots, +\infty[$. Elle est calculée en résolvant l'équation suivante :

$$\lambda + 1 = \prod_{i=1}^{N} (1 + \lambda g(Z_i)) \tag{3.2}$$

3.4 Integrale de Sugeno

L'intégrale de Sugeno est un type d'intégrale floue utilisée pour agréger des informations provenant de différentes sources ou modéles. Elle est particulièrement utile dans le cadre des prises de décision pour des problématique de classification multiclasse
Pour calculer l'intégrale floue de Sugeno (SFI), pour une séquence de classifieurs $A_j = \{Z_1, \ldots, Z_j\}$, dont leurs mesures floues sont representées par $g(Z_j)$.
l'intégrale de Sugeno I_S d'une fonction $h : X \to [0, 1]$ par rapport à une mesure floue g , est calculée pour chaque classe par:

$$I_S(X_i) = \max_{j=1:N} [\min (h_j(X_i), g(Z_j))] \tag{3.3}$$

où $h_j(X_i)$ représente la probabilité de prédiction de la classe X_i par le classifieur Z_j.
et $g(Z_j)$ represent la mesure floue de classifieur Z_j

Pour illustrer le fonctionnement de cette intégrale de Sugeno, considérons un exemple avec trois classifieurs (3 modèles) donc $N = 3$ et calculons l'intégrale de Choquet pour la classe X_1:

$$I_S(X_1) = \max_{j=1:3} [\min (h_j(X_1), g(Z_j))] \tag{3.4}$$

En développant cette expression, nous obtenons:

$$I_S(X_1) = \max [\min (h_1(X_1), g(Z_1)) , \min (h_2(X_1), g(Z_2)) , \min (h_3(X_1), g(Z_3))] \tag{3.5}$$

Le principe de fonctionnement de cette intégrale Sugeno est illustré en détail dans la Figure 3.2, qui montre un exemple d'application de l'intégrale de sugeno pour trois modèles.

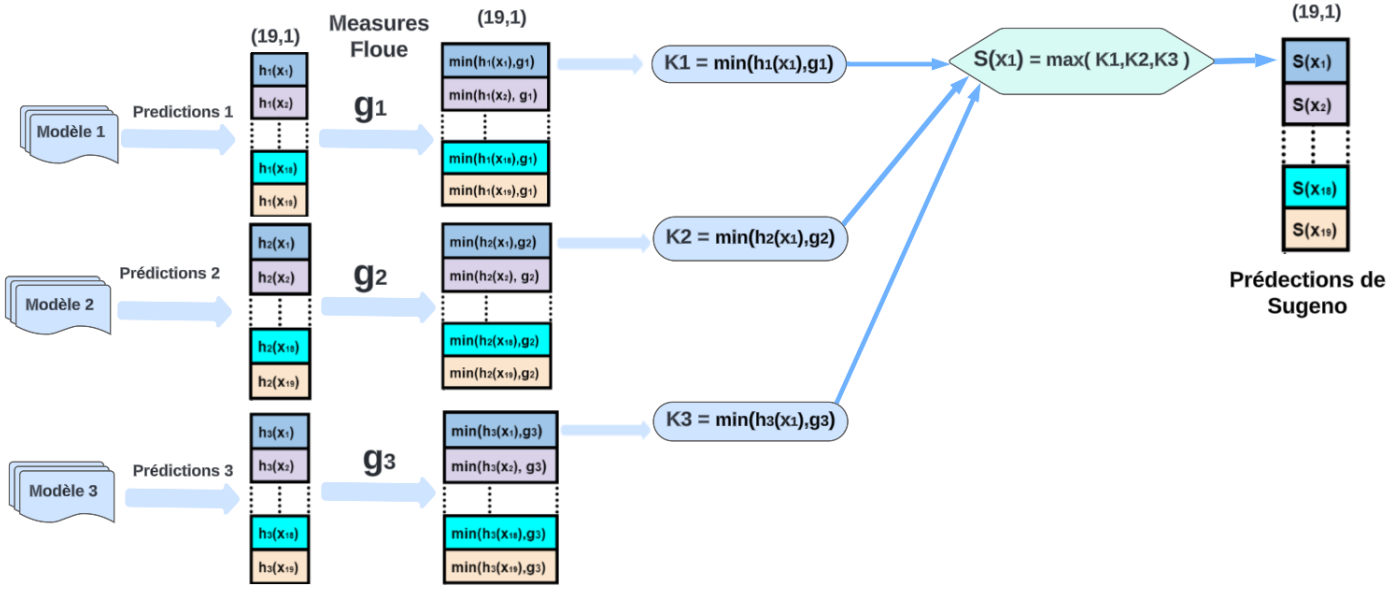

Figure 3.2: Exemple de principe de l'integrale de Sugeno cas 3 modéle

3.5 Integrale de Choquet

L'intégrale de Choquet [63] est une intégrale sous-additive ou sur-additive utilisée pour agréger les prédictions de plusieurs modèles de manière flexible. Créée par le mathématicien français Gustave Choquet, cette intégrale permet de combiner les sorties des modèles tout en prenant en compte les interactions entre eux.

Pour calculer l'intégrale de Choquet, les prédictions des classifieurs $h_j(X_i)$ doivent être réarrangées de manière à respecter la relation suivante :

$$h_1(X_i) \geq \ldots \geq h_N(X_i) \geq 0$$

où $h_j(X_i)$ représente la probabilité de prédiction de la classe X_i par le classifieur j.

Nous disposons d'une séquence de classifieurs $A_j = \{Z_1, \ldots, Z_j\}$, dont les mesures floues g sont définies comme suit :

$$g(A_1) = g(Z_1) = g_1 \tag{3.6}$$

$$g(Z_2) = g_2 \tag{3.7}$$

$$g(A_2) = g(A_1 \cup Z_2) = g(Z_1) + g(Z_2) + \lambda g(Z_1)g(Z_2) \tag{3.8}$$

$$g(A_j) = g(A_{j-1} \cup Z_j) \tag{3.9}$$

$g(A_j)$ represente la mesure floue conjointe sur un ensemble A_j peut être obtenue par :

$$g(A_j) = g(A_{j-1}) + g(Z_j) + \lambda g(A_{j-1})g(Z_j) \tag{3.10}$$

Ensuite, l'intégrale de Choquet I_C d'une fonction $h : X \to [0, 1]$ par rapport à une mesure floue g est calculée pour chaque classe par :

$$I_C(X_i) = \sum_{j=1}^{N} \left[g(A_j) - g(A_{j-1})\right] h_j(X_i) \tag{3.11}$$

Pour illustrer le fonctionnement de cette intégrale de Choquet, considérons un exemple avec trois classifieurs (donc $N = 3$) et calculons l'intégrale de Choquet pour la classe X_1:

$$I_C(X_1) = \sum_{j=1}^{3} \left[g(A_j) - g(A_{j-1})\right] h_j(X_1) \tag{3.12}$$

$$I_C(X_1) = \left[g(A_1)\right] h_1(X_1) + \left[g(A_2) - g(A_1)\right] h_2(X_1) + \left[g(A_3) - g(A_2)\right] h_3(X_1) \tag{3.13}$$

En posant $g(Z_1) = g_1$, $g(Z_2) = g_2$, et $g(Z_3) = g_3$, nous obtenons :

$$g(A_1) = g(Z_1) = g_1 \tag{3.14}$$

$$g(A_2) = g(A_1 \cup Z_2) = g(A_1) + g(Z_2) + \lambda g(A_1)g(Z_2) \tag{3.15}$$

$$g(A_3) = g(A_2 \cup Z_3) = g(A_2) + g(Z_3) + \lambda g(A_2)g(Z_3) \tag{3.16}$$

Le principe de fonctionnement de cette intégrale de Choquet est illustré en détaille dans la Figure 3.3, qui montre un exemple avec trois modèles.

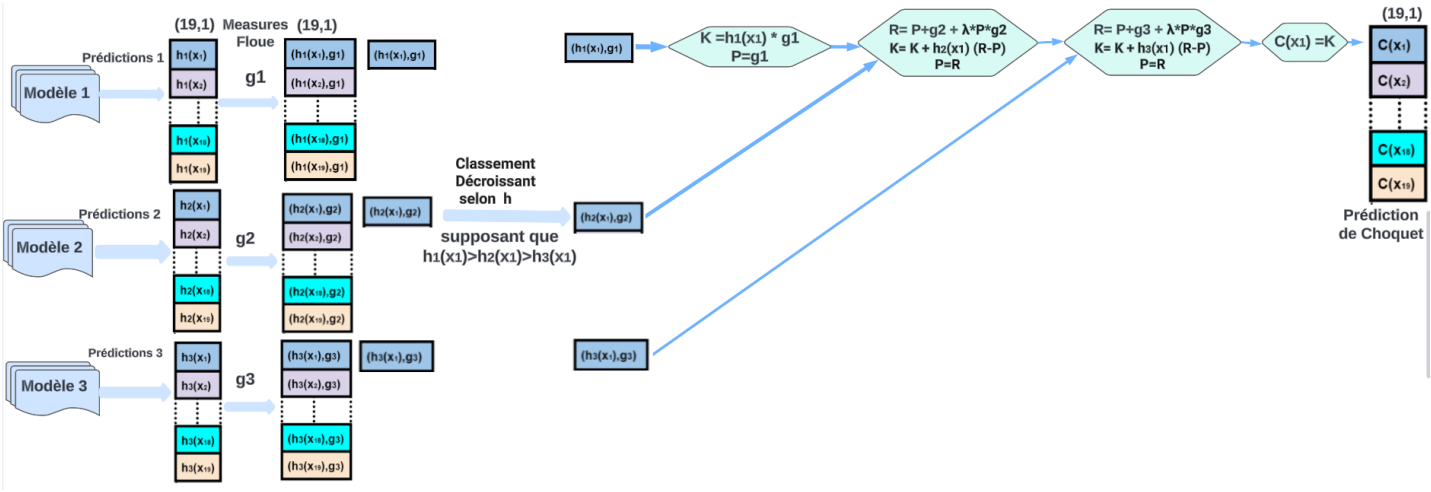

Figure 3.3: Exemple de principe de l'integrale de Choquet cas 3 modéle

Cependant, un inconvénient des méthodes basées sur l'intégrale floue (FI) est la spécification de la mesure floue (FM). Par exemple, définir manuellement la FM, comme l'a fait l'auteur dans [61], où les valeurs de la FM sont attribuées en fonction de la précision d'entraînement des modèles, ne semble pas très précis. C'est pourquoi, dans ce travail, l'optimisation par essaims de particules (PSO) est utilisée pour trouver les valeurs optimaux des ces measures floues à attribuer aux classifieurs employés.

3.6 Optimisation par essaim de particules

Pour attribuer efficacement les mesures floues aux classifieurs dans l'intégrale floue, on utilise l'optimisation par essaims particulaires (PSO), une technique d'intelligence en essaim inspirée du comportement des vols d'oiseaux et des bancs de poissons comme illustré dans la Figure 3.4. Proposée par Russel Eberhart et James Kennedy en 1995, cette méthode vise à optimiser un modèle en améliorant itérativement une solution candidate en fonction d'une mesure de qualité. Le PSO utilise une population de particules, chaque particule représentant une solution avec une position (vecteur solution) et une vitesse. Chaque particule mémorise ses meilleures performances ainsi que celles de ses voisines (informateurs), permettant de converger vers une solution optimale.[64]

(a) (b)

Figure 3.4: Groupe de (a) oiseaux, (b) poissons.

3.6.1 Problème d'optimisation

L'optimisation appartient au domaine de la Recherche Opérationnelle, qui représente la science de la prise de décision. Un problème d'optimisation en mathématiques consiste à trouver la meilleure solution possible pour un problème donné.

Soit $n \in \mathbb{N}^*$ un entier strictement positif,
$X \subset \mathbb{R}^n$ un sous-ensemble non vide,
$f : X \to \mathbb{R}$ une fonction à valeur réelle.

Un problème d'optimisation consiste à trouver la meilleure solution $x \in X$, appelée solution globale.

La meilleure solution du problème varie selon l'objectif. On parle de minimisation ou de maximisation de la fonction f sur l'ensemble X. On note :

$$f(x^*) = \min_{x \in X} f(x)$$

ou

$$f(x^*) = \max_{x \in X} f(x).$$

3.6.2 Fonction objectif

La fonction f, également connue sous le nom de critère d'optimisation ou fonction de fitness, est celle que cet algorithme d'optimisation cherche à optimiser en trouvant son optimum. Par exemple, dans le cas de notre projet, la fonction de fitness que nous voulons optimiser est celle qui calcule l'exactitude en utilisant l'intégrale de Choquet ou bien de Sugeno. Ainsi, nous cherchons à trouver les valeurs optimales des mesures floues qui nous permettent d'atteindre l'exactitude maximale en utilisant ces intégrales.

3.6.3 Principe de fonctionnement de PSO

Le principe de cet algorithme consiste à déplacer des particules dans l'espace de recherche de sa fonction objectif, dans notre cas, la fonction objectif (accuracy-sugeno ou accuracy-choquet). Les étapes de déroulement de l'algorithme de PSO sont illustré dans la Figure 3.5

Voici un aperçu détaillé des étapes de fonctionnement de l'algorithme de PSO :

1. **Définition des paramètres :** Dans cette première étape, on définit l'espace de recherche de la fonction objectif, le nombre de particules dans notre essaim ainsi que le critère d'arrêt, qui est, dans notre cas, le nombre maximal d'itérations, l'inertie, et les constantes d'accélération.

2. **Initialisation** : Initialiser aléatoirement les positions et les vitesses des particules dans l'espace de recherche souhaité.

3. **Déplacement, Évaluation et Mise à Jour des Particules** : Les particules se déplacent pour trouver la meilleure position (les mesures floues optimales) qui maximise la fonction objective. Chaque particule évalue la qualité de sa position actuelle et garde

en mémoire sa meilleure position, c'est-à-dire la position avec laquelle elle a atteint la meilleure performance. À chaque pas de temps, chaque particule compare la performance obtenue avec sa position actuelle X_i à celle obtenue en utilisant sa meilleure position de particule P_i. Si, avec cette position actuelle, elle obtient des performances meilleures que celles obtenues avec sa meilleure position, alors la particule gardera cette position actuelle comme sa meilleure position. Ensuite, elle compare les performances atteintes avec sa nouvelle position de particule avec les performances atteintes en utilisant la meilleure position de l'essaim P_g. Si elle est meilleure que celle-ci, cette nouvelle position deviendra la meilleure position de l'essaim, comme illustré dans la figure 3.5. Cette opération doit être effectuée pour chaque particule de l'essaim à chaque itération, et les opérations se répètent jusqu'à atteindre le nombre maximal d'itérations. Dans ce cas, l'algorithme s'arrête et retourne la meilleure position qui correspond aux mesures floues optimales recherchées.

La position et la vitesse des particules sont mises à jour pour l'itération **(t+1)** selon les équations suivantes

$$v_{i,d}^{t+1} \leftarrow \omega v_{i,d}^{t} + a_1 r_1 (p_{i,d}^{t} - x_{i,d}^{t}) + a_2 r_2 (p_{g,d}^{t} - x_{i,d}^{t}) \tag{3.17}$$

$$x_{i,d}^{t+1} \leftarrow x_{i,d}^{t} + v_{i,d}^{t+1} \tag{3.18}$$

où ω est le poids d'inertie.Lorsque ω est inférieur à 1, les vitesses de toutes les particules tendront vers zéro, conduisant à la convergence des particules vers la solution optimale.
Les coefficients a_1 et a_2 sont les constantes d'accélération.
r_1 et r_2 sont des nombres aléatoires issus d'une distribution uniforme $U(0,1)$.
$p_{g,d}^{t} = (p_{g1}, p_{g2}, \ldots, p_{gd})$ est la meilleure position connue de l'ensemble (l'essaim) à l'itération **(t)** dans l'espace de recherche de dimension d.
$p_{i,d}^{t} = (p_{i1}, p_{i2}, \ldots, p_{id})$ est la meilleure position connue de la particule i à l'itération **(t)** dans l'espace de recherche de dimension d.
$x_{i,d}^{t+1} = (x_{i1}, x_{i2}, \ldots, x_{id})$ représente la position actuelle de la particule i à l'itération **(t+1)** dans l'espace de recherche de dimension d.
$v_{i,d}^{t+1} = (v_{i1}, v_{i2}, \ldots, v_{id})$ représente la vitesse actuelle de la particule i à l'itération **(t+1)** dans l'espace de recherche de dimension d.

Chaque particule décide de son prochain mouvement en combinant trois informations : sa position actuelle, sa meilleure position actuelle, et la meilleure position de l'essaim.

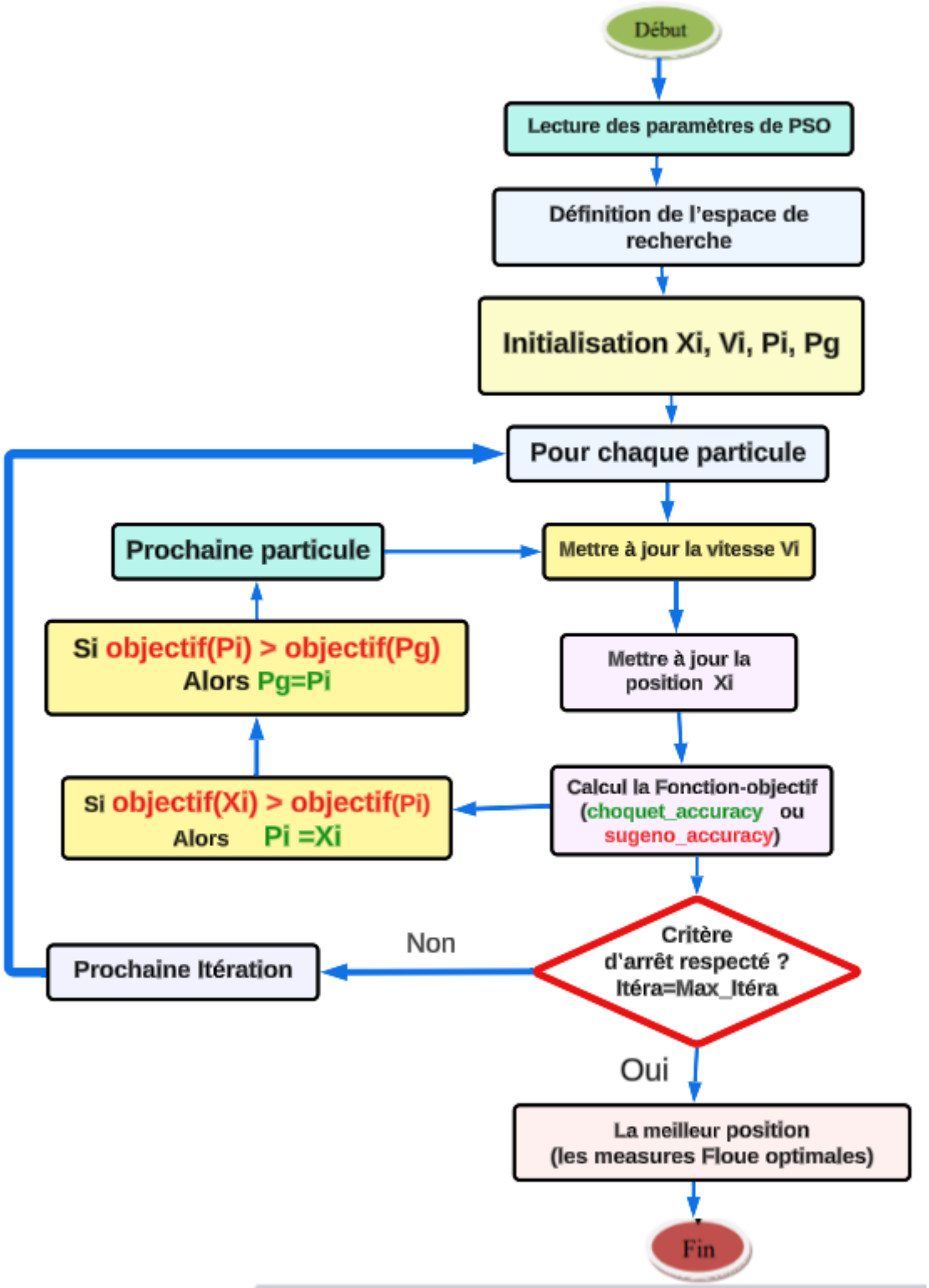

Figure 3.5: Déroulement de l'algorithme de l'optimisation par essaim de particules

3.6.4 Paramètres de l'algorithme(PSO)

- Le nombre de particules (la taille d'essaim)
- La dimension du problème(dimension de l'espace de recherche de la fonction objectif)
- L'intervalle de recherche (l'espace de recherche)
- Les valeurs des coefficients
- L'inertie
- Le critère d'arrêt(nombre d'itérations)

3.6.5 Avantages de l'algorithme(PSO)

- peut converger rapidement vers des bonnes solutions.
- implémentations simples, avec peu de paramètres
- versatilité: peut résoudre beaucoup de différents problèmes.

3.7 Conclusion

À travers ce chapitre, nous avons exploré les concepts d'intégrale floue et d'optimisation par essaim de particules. Nous avons détaillé le principe de fonctionnement de deux types d'intégrales floues, à savoir l'intégrale de Sugeno et celui de Choquet, chacun offre des méthodes distinctes pour l'agrégation de l'information. Nous avons également découvert comment surmonter la difficulté de la détermination des mesures floues de ces intégrales en utilisant la technique de l'optimisation par essaim de particules (PSO), qui permet de trouver les valeurs optimales de ces mesures floues à attribuer aux modèles employés.

Dans le chapitre prochain, nous détaillerons la méthodologie de travail que nous avons adoptée, ainsi que les multiples modèles de classification que nous avons développés au cours de ce projet. Nous exposerons également les résultats expérimentaux obtenus.

CHAPTER 4

MÉTHODOLOGIE ET RÉSULTATS EXPÉRIMENTAUX

4.1 Introduction

Dans ce chapitre, nous détaillerons la méthodologie de travail adoptée ainsi que les multiples modèles de classification développés au cours de ce projet. Nous commencerons par décrire les étapes précises de notre approche méthodologique, en expliquant les choix stratégiques et techniques qui ont guidé notre travail. Ensuite, nous introduirons les différents modèles de classification élaborés, en mettant en lumière leurs spécificités et les raisons de leur sélection.

Nous présenterons ensuite les résultats expérimentaux obtenus lors de nos tests. Ces résultats seront analysés en profondeur afin de mettre en évidence les performances et les limites de chaque modèle.Les modèles ont été entraînés en utilisant Google Colaboratory.

4.2 Ensembles de données utilisés

L'ensemble de données sur lequel nous avons travaillé comprenant 4 types d'aliments (Potato, Tomato et Grape et Pepper bell) comprenant 19 classes dont 15 maladies, avec un total de 26 847 images.Cet ensemble de données provient de la base de données PlantVillage, accessible au public sur Kaggle [27] .

PlantVillage est une base de données publique Publiée par l'Université d'État de Pennsylvanie, cette base de données contient 54 305 images RGB réparties en 38 classes de maladies couvrant 14 espèces de plantes. Les images montrent à la fois des feuilles saines et malades, avec des dimensions standardisées de 256 $\times$ 256 pixels[65].Des images d'échantillon du cette base de données sont présentées dans la Figure 4.1
Toutes les images de PlantVillage ont été prises dans des stations de recherche expérimentale associées aux Land Grant universités aux Etats-Unis (Penn State, Florida State, Cornell, et d'autres).Les images ont été capturées par des techniciens dans diverses conditions de lumière, allant du soleil éclatant à un ciel nuageux, pour refléter les conditions réelles auxquelles les utilisateurs finaux (comme les agriculteurs utilisant des smartphones) pourraient faire face. Pour chaque feuille, plusieurs images (généralement 4 à 7) ont été prises sous différents angles à l'aide d'un appareil photo numérique standard (Sony DSC-Rx100/13, 20,2 mégapixels) en mode automatique. Les feuilles étaient détachées des plantes infectées et placées sur un fond

de papier gris ou noir pour la prise de vue. Après la capture, les images étaient éditées pour recadrer une grande partie de l'arrière-plan et orienter les feuilles de manière uniforme, avec la pointe dirigée vers le haut. Pour les cultures aux feuilles trop grandes, comme le maïs et la courge, des images de différentes sections de la même feuille ont été prises afin de conserver une haute résolution et une vue de proximité [66].

PlantVillage continue de s'enrichir avec de nouvelles images collectées, et la liste des sources utilisées devrait s'élargir à l'avenir. Les stations de recherche expérimentale, tant publiques que privées, permettent de collecter de nombreuses images en peu de temps, rendant cette base de données un outil précieux pour la recherche sur les maladies des plantes.

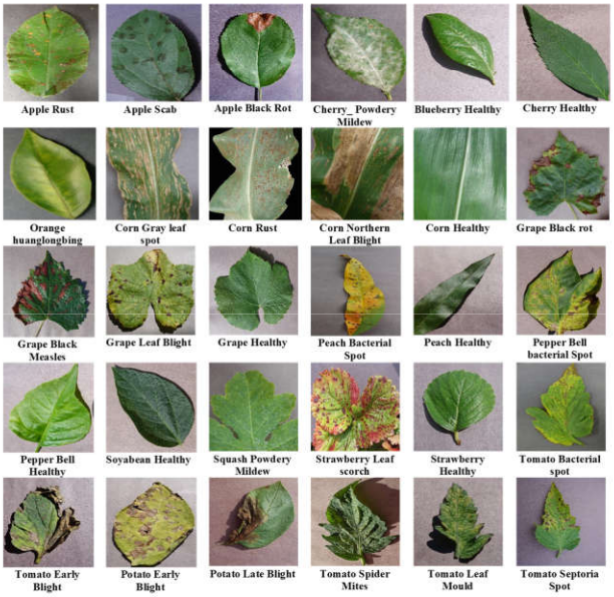

Figure 4.1: Exemples des images des plantes de la base de données PlantVillage[65].

4.2.1 Distribution des échantillons

Notre travail est réalisé sur un ensemble comprenant 26 847 images représentant 4 types de plantes différents, à savoir : la tomate (Tomato), le raisin (Grape), la pomme de terre (Potato) et le poivron (Pepper bell). Cet ensemble de données comporte un total de 19 classes, tel que Grape comporte 4 classes, Potato comporte 3 classes,Tomato comporte 10 classes et Pepper bell comporte 2 classes.

Il est crucial de souligner que le nombre d'échantillons par classe dans cet ensemble de données est très déséquilibré comme l'indique le tableau 4.1.D'après ce tableau on distingue des classes minoritaires comme Tomato Leaf-Mold qui compte 952 images et des classe majoitaire comme Tomato yellow Leaf Curl Virus qui compte 5357 images.Ainsi, cette répartition déséquilibrée constitue un défi majeur sur les performances des modèles d'apprentissage automatique. Ce déséquilibre des données peut conduire à des modèles fortement biaisés en faveur de la classe majoritaire , ce qui fait que la classe minoritaire peut être mal classée ou complètement ignorée. En effet, le nombre limité d'échantillons pour les classes minoritaires réduit les possibilités d'apprentissage et de généralisation des caractéristiques associées à ces classes. Par conséquent, bien que le modèle optimise la précision globale, il affiche une haute précision pour les classes majoritaires mais de faibles performances pour les classes minoritaires. Cela entraîne une incapacité à généraliser correctement les données non observées, conduisant à des erreurs de classification et à une mauvaise prise de décision.

Pour résoudre ce problème du déséquilibre des données, nous procédons à une augmentation de données. Cette technique, connue sous le nom de "data augmentation", sera abordée en détail dans la section prochaine.

Type d'aliments	Maladie associée	Nombre d'images
Grape	Black rot	1180
	Esca (Black Measles)	1383
	Leaf blight (Isariopsis Leaf Spot)	1076
	Healthy	423
Pepper bell	Bacterial spot	997
	Healthy	1477
Potato	Early blight	1000
	Late blight	1000
	Healthy	152
Tomato	Bacterial spot	2127
	Early blight	1000
	Late blight	1908
	Leaf Mold	952
	Septoria leaf spot	1771
	Spider mites Two-spotted spider mite	1676
	Target Spot	1404
	Yellow Leaf Curl Virus	5357
	Mosaic virus	373
	Healthy	1591
Nombre de classes	19	26847

Table 4.1: Répartition des échantillons de l'ensemble de donnée Plant Village

4.2.2 Augmentation des données

L'augmentation des données est une technique cruciale utilisée pour augmenter la quantité de données disponible pour l'apprentissag Elle consiste à créer de nouvelles versions des données existantes en y apportant des modifications mineures. Cette technique permet de régulariser le modèle et de réduire le risque de surapprentissage.

Dans notre étude, nous avons appliqué les transformations suivantes sur les images pour augmenter notre ensemble de données :

1. Rotation des images à différents angles (-45°, 45°, 90°, -90°, 180°).(voir la figure 4.2)
2. Zoom sur les images.(voir la figure 4.3)
3. Inversion des images.(voir la figure 4.4)

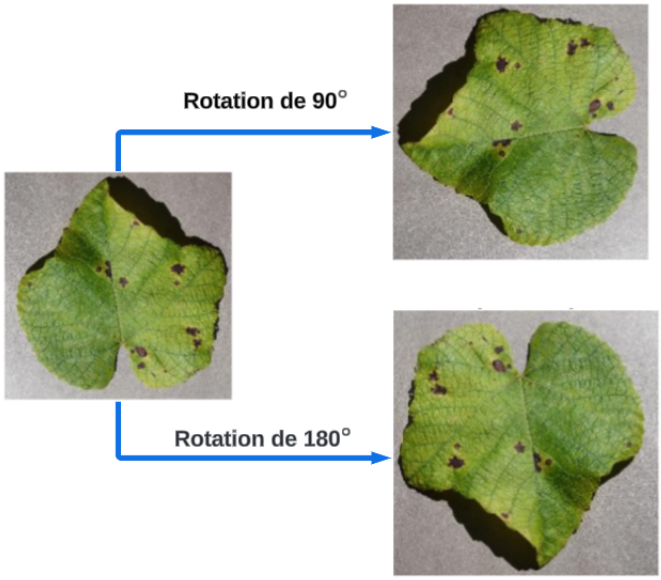

Figure 4.2: Exemples des Rotations faites sur les images

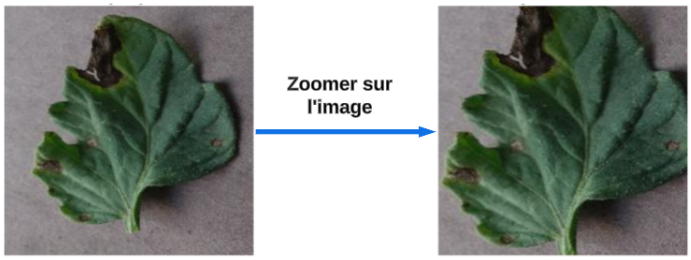

Figure 4.3: Exemple d'une image zoomée

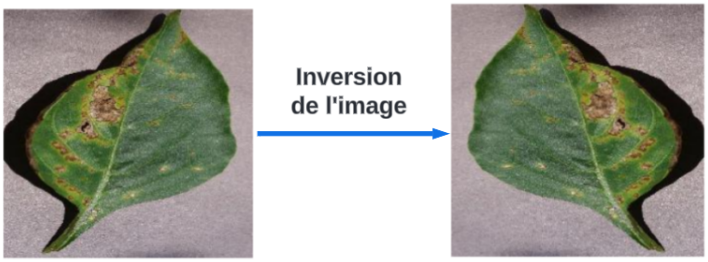

Figure 4.4: Exemple d'inversion d'une image

Le tableau 4.2 montre la nouvelle répartition des échantillons de notre ensemble de données après l'augmentation de données.

Durant nos entraînements et évaluations, on a procédé à la subdivisions suivante que l'on a trouvé dans la littérature :80% des images sont utilisés pour la phase d'Entraînement, 10% pour la validation et 10% pour la phase de test comme indique dans le tableau 4.2.

Type d'aliment	Maladie associée	Entraînement 80%	Validation 10%	Test 10%	Totale
Grape	Black rot	944	118	118	1180
	Esca (Black Measles)	1106	138	139	1383
	Leaf blight (Isariopsis Leaf Spot)	860	107	109	1076
	Healthy	800	100	100	1000
Pepper bell	Bacterial spot	1178	147	148	1473
	Healthy	1181	147	149	1477
Potato	Early blight	800	100	100	1000
	Late blight	800	100	100	1000
	Healthy	799	100	101	1000
Tomato	Bacterial spot	1701	212	214	2127
	Early blight	1901	237	239	2377
	Late blight	1892	236	238	2366
	Leaf Mold	1884	235	236	2355
	Septoria leaf spot	1749	218	220	2187
	Spider mites Two-spotted spider mite	1744	218	218	2180
	Target Spot	1852	231	233	2316
	Yellow Leaf Curl Virus	1960	245	245	2450
	Mosaic virus	1848	231	231	2310
	Healthy	1930	241	242	2413
Total	**19**	**26971**	**3366**	**3386**	**33723**

Table 4.2: Répartition des différents ensembles : entraînement, validation et test après l'augmentation des données

4.3 Logiciels, libraires et matériels

4.3.1 Python

Python est un langage de programmation de haut niveau, interprété et orienté objet, développé par Guido van Rossum en 1989 et initialement lancé en 1991. Son nom fait référence à la troupe Monty Python et il est apprécié pour sa simplicité et sa lisibilité. Python est largement utilisé dans divers domaines tels que le développement Web, l'analyse de données, l'intelligence artificielle et la création d'applications de bureau.Ses caractéristiques incluent une syntaxe simple, une compatibilité multiplateforme et un processus de développement rapide grâce à son interpréteur. Python est également flexible, prenant en charge différents paradigmes de programmation. Il est le choix privilégié pour les développeurs en raison de sa productivité et de son écosystème riche en bibliothèques, notamment dans le domaine de l'intelligence artificielle avec des outils comme TensorFlow et scikit-learn.[67]

4.3.2 Scikit-learn

Scikit-Learn est une bibliothèque open source Python dédiée à l'apprentissage automatique. Créée en 2007 par l'ingénieur David Cournapeau, elle est réputée pour sa documentation complète et sa communauté active, qui fournissent un soutien et des ressources supplémentaires. Scikit-Learn permet de construire des modèles prédictifs en s'appuyant sur des algorithmes de régression linéaire et logistique, des Support Vector Machine (SVM), des arbres de dé-

cision et des forêts aléatoires. Intégrée avec d'autres bibliothèques Python telles que NumPy et pandas, elle simplifie l'importation et la manipulation des données.[68]

4.3.3 Keras

Keras est une bibliothèque open source écrite en Python, principalement basée sur les travaux du développeur de Google, François Chollet, dans le cadre du projet ONEIROS (Open-ended Neuro-Electronic Intelligent Robot Operating System). Une première version du logiciel multiplateforme a été publiée en 2015. Son objectif principal est de permettre la création rapide de réseaux neuronaux. Contrairement à un framework propre, Keras fonctionne comme une interface de programmation applicative (API) permettant d'accéder et de programmer différents frameworks d'apprentissage automatique. Parmi les frameworks pris en charge par Keras, on trouve notamment Theano, Microsoft Cognitive Toolkit (anciennement CNTK) et TensorFlow. [69]

4.3.4 Tensorflow

TensorFlow est un framework complet et open source utilisé par de nombreux développeurs et chercheurs pour créer des modèles de Deep Learning et effectuer des tâches d'apprentissage automatique complexes. Il s'agit d'une boîte à outils de mathématiques symboliques qui exécute une variété de tâches, notamment la formation et l'inférence de réseaux neuronaux profonds. Il a été développé par Google et a été initialement publié fin 2015, avec la première version stable suivant en 2017. TensorFlow fournit une interface de programmation Python et C++ pour le développement de modèles d'apprentissage automatique. Il se distingue de ses concurrents par une communauté très active, une documentation complète et une architecture moderne.[70]

4.3.5 Matériel (Hardware)

Notre problématique étant relativement complexe, c'est la raison pour laquelle nous avons mené tous les entraînements sur le cloud. Étant donné que notre projet implique l'utilisation de CNN, lesquels sont très gourmands en ressources, nous avons opté pour la solution décrite ci-dessous pour l'entraînement des différents modèles.

4.3.5.1 Google Colaboratory

Colaboratory, souvent abrégé en "Colab", est un produit de Google Research. Colab permet à n'importe qui d'écrire et d'exécuter le code Python de son choix par le biais du navigateur. C'est un environnement particulièrement adapté au machine learning, à l'analyse de données. Plus précisément, Colab est un service hébergé de notebooks Jupyter qui ne nécessite aucune configuration et permet d'accéder sans frais à des ressources informatiques, y compris des GPU et des TPU.[71]

4.4 Métriques d'évaluation

Une métrique d'évaluation quantifie la performance d'un modèle prédictif. Ces métriques permettent d'identifier les points faibles du modèle et de guider son ajustement. Pour un problème de classification, le choix des métriques varie selon les distributions des données, qu'elles soient équilibrées ou déséquilibrées. Cependant, sélectionner la métrique appropriée n'est pas évident pour tous les modèles. Dans notre cas, nous travaillons avec un ensemble de données équilibré. Dans la section suivante, nous décrirons les métriques utilisées pour évaluer nos modèles.

4.4.1 Exactitude de la classification (Classification Accuracy)

L'Exactitude est la mesure d'évaluation la plus fondamentale. Il mesure l'exactitude globale des prédictions d'un modèle en calculant le rapport entre les échantillons correctement classés et le nombre total d'échantillons.La formule de l'exactitude est la suivante :

$$Accuracy = \frac{VP + VN}{VP + VN + FP + FN} \tag{4.1}$$

Bien que l'exactitude fournisse un aperçu général des performances d'un modèle, elle peut ne pas convenir aux ensembles de données présentant des distributions de classes déséquilibrées. Pour mieux évaluer notre modèle, nous devons explorer des mesures d'évaluation supplémentaires.

4.4.2 Matrice de confusion(Confusion Matrix)

La matrice de confusion constitue un outil d'évaluation puissant qui fournit une représentation tabulaire des prédictions faites par un modèle par rapport aux étiquettes de vérité terrain. Elle permet aux professionnels de comprendre les forces et les faiblesses de leurs modèles de manière systématique. Elle est généralement présentée sous forme de matrice $n \times n$, où n correspond au nombre de classes disponibles dans notre problème de classification.

Il y a quatre éléments clés dans cette matrice : True Positive (TP), True Negative (TN), False Positive (FP) et False Negative (FN). Les colonnes représentent les valeurs réelles et les lignes correspondent aux valeurs prédites par le modèle, comme le montre la Figure 4.5 qui représente la matrice de confusion pour un problème de classification binaire (2×2).

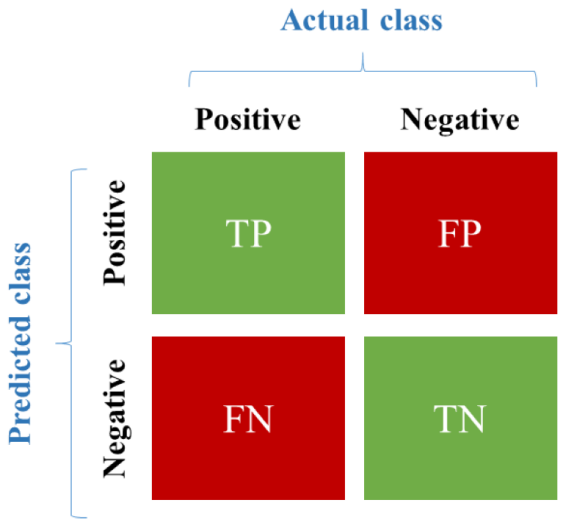

Figure 4.5: Matrice de confusion

- **True Positive (TP)** : sont des valeurs vraies réelles correctement prédites comme vraies.
- **True Negative (TN)** :sont des valeurs fausses réelles correctement prédites comme fausses.
- **False Positive (FP)**:sont des valeurs fausses réelles incorrectement prédites comme vraies.
- **False Negative (FN)**:sont des valeurs vraies réelles incorrectement prédites comme fausses.

4.4.2.1 Précision

La métrique de précision révèle le nombre de classes prédites qui sont correctement étiquetées. Il s'agit du rapport entre les positifs correctement identifiés (vrais positifs, VP) et tous les positifs identifiés (VP et FP), calculé par la formule suivante :

$$Précision = \frac{VP}{VP + FP} \tag{4.2}$$

Une valeur de précision élevée indique que lorsque le modèle prédit un résultat positif, il est souvent correct.

4.4.2.2 Recall ou Sensibilité ou Rappel

Le rappel, également connu sous le nom de sensibilité ou recall ou taux de vrais positifs, mesure le rapport entre les vrais positifs et le nombre total d'échantillons positifs réels et est calculé par la formule suivante :

$$Recall = \frac{VP}{VP + FN} \tag{4.3}$$

Une valeur de rappel élevée indique que le modèle peut détecter efficacement les échantillons positifs.

4.4.2.3 F1-score

Le score F1 est une mesure qui combine précision et rappel en une seule valeur, fournissant une évaluation équilibrée des performances d'un modèle. Elle est calculée par la formule suivante :

$$F1 - score = \frac{2 \cdot Précision \cdot Recall}{Précision + Recall} \tag{4.4}$$

4.5 Protocole expérimental

La mise en œuvre d'un système de classification des maladies des plantes nécessite un protocole expérimental rigoureux pour obtenir des résultats satisfaisants. Dans cette section, nous décrivons en détail le protocole expérimental adopté, couvrant la méthodologie, la répartition des données et de paramétrage de nos réseaux de neurones convolutifs. Nous abordons également la façon d'entraîner nos classifieurs utilisés à la suite de nos descripteurs et la sélection des hyperparamètres optimaux afin de garantir une classification précise et efficace.

4.5.1 Méthodologie

Notre but consiste à concevoir un modèle permettant de classifier 19 classes des maladies des plantes. Pour ce faire, nous avons implémenté deux protocoles, comme montre dans la figure 4.6 ,afin de comparer les performances obtenues par chacun d'eux.

Pour le protocole 1, nous allons créer au total de 5 modèles. Tout d'abord, un modèle pour la classification selon le type de plante. Ensuite, 4 modèles, chacun dédié à la classification des maladies d'un type spécifique de plante, étant donné que nous disposons de 4 types de plantes différents. Nous ferons fonctionner ces 5 modèles ensemble pour effectuer la classification en 19 classes. Nous fournissons d'abord une image au modèle de classification selon le type de plante. En fonction du résultat, nous sélectionnons le modèle approprié pour détecter la maladie.

Pour le protocole 2, nous utilisons un seul modèle pour effectuer la classification en 19 classes à la fois.

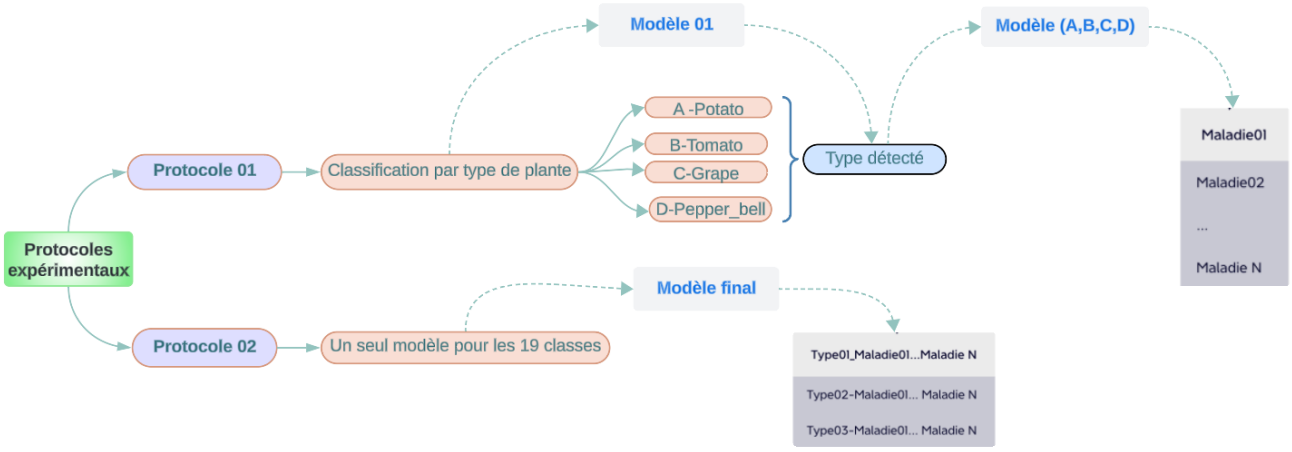

Figure 4.6: Les protocoles utilisées pour classification les maladies des plantes

4.5.2 Répartition des données

Notre méthodologie repose sur l'utilisation de deux protocoles distincts, comme nous l'avons vu dans la section précédente, et la distribution des données est étroitement liée au choix spécifique du protocole. Dans la section suivante, nous détaillerons la manière dont les données sont réparties de façon appropriée pour chacun des protocoles utilisés

4.5.2.1 Protocole 01

Pour le premier modèle du protocole 1, c'est-à-dire le modèle de classification selon le type de plante, nous avons quatre classes : Tomato, Potato, Grape et Pepper bell. Pour ce modèle, nous avons pris les images de toutes les classes constituant chaque type de plante afin d'assurer la plus grande diversité possible. À savoir, Grape comprend 4 classes avec 4 692 images, Potato contient 3 classes avec 3 000 images, Pepper bell inclut 2 classes avec 2 951 images, et enfin, Tomato regroupe 10 classes avec 23 081 images. Il est important de noter que 80 % des images

sont utilisées pour l'entraînement du modèle, 10% pour la validation et 10 % pour le test. Le tableau 4.3 illustre cette répartition.

	Tomato	Grape	Potato	Pepper bell
Entraînement 80%	18461	3752	2399	2360
Validation 10%	2304	468	300	294
Test 10%	2316	472	301	297
Total	23081	4692	3000	2951

Table 4.3: Répartition des données pour la classification du type de plante

Pour la classification des maladies de Potato sachant qu'il dispose de 3 classes ,les données ont été répartie sur 3 ensembles : 80 % des images sont utilisées pour l'entraînement du modèle, 10% pour la validation et 10 % pour le test. Le tableau 4.4 illustre cette répartition.

	Entraînement 80%	Validation 10%	Test 10%
Early blight	800	100	100
Late blight	800	100	100
Healthy	799	100	101
Totale	2399	300	301

Table 4.4: Répartition des données
Type: Potato

Pour la classification des maladies de Tomato sachant qu'il dispose de 10 classes ,les données ont été répartie sur 3 ensembles : 80 % des images sont utilisées pour l'entraînement du modèle, 10% pour la validation et 10 % pour le test. Le tableau 4.5 illustre cette répartition.

	Entraînement 80%	Validation 10%	Test 10%
Bacterial spot	1701	212	214
Early blight	1901	237	239
Late blight	1892	236	238
Leaf Mold	1884	235	236
Septoria leaf spot	1749	218	220
Spider mites Two-spotted spider mite	1744	218	218
Target Spot	1852	231	233
Yellow Leaf Curl Virus	1960	245	245
Mosaic virus	1848	231	231
Healthy	1930	241	242
Totale	18461	2304	2316

Table 4.5: Répartition des données
Type: Tomato

Pour la classification des maladies de Grape sachant qu'il dispose de 4 classes ,les données ont été répartie sur 3 ensembles : 80 % des images sont utilisées pour l'entraînement du modèle, 10% pour la validation et 10 % pour le test. Le tableau 4.6 illustre cette répartition.

	Entraînement 80%	Validation 10%	Test 10%
Black rot	944	118	118
Esca (Black Measles)	1106	138	139
Leaf blight (Isariopsis Leaf Spot)	860	107	109
Healthy	842	105	106
Totale	3752	468	472

Table 4.6: Répartition des données
Type: Grape

Pour la classification des maladies de Pepper bell sachant qu'il dispose de 2 classes ,les données ont été répartie sur 3 ensembles : 80 % des images sont utilisées pour l'entraînement du modèle, 10% pour la validation et 10 % pour le test. Le tableau 4.7 illustre cette répartition.

	Entraînement 80%	Validation 10%	Test 10%
Bacterial spot	1178	147	148
Healthy	1181	147	149
Totale	2359	294	297

Table 4.7: Répartition des données
Type: Pepper bell

4.5.2.2 Protocole 02

Dans ce protocole, nous avons utilisé un seul modèle pour la classification des 19 classes à la fois , en utilisant la répartition illustrée dans le tableau 4.8.

Type d'aliment	Maladie associée	Entraînement 80%	Validation 10%	Test 10%
Grape	Black rot	944	118	118
	Esca (Black Measles)	1106	138	139
	Leaf blight (Isariopsis Leaf Spot)	860	107	109
	Healthy	800	100	100
Pepper bell	Bacterial spot	1178	147	148
	Healthy	1181	147	149
Potato	Early blight	800	100	100
	Late blight	800	100	100
	Healthy	799	100	101
Tomato	Bacterial spot	1701	212	214
	Early blight	1901	237	239
	Late blight	1892	236	238
	Leaf Mold	1884	235	236
	Septoria leaf spot	1749	218	220
	Spider mites Two-spotted spider mite	1744	218	218
	Target Spot	1852	231	233
	Yellow Leaf Curl Virus	1960	245	245
	Mosaic virus	1848	231	231
	Healthy	1930	241	242
Total	**19**	**26971**	**3366**	**3386**

Table 4.8: Répartition des données des 19 classes

4.5.3 Entraînement

La phase d'entraînement des modèles CNN se distingue par plusieurs paramètres essentiels. Tout d'abord, la taille des images d'entrée est fixée à 256×256. Avant de passer ces images au modèle, elles sont toutes normalisées entre 0 et 1 afin que les modèles puissent extraire les caractéristiques appropriées. Ensuite, tous les modèles CNN utilisent une architecture convolutive, comprenant 5 couches Conv2D et 5 couches MaxPooling2D comme montré dans la figure 4.7. Cela est suivi par une couche d'aplatissement (Flatten Layer) qui transforme les matrices 2D en un vecteur 1D (4608,1), préparant ainsi les données pour le bloc de classification constitué de deux couches Dense. La couche de sortie a un nombre de neurones correspondant au nombre de classes de chaque modèle.

Chaque paramètre est soigneusement sélectionné et configuré pour maximiser les performances des modèles.

- La fonction d'activation utilisée est **ReLU** pour les couches cachées et la fonction d'activation **Softmax** est utilisée pour la couche de sortie, permettant une classification multi-classe.
- Le taux d'apprentissage est fixé à **0,001**. L'optimisation est réalisée avec l'**optimiseur Adam**, choisi pour sa robustesse et sa capacité à gérer les variations des données.
- Le nombre d'époques varie de **20 à 30**, permettant un apprentissage profond tout en évitant le surapprentissage. La **taille du batch** est de **32**, équilibrant la vitesse d'entraînement et la stabilité des mises à jour des gradients.

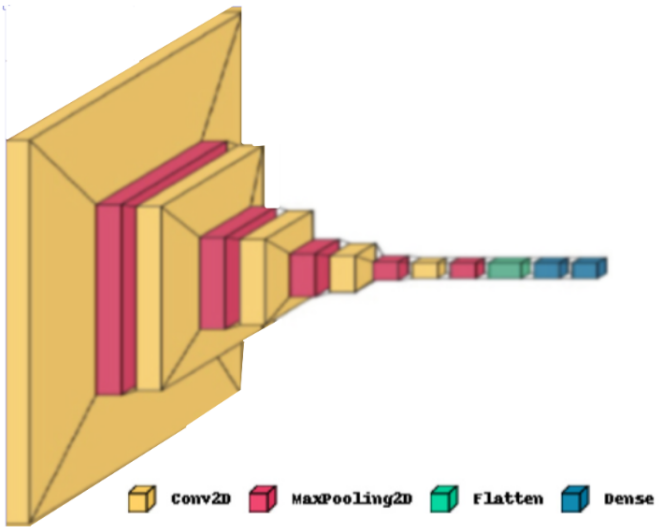

Figure 4.7: architecture de notre Modele CNN

Le tableau 4.9 Résume les Architectures et les paramètres d'entraînement des modèles CNN.

Architecture des Modèles CNN						
	Détection de type de plante	**Grape**	**Pepper bell**	**Potato**	**Tomato**	**Modele(CNN) de protocole 2**
Taille de l'image d'entrée en pixels	256×256					
Type	**Forme**					
Conv2D	(None,254,254,32)					
MaxPooling2D	(None,127,127,32)					
Conv2D	(None,125,125,64)					
MaxPooling2D	(None,62,62,64)					
Conv2D	(None,28,28,128)					
MaxPooling2D	(None,14,14,128)					
Conv2D	(None,12,12,128)					
MaxPooling2D	(None,6,6,128)					
Flatten	(None,4608)					
Dense	(None,64)					
Dense	(None,4)	(None,4)	(None,2)	(None,3)	(None,10)	(None,19)
Nombre de paramètres	572996	572996	572866	572931	573386	573971
Paramètres d'entraînement						
Fonction d'activation	Relu, sortie (Softmax)					
Taux d'apprentissage	0.001					
Epochs	20	30	20	30	25	20
Batch Size	32					
Optimiseur	Adam					

Table 4.9: Architecture et Paramètres d'entraînement adoptés

4.5.3.1 CNN en tant qu'extracteur de caractéristiques

Dans notre étude de classification des maladies des plantes, nous avons testé des modèles hybrides constitués d'un descripteur (extracteur de caractéristiques) et d'un classifieur. Nous avons choisi d'utiliser un CNN comme extracteur de caractéristiques. Ce choix a été motivé par la simplicité de l'architecture des CNN, ce qui nous permet d'apporter des améliorations simples et efficaces, comme illustré dans la figure 4.8.

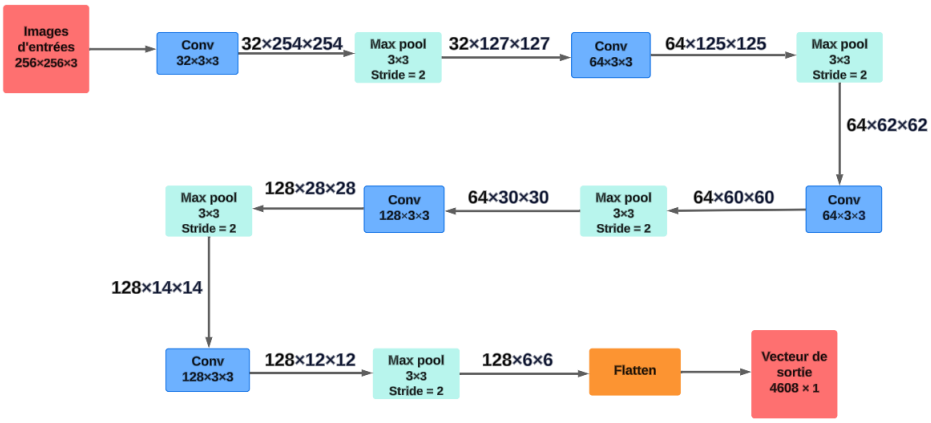

Figure 4.8: CNN en tant qu'extracteur de caractéristiques

Comme on peut le constater d'après la figure 4.8, il s'agit d'une architecture relativement simple, composée de

- 5 couches de convolution.
- 5 couches de max-pooling.
- 1 couche Flatten (fournissant le vecteur des caractéristiques représentatif pour chaque image).

Pour l'utilisation de ce descripteur on normalise les niveaux de gris faisant passer nos niveaux de gris de [0,255] à [0,1]. On a en sortie de ce descripteur un vecteur caractéristique d'une taille de 4608 éléments.
À la suite de ce descripteur on a testé 3 classifieurs : KNN, SVM et Random Forest telle que dans la prochaine section on va décrire la méthode pour le choix optimal des paramètres de ces classifieurs basée sur la validation croisée et la recherche en grille.

4.5.3.2 Choix des paramètres des classifieurs utilisés

Dans cette section on va décrire la façon de l'entraînement et du paramétrage de nos classifieurs. Cette étape est cruciale car elle détermine comment nos classifieurs vont apprendre à partir des caractéristiques extraites en utilisant l'extracteur des caractéristqiue que nous avons décrire dans la section précedent.
Cependant, la performance d'un classifieur dépend fortement de ses paramètres, que nous devons donc ajuster soigneusement. nous dans notre travaille pour l'entraînement des classifieur on a utiliser une combinaison de recherche en grille et de validation croisée qu'on va détailler dans la section prochaine .

4.5.3.3 Recherche en grille(GridSearch)

La méthode de recherche en grille, parfois appelée "Grid Search", est une technique d'optimisation des hyperparamètres. Elle consiste à définir une grille d'hyperparamètres potentiels et à évaluer systématiquement la performance du modèle pour chaque combinaison d'hyperparamètres dans cette grille. Cela nous permet de trouver la combinaison d'hyperparamètres qui donne les meilleurs résultats pour nos classifieurs.
Dans le code, la grille d'hyperparamètres peut être définie en créant un dictionnaire où chaque clé est le nom d'un hyperparamètre et chaque valeur est une liste de valeurs à explorer.
Voici la forme standard d'une grille d'hyperparamètres.

```
Param_Grid_ = {
    'paramètre1': [valeur1, valeur2, valeur3],
    'paramètre2': [valeur1, valeur2, valeur3,valeur4],
    'paramètre3': [valeur1, valeur2],
    'paramètre4': [valeur1, valeur2, valeur3]
}
```

Le tableau 4.10 représente les grilles d'hyperparamètres que nous avons utilisées pour trouver les hyperparamètres optimaux pour nos trois classifieurs.

Classifieur	Paramètre	Valeurs possibles
SVM	'kernel'	['linear', 'poly', 'rbf']
	'C'	[0.1, 1,3, 10]
	'gamma'	[0.001, 0.0001]
	'decision_function_shape'	['ovo', 'ovr']
KNN	'n_neighbors'	[3, 5, 7, 9, 11]
	'metric'	['euclidean', 'manhattan']
Random Forest	'max_depth'	[None]
	'max_features'	[1, 3, 10]
	'min_samples_split'	[1, 3, 10]
	'min_samples_leaf'	[1, 3, 10]
	'criterion'	['gini', 'entropy']
	'n_estimators'	[10, 20, 50, 100]

Table 4.10: Exemples des grilles d'hyperparamètres des classifieurs

4.5.3.4 Validation croisée (cross validation)

La validation croisée est une technique d'évaluation des classifieurs. Elle consiste à diviser les données d'entraînement en K sous-ensembles(plis). Le modèle est formé sur K-1 de ces sous-ensembles et évalué sur le sous-ensemble restant. Ce processus est répété K fois, chaque sous-ensemble étant utilisé une fois comme données de validation. La moyenne des résultats de ces K évaluations produit une mesure finale de performance du modèle. La figure 4.9 illustre une validation croisée avec K=5. Ce processus est répété pour différents hyperparamètres. Les moyennes des performances des modèles sont comparés, et un modèle est sélectionné. Ce modèle est ensuite testé sur l'ensemble de données de test mis de côté.

La validation croisée est particulièrement utile lorsque les données d'apprentissage sont limitées, car elle utilise efficacement toutes les observations pour évaluer le modèle.

La combinaison de la recherche en grille et de la validation croisée, appelée "Grid Search Cross Validation", est une approche robuste pour l'optimisation des hyperparamètres et l'évaluation des modèles.

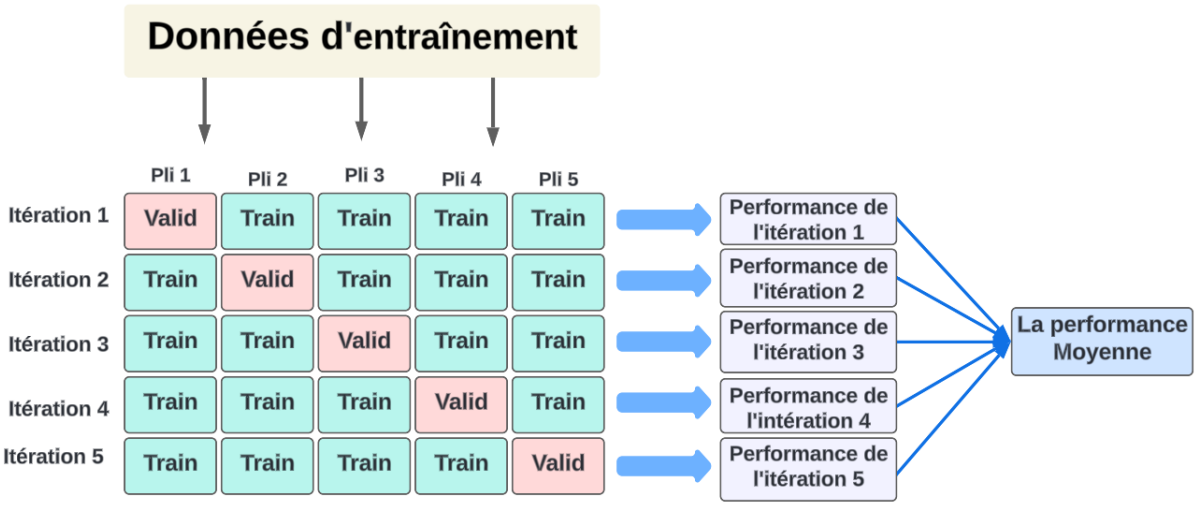

Figure 4.9: Principe de la validation croisée avec 5 plis

4.5.3.5 Choix optimal des paramètres des classifieurs utilisés

Le tableau 4.11 résume les hyperparamètres optimales pour nos classifieurs obtenu en utilisant "Grid Search Cross Validation".

Classifieur	Hyperparamètres
SVM	'kernel' =['rbf']
	'C' =[3]
	'gamma' =[0.001]
	'decision_function_shape' = ['ovr']
KNN	"n_neighbors"=[5]
	"metric"=['manhattan']
Random Forest	"max_depth"=[None]
	"max_features" =[3]
	"min_samples_split"=[3]
	"min_samples_leaf" =[1]
	"criterion" =['gini']
	"n_estimators" =[100]

Table 4.11: Les hyperparamètres optimaux des classifieurs

4.6 Performances atteintes

Dans un premier temps, nous allons évaluer les différents modèle adoptés pour le diagnostic des maladies des plantes selon les 2 protocoles présentés précédemment.

4.6.1 Protocole 1

Dans ce premier protocole, nous procédons d'abord à une classification par type de plante avant de passer à la classification de la maladie présente sur la feuille, en fonction du type de plante identifié. Une fois le type de plante détecté, le modèle de classification approprié pour la maladie est sélectionné. Plusieurs critères d'évaluation sont pris en compte, notamment le temps nécessaire à l'entraînement du modèle et le temps d'inférence pour une seule image. En effet, pour affirmer qu'un modèle est performant, il faut considérer son implémentation dans une application mobile. L'évaluation quantitative de la performance des différents modèles est principalement basée sur l'exactitude (accuracy) dans les ensembles d'entraînement, de validation et de test. Le tableau 4.12 résume les performances obtenues lors de l'entraînement des différents modèles constituant ce protocole.

Détection du type de plante						
Descripteur	**Classifieur**	**Exactitude d'entraînement (%)**	**Exactitude de validation (%)**	**Exactitude de test (%)**	**Temps d'entraînement**	**Temps d'inférence d'une seule image**
CNN	**Couche entièrement connectée**	99.58	98.13	**98.49**	23 min 42 s (20 epochs)	**30 ms**
Grape						
Descripteur	**Classifieur**	**Exactitude d'entraînement (%)**	**Exactitude de validation (%)**	**Exactitude de test (%)**	**Temps d'entraînement**	**Temps d'inférence d'une seule image**
CNN	**Couche entièrement connectée**	100	98.93	**98.73**	4 min 36 s (30 epochs)	**31 ms**
Pepper bell						
Descripteur	**Classifieur**	**Exactitude d'entraînement (%)**	**Exactitude de validation (%)**	**Exactitude de test (%)**	**Temps d'entraînement**	**Temps d'inférence d'une seule image**
CNN	**Couche entièrement connectée**	100	100	**99.66**	2 min 4 s (20 epochs)	**31 ms**
Potato						
Descripteur	**Classifieur**	**Exactitude d'entraînement (%)**	**Exactitude de validation (%)**	**Exactitude de test (%)**	**Temps d'entraînement**	**Temps d'inférence d'une seule image**
CNN	**Couche entièrement connectée**	100	97.33	**98.67**	3 min 10 s (30 epochs)	**29 ms**
Tomato						
Descripteur	**Classifieur**	**Exactitude d'entraînement (%)**	**Exactitude de validation (%)**	**Exactitude de test (%)**	**Temps d'entraînement**	**Temps d'inférence d'une seule image**
CNN	**Couche entièrement connectée**	98.63	95.27	**95.51**	19 min 45 s (25 epochs)	**36 ms**

Table 4.12: Performances des modèles de protocole 1

Pour ce protocole, nous utilisons cinq modèles : un pour la détection du type de plante et quatre pour la classification des maladies associées à chaque type de plante parmi les quatre disponibles. Nous commençons par appliquer les données de test du tableau 4.8 au modèle de détection du type de plante. Ensuite, nous structurons un dictionnaire contenant quatre clés, chacune représentant un modèle dédié à la classification des maladies spécifiques détectées chez chaque type de plante. Le tableau suivant synthétise les résultats obtenus après l'exécution simultanée de ces cinq modèles pour classifier 19 classes de maladies.

Résultats de Protocole 1			
Modèle 1	**Modèle 2**	**Exactitude de test (%)**	**Temps d'inférence d'une seule image**
Détection du type de plante	**Détection de la maladies**	**95.72**	1.7 s

En analysant la matrice de confusion 4.10 qui représente le protocole 1, nous constatons que parmi les 3386 image de teste 145 images sont mal classées. Par exemple:

- 10 plantes de la classe *Tomato-Late-blight* sont incorrectement classées dans la classe *Tomato-Early-blight* telle que le modèle effectue une confusion entre c'est deux classe.

- 15 plantes de la classe *Tomato-Spider mites Two-spotted spider mite* sont mal classées pour la classe *Tomato-Target-spot* telle que le modèle effectue aussi la confusion entre c'est deux classe.

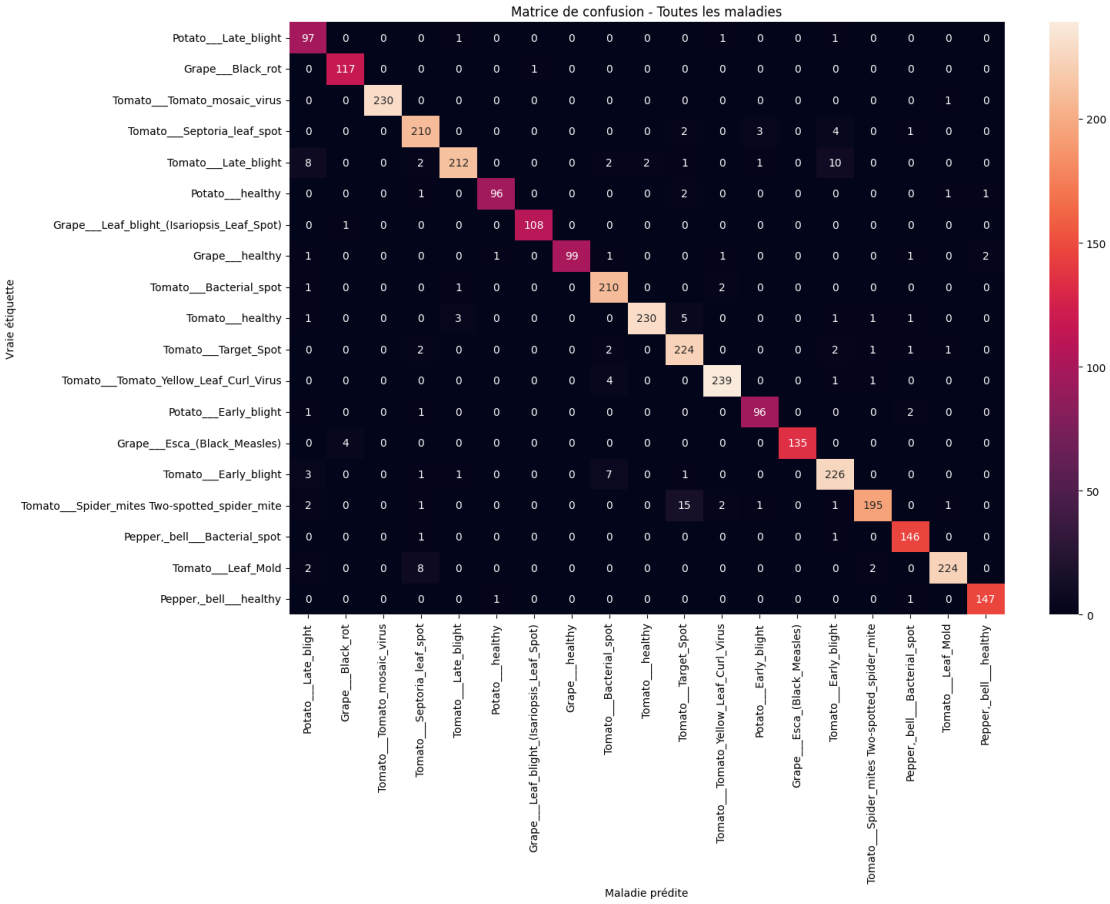

Figure 4.10: Matrice de confusion de protocole 1

La figure 4.11 représente des exemples de classification des maladies des plantes selon le protocole 1 .

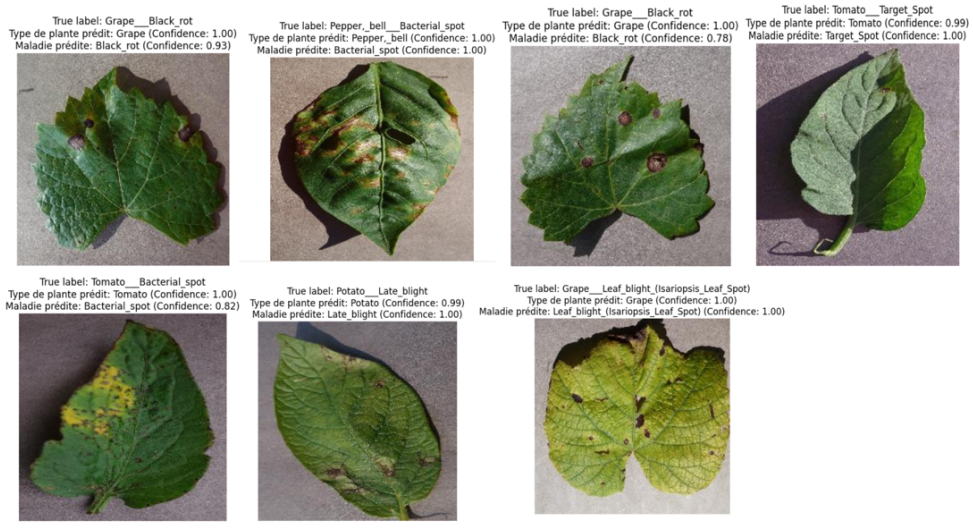

Figure 4.11: Des exemples de classification des images selon le protocole 1

Cette observation est soutenue par les valeurs de précision et de recall fournies dans le tableau 4.13. Une précision élevée indique que le modèle réduit au minimum le nombre de faux positifs, tandis qu'un recall élevé signifie que le modèle maximise le nombre de vrais positifs. Pour une évaluation plus complète de la performance du modèle, nous examinons également le score F1, plus il est élevé, plus le modèle est considéré comme performant.

	Précision (%)	Recall (%)	F1-score (%)	Nombre d'images
Grape_Black rot	95.90	99.15	97.50	118
Grape_Esca (Black Measles)	100.00	97.12	98.54	139
Grape_Leaf blight (Isariopsis Leaf Spot)	99.08	99.08	99.08	109
Grape_Healthy	100.00	93.40	96.59	106
Pepper bell_Bacterial spot	95.42	98.65	97.01	148
Pepper bell_Healthy	98.00	98.66	98.33	149
Potato_Early blight	95.05	96.00	95.52	100
Potato_Late blight	83.62	97.00	89.81	100
Potato_Healthy	97.96	95.05	96.48	101
Tomato_Bacterial spot	92.92	98.13	95.45	214
Tomato_Early blight	91.50	94.56	93.00	239
Tomato_Late blight	97.25	89.08	92.98	238
Tomato_Leaf Mold	98.25	94.92	96.55	236
Tomato_Septoria leaf spot	92.51	95.45	93.96	220
Tomato_Spider mites Two-spotted spider mite	97.50	89.45	93.30	218
Tomato_Target Spot	89.60	96.14	92.75	233
Tomato_Yellow Leaf Curl Virus	97.55	97.55	97.55	245
Tomato_Mosaic virus	100	99.57	99.78	231
Tomato_Healthy	99.14	95.04	97.05	242
accuracy			95.72	3386
macro avg	95.86	96.00	95.86	3386
weighted avg	95.88	95.72	95.73	3386

Table 4.13: Précision, recall et F1-score par classe du protocole 1

4.6.2 Protocole 2

Pour ce deuxième protocole, nous avons abordé la classification des maladies en combinant les quatre types de plantes pour créer un seul modèle capable de distinguer 19 classes. Pour effectuer cette classification en 19 classes, nous avons testé quatre modèles. Plusieurs critères d'évaluation sont pris en compte, notamment le temps nécessaire à l'entraînement du modèle et le temps d'inférence pour une seule image. L'évaluation quantitative de la performance des différents modèles est principalement basée sur l'exactitude (accuracy) dans les ensembles d'entraînement, de validation et de test. Le tableau 4.14 résume les performances obtenues lors de l'entraînement des différents modèles testés dans ce protocole.

Descripteur	Classifieur	Exactitude d'entraînement (%)	Exactitude de validation (%)	Exactitude de test (%)	Temps d'entraînement	Temps d'inférence d'une seule image
CNN	Couche entièrement connectée	98.37	95.07	94.42	24min 35 s (20 epochs)	0.042 s
CNN	SVM	99.49	96.05	95.75	42 min 19 s	0.002 s
CNN	KNN	99.28	96.17	95.30	31 min 25s	0.004s
CNN	Random Forest	100.00	95.90	95.42	36 min 49 s	0.009s

Table 4.14: Résultats de la classification sur 19 classe

D'après le tableau 4.14, qui présente les performances des modèles développés pour la classification des 19 classes à la fois, nous constatons que le modèle CNN+SVM offre l'exactitude la plus élevée 95,75% et possède également le temps d'inférence pour une seule image le plus court (0,002 s), ce qui est très favorable. Nous observons également que, de manière générale,

les modèles hybrides fournissent de meilleurs résultats que le modèle CNN bout à bout (celui qui effectue la classification en utilisant des couches entièrement connectées).

Performance Modèle CNN + SVM :
En analysant la matrice de confusion (voir la figure 4.12) de notre modèle le plus performant (CNN+SVM), nous constatons que parmi les 3386 images testées, 144 images sont mal classées. Le modèle effectue des confusions entre certaines classes. Par exemple :

- 10 plantes de la classe *Tomato-Spider-Mites-Two-Spotted-Spider-Mite* sont incorrectement classées dans la classe *Tomato-Target-Spot*.
- 7 plantes de la classe *Tomato-Target-Spot* sont incorrectement classées dans la classe *Tomato-Spider-Mites-Two-Spotted-Spider-Mite*.

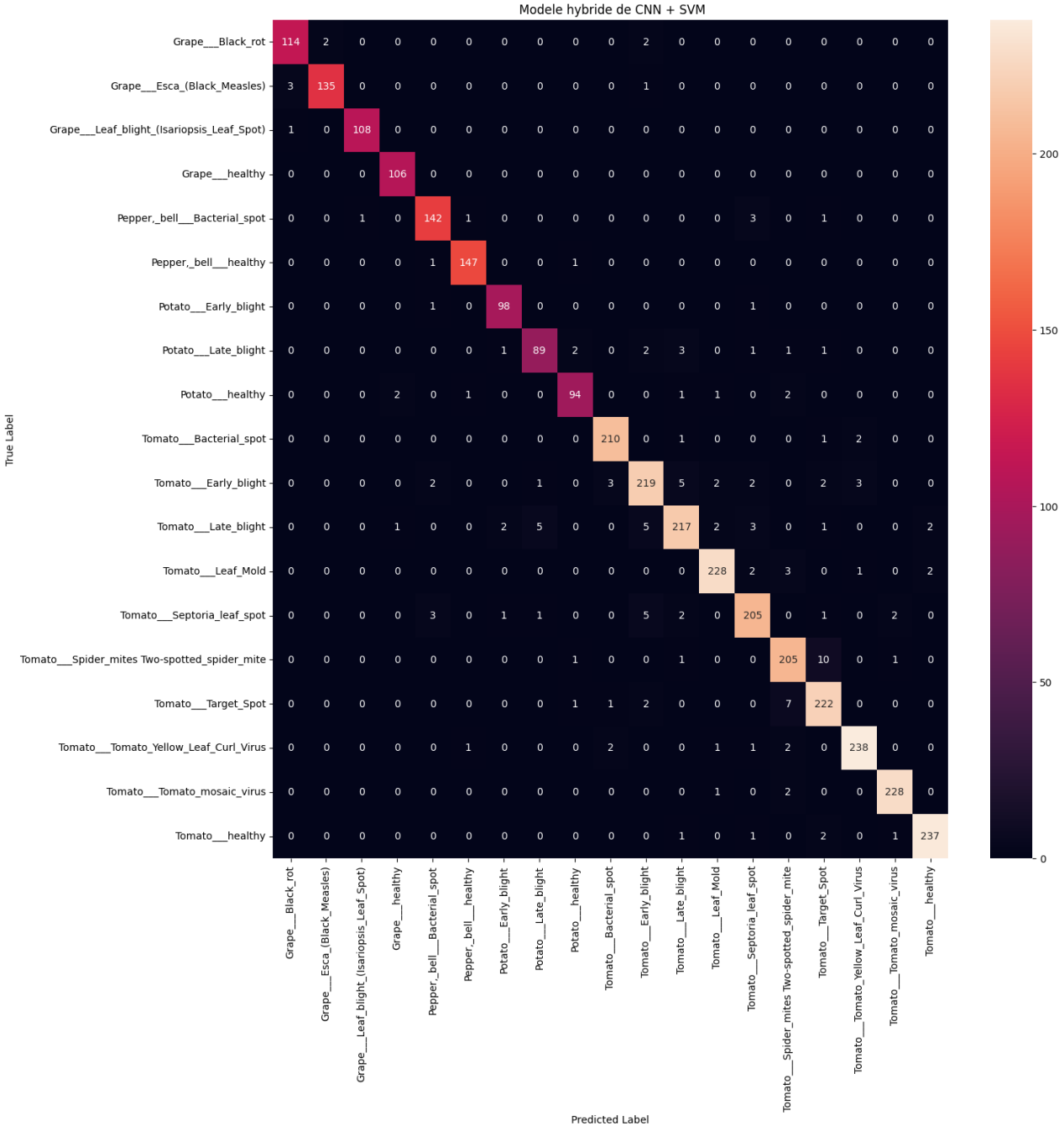

Figure 4.12: Matrice de confusion CNN +SVM

Cette observation est soutenue par les valeurs de précision et de recall fournies dans le tableau 4.15. Une précision élevée indique que le modèle réduit au minimum le nombre de faux positifs, tandis qu'un recall élevé signifie que le modèle maximise le nombre de vrais positifs. Pour une évaluation plus complète de la performance du modèle, nous examinons également le score F1 , plus il est élevé plus le modèle est considéré comme performant.

	Précision (%)	Recall (%)	F1-score (%)	Nombre d'images
Grape_Black rot	96.61	96.61	96.61	118
Grape_Esca (Black Measles)	98.54	97.12	97.83	139
Grape_Leaf blight (Isariopsis Leaf Spot)	99.08	99.08	99.08	109
Grape_Healthy	97.25	100.00	98.60	106
Pepper bell_Bacterial spot	95.30	95.95	95.62	148
Pepper bell_Healthy	98.00	98.66	98.33	149
Potato_Early blight	96.08	98.00	97.03	100
Potato_Late blight	92.71	89.00	90.82	100
Potato_Healthy	94.95	93.07	94.00	101
Tomato_Bacterial spot	97.22	98.13	97.67	214
Tomato_Early blight	92.80	91.63	92.21	239
Tomato_Late blight	93.94	91.18	92.54	238
Tomato_Leaf Mold	97.02	96.61	96.82	236
Tomato_Septoria leaf spot	93.61	93.18	93.39	220
Tomato_Spider mites Two-spotted spider mite	92.34	94.04	93.18	218
Tomato_Target Spot	92.34	95.28	93.67	233
Tomato_Yellow Leaf Curl Virus	97.54	97.14	97.34	245
Tomato_Mosaic virus	98.28	98.70	98.49	231
Tomato_Healthy	98.34	97.93	98.14	242
accuracy			**95.75**	3386
macro avg	95.88	95.86	95.86	3386
weighted avg	95.75	95.75	95.74	3386

Table 4.15: Précision, recall et F1-score par classe du modèle CNN +SVM

La figure 4.13 présente l'exactitude des modèles des protocoles 1 et 2. Nous remarquons que le meilleur modèle du protocole 2 est le modèle (CNN+SVM) atteint une exactitude de 95,75%, tandis que le modèle représentant le protocole 1 a une exactitude de 95,72%. Bien que les deux exactitude soient très proches, le protocole 2 s'avère être le choix le plus performant en raison du temps d'inférence beaucoup plus court pour une seule image 0,002s pour le protocole 2 contre 1,7s pour le protocole 1. Ce paramètre est crucial pour que le système soit implémenté dans une application mobile. Afin d'améliorer davantage les performances de précision du protocole 2, nous avons exploré de nouveaux outils mathématiques, dont les résultats prometteurs seront présentés dans le chapitre suivant.

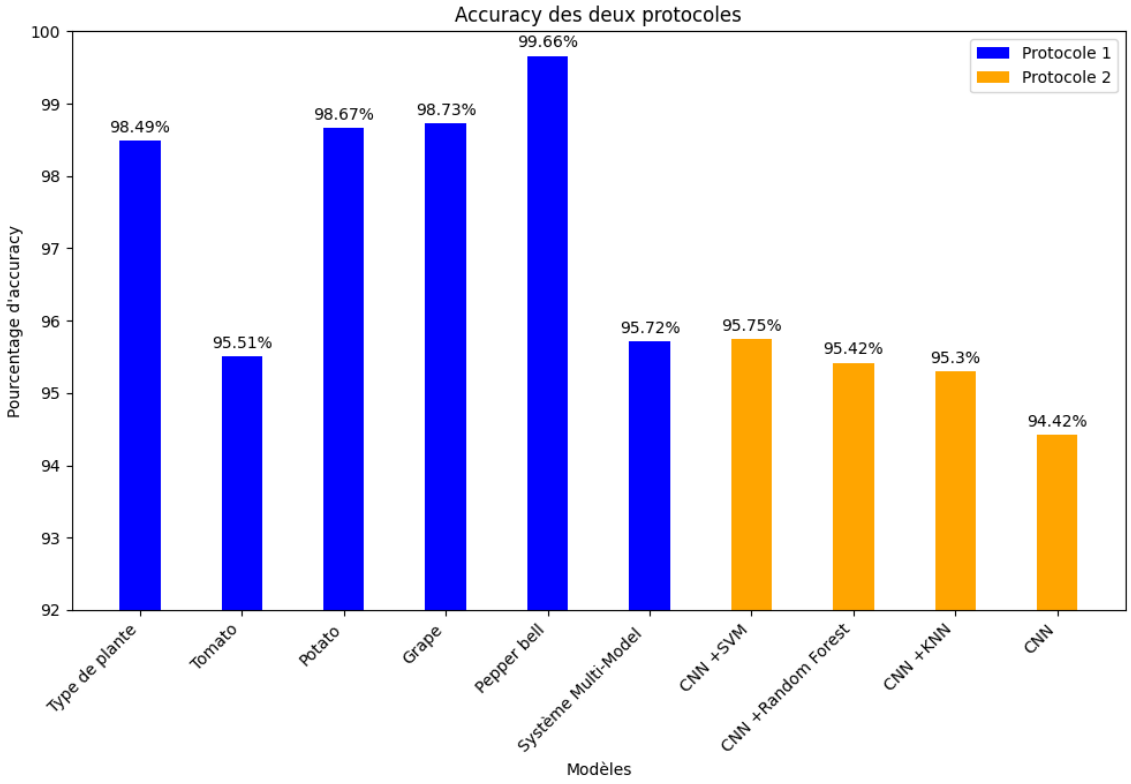

Figure 4.13: Comparaison des modèles des deux protocole

4.7 Conclusion

A travers ce chapitre, nous avons présenté les outils logiciels et matériels utilisés dans notre projet, en passant par la description de l'ensemble de données que nous avons utilisé, ainsi que les méthodes d'évaluation de nos différents modèles. Nous concluons par la présentation des différents modèles que nous avons développés, à savoir le modèle CNN bout à bout, ainsi que les trois modèles hybrides combinant un CNN comme extracteur de caractéristiques avec différents classifieurs (SVM, KNN et random forest). Nous présentons également les résultats obtenus avec chaque modèle En terminant ce chapitre, nous conclons que les modèles hybrides (CNN + classifieur) donnent de meilleurs résultats que ceux obtenus en utilisant un CNN bout à bout.

Dans le but d'améliorer les performances obtenues avec ces modèles ,dans le chapitre prochaine ,nous présentons une technique innovante d'amélioration des performances , à savoir les intégrales floues. Nous testerons deux types d'intégrales : l'intégrale de Choquet et celle de Sugeno. Pour déterminer précisément les mesures floues à attribuer aux différents modèles lors de l'utilisation de ces intégrales, nous recourrons à l'optimisation par essaim de particules (PSO) afin de trouver les mesures floues optimales.

CHAPTER 5

AMÉLIORATION DES PERFORMANCES

5.1 Introduction

Dans l'objectif d'améliorer les performances de notre modèle de classification de 19 classes à la fois , nous avons opté pour l'utilisation des intégrales floues. Nous avons testé deux types d'intégrales, à savoir l'intégrale de Sugeno et l'intégrale de Choquet, et nous avons utilisé une optimisation par essaim de particules pour déterminer de manière très précise les mesures floues optimales à utiliser pour chaque intégrale. Afin de trouver le modèle le plus performant, nous avons testé ces intégrales avec différentes combinaisons des quatre modèles que nous avions déjà développés dans le protocole 2. Nous présenterons en détail les résultats de ces combinaisons dans les sections suivantes.

5.2 Les combinaisons de l'intégrale de Sugeno avec PSO

Dans cette partie on va décrire les résultat obtenu on applicant l'intégrale de sugeno avec l'optimisation par essaim de particule pour les différents combianison des modèles qu'on dispose

5.2.1 CNN et CNN +SVM et CNN +KNN

Dans cette partie, nous allons décrire les performances obtenues en appliquant l'intégrale de Sugeno avec PSO sur trois modèles, à savoir les modèles CNN avec comme classifieur les couches entièrement connectées, le modèle CNN+SVM et le modèle CNN+KNN, comme montré dans la figure 5.1. Le tableau 5.1 résume les paramètres de l'optimisation par essaim de particules utilisés pour cette combinaison, à savoir les valeurs des positions initiales, la taille de l'essaim, la dimension de l'espace de recherche, l'intervalle de recherche, les valeurs des coefficients a_1 et a_2, l'inertie w, le nombre d'itérations, et le temps de recherche. Nous incluons également l'exactitude de chaque modèle avant l'application de l'intégrale ainsi que l'exactitude obtenue après l'application de l'intégrale de Sugeno avec PSO, et le temps d'inférence d'une image .

Paramètres de PSO pour (CNN , CNN + SVM et CNN + KNN)						
Valeur des positions initiales	Taille de l'essaim	Dimension de l'espace de recherche	L'intervalle de recherche	Valeurs des coefficients a_1, a_2 et inertie w	Nombre d'itérations	Temps de recherche
[0.65,0.5,0.4]	5	3	[0,1]	a_1=1, a_2=2, w =0.5	10	1h 33min
Sugeno intégrale avec PSO pour (CNN , CNN + SVM et CNN + KNN)						
Descripteur	**Classifieur**	**Exactitude des modèles (%)**	**Mesures floues optimales obtenues avec PSO**	**Exactitude obtenue avec Choquet et PSO (%)**	**Temps d'inférence d'une seule image**	
CNN	Couche entièrement connectée	94.42	0.38	97.46	0.5 s	
CNN	SVM	95.75	1			
CNN	KNN	95.30	0.93			

Table 5.1: Paramètres de modèle 1 de l'intégrale de sugeno avec PSO adoptée

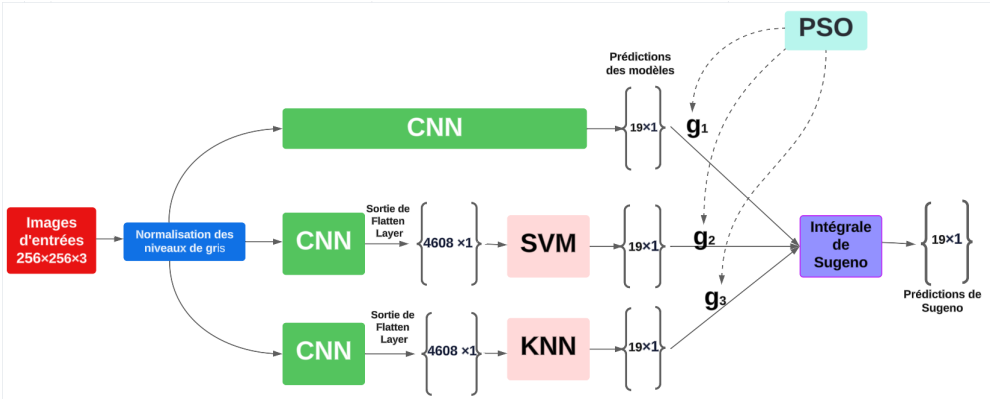

Figure 5.1: Sugeno avec PSO appliqué à CNN, CNN+SVM et CNN+KNN.

D'après le tableau 5.1 On remarque que le temps de recherche pour cette optimisation est très élevé(1h 33min), due au fait que l'entraînement de cette optimisation a été effectué en utilisant le CPU de Google Colaboratory.
Nous avons choisi une dimension de l'espace de recherche égale à 3 car pour cette combinaison, nous disposons de trois modèles. Il est donc nécessaire de déterminer la mesure floue à attribuer à chaque modèle.
On constate que cette intégrale a permis une amélioration de 1,7% par rapport au meilleur des modèles disponibles, ce qui est bon.
En analysant la matrice de confusion 5.2, on observe que parmi les 3386 images testées, 85 sont mal classées. Par exemple :

- 11 images de la classe *Tomato-target-spot* sont incorrectement classées comme *Tomato-spider-mites-two-spotted-spider-mite*.
- 6 images de la classe *Tomato-septoria-leaf-spot* sont incorrectement classées comme *Tomato-Early-blight*.
- 7 images de la classe *Tomato-Late-blight* sont incorrectement classées comme *Potato-late-blight*.

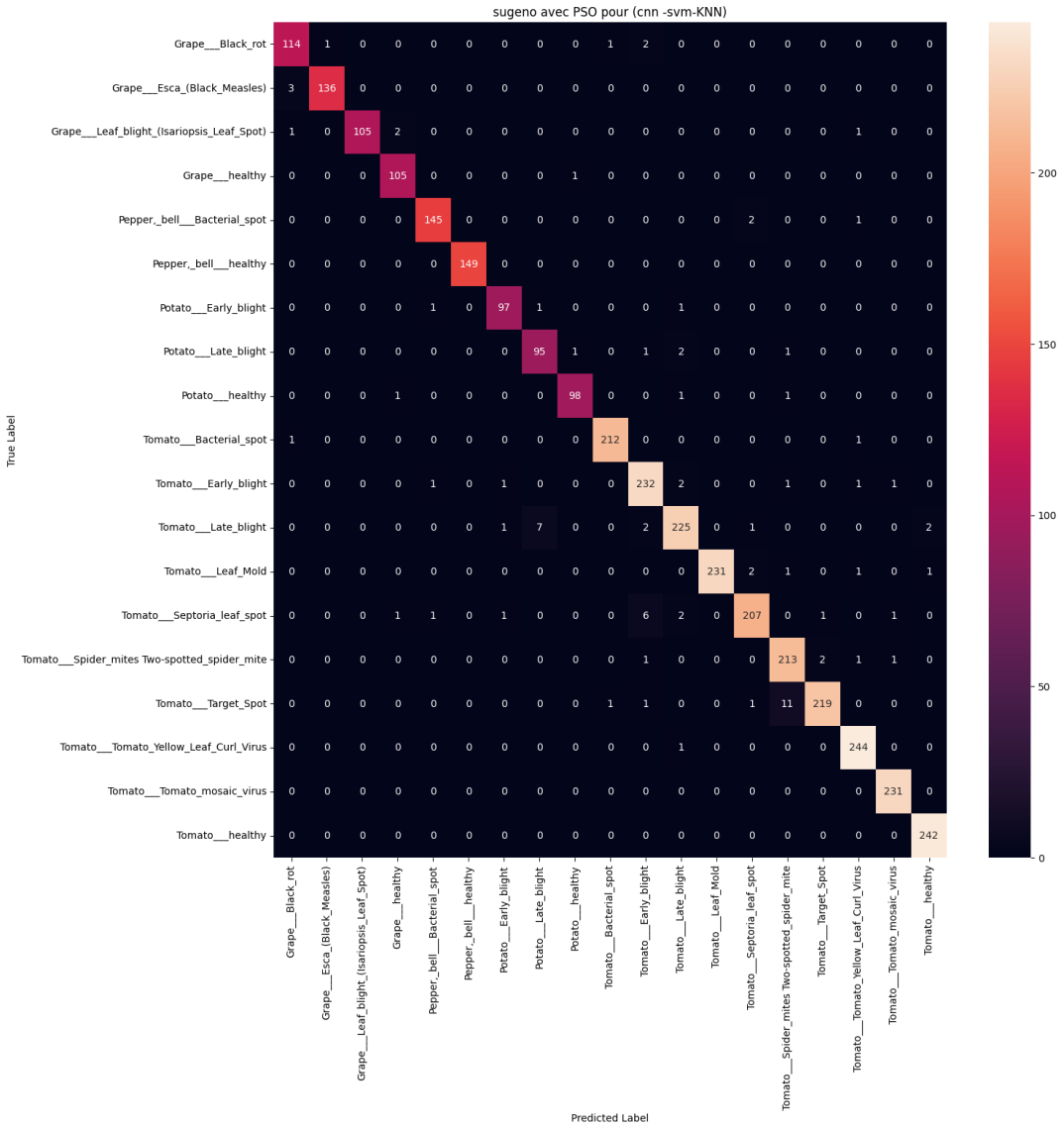

Figure 5.2: Matrice de confusion du modèle Sugeno avec PSO appliqué à CNN, CNN+SVM et CNN+KNN

Cette observation est soutenue par les valeurs de précision et de recall fournies dans le tableau 5.2 Une précision élevée indique que le modèle réduit au minimum le nombre de faux positifs, tandis qu'un recall élevé signifie que le modèle maximise le nombre de vrais positifs. Pour une évaluation plus complète de la performance du modèle, nous examinons également le score F1, plus il est élevé, plus le modèle est considéré comme performant.

	Précision (%)	Recall (%)	F1-score (%)	Nombre d'images
Grape_Black rot	95.80	96.61	96.20	118
Grape_Esca (Black Measles)	99.27	97.84	98.55	139
Grape_Leaf blight (Isariopsis Leaf Spot)	100.00	96.33	98.13	109
Grape_Healthy	96.33	99.06	97.67	106
Pepper bell_Bacterial spot	97.97	97.97	97.97	148
Pepper bell_Healthy	100.00	100.00	100.00	149
Potato_Early blight	97.00	97.00	97.00	100
Potato_Late blight	92.23	95.00	93.60	100
Potato_Healthy	98.00	97.03	97.51	101
Tomato_Bacterial spot	99.07	99.07	99.07	214
Tomato_Early blight	94.69	97.07	95.87	239
Tomato_Late blight	96.15	94.54	95.34	238
Tomato_Leaf Mold	100.00	97.88	98.93	236
Tomato_Septoria leaf spot	97.18	94.09	95.61	220
Tomato_Spider mites Two-spotted spider mite	93.42	97.71	95.52	218
Tomato_Target Spot	98.65	93.99	96.26	233
Tomato_Yellow Leaf Curl Virus	97.60	99.59	98.59	245
Tomato_Mosaic virus	98.72	100.00	99.35	231
Tomato_Healthy	98.78	100.00	99.38	242
accuracy			**97.46**	3386
macro avg	97.41	97.41	97.40	3386
weighted avg	97.49	97.46	97.46	3386

Table 5.2: Précision, recall et F1-score par classe du modèle Sugeno avec PSO appliqué à CNN, CNN+SVM et CNN+KNN

5.2.2 CNN et CNN +SVM et CNN +Random Forest

Dans cette partie, nous allons décrire les performances obtenues en appliquant l'intégrale de Sugeno avec PSO sur trois modèles, à savoir les modèles CNN avec comme classifieur les couches entièrement connectées, le modèle CNN+SVM et le modèle CNN+Random Forest, comme montré dans la figure 5.3. Le tableau 5.3 résume les paramètres de l'optimisation par essaim de particules utilisés pour cette combinaison, à savoir les valeurs des positions initiales, la taille de l'essaim, la dimension de l'espace de recherche, l'intervalle de recherche, les valeurs des coefficients a_1 et a_2, l'inertie w, le nombre d'itérations, et le temps de recherche . Nous incluons également l'exactitude de chaque modèle avant l'application de l'intégrale ainsi que l'exactitude obtenue après avoir appliqué l'intégrale de Sugeno avec PSO, et le temps d'inférence d'une image.

Paramètres de PSO pour (CNN et CNN +SVM et CNN +Random Forest)						
Valeur des positions initiales	Taille de l'essaim	Dimension de l'espace de recherche	L'intervalle de recherche	Valeurs des coefficients a_1, a_2 et inertie w	Nombre d'itérations	Temps de recherche
[0.65,0.5,0.4]	5	3	[0,1]	a_1=1, a_2=2 , w=0.5	10	1h 36 min
Sugeno intégrale avec PSO pour (CNN , CNN + SVM et CNN + Random Forest)						
Descripteur	Classifieur	Exactitude des modèles (%)	Mesures floues optimales obtenues avec PSO	Exactitude obtenue avec Choquet et PSO (%)	Temps d'inférence d'une seule image	
CNN	Couche entièrement connectée	94.42	0.69	**97.55**	0.5 s	
CNN	SVM	95.75	0.79			
CNN	Random Forest	95.42	0.78			

Table 5.3: Paramètres de modèle 2 de l'intégrale de sugeno avec PSO adoptée

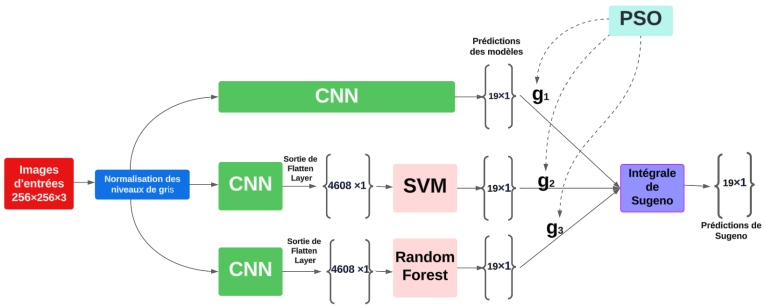

Figure 5.3: Sugeno avec PSO appliqué à CNN, CNN+SVM et CNN+Random Forest.

D'apès le tableau5.3 On constate que cette intégrale a permis une amélioration de 1,8% par rapport au meilleur des modèles disponibles, ce qui est bon.
En analysant la matrice de confusion 5.4, on observe que parmi les 3386 images testées, 83 sont mal classées. Par exemple :

- 9 images de la classe *Tomato-target-spot* sont incorrectement classées comme *Tomato-spider-mites-two-spotted-spider-mite*.
- 10 images de la classe *Tomato-Late-blight* sont incorrectement classées comme *Potato-late-blight*.

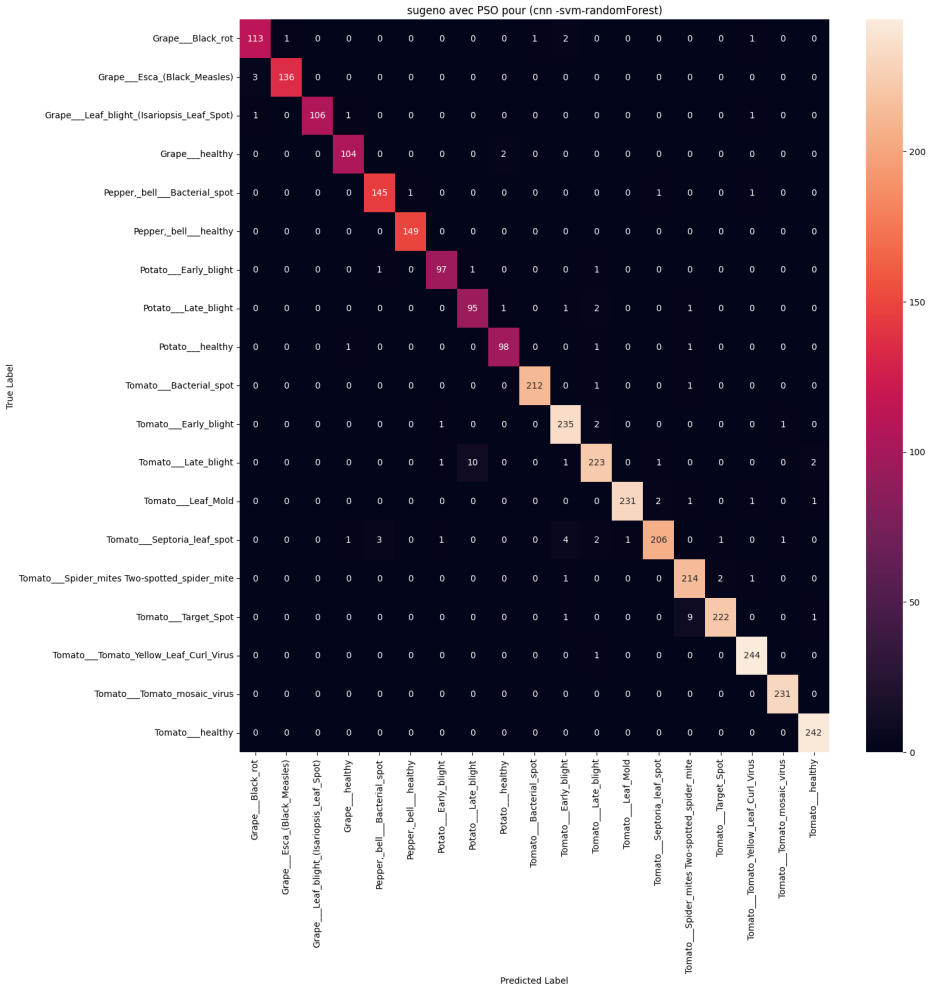

Figure 5.4: Matrice de confusion du modèle Sugeno avec PSO appliqué à CNN, CNN+SVM et CNN+Random Forest

Cette observation est soutenue par les valeurs de précision et de recall fournies dans le tableau 5.4. Une précision élevée indique que le modèle réduit au minimum le nombre de faux positifs, tandis qu'un recall élevé signifie que le modèle maximise le nombre de vrais positifs. Pour une évaluation plus complète de la performance du modèle, nous examinons également le score F1, plus il est élevé, plus le modèle est considéré comme performant.

	Précision (%)	Recall (%)	F1-score (%)	Nombre d'images
Grape_Black rot	96.58	95.76	96.17	118
Grape_Esca (Black Measles)	99.27	97.84	98.55	139
Grape_Leaf blight (Isariopsis Leaf Spot)	100.00	97.25	98.60	109
Grape_Healthy	97.20	98.11	97.65	106
Pepper bell_Bacterial spot	97.32	97.97	97.64	148
Pepper bell_Healthy	99.33	100.00	99.67	149
Potato_Early blight	97.00	97.00	97.00	100
Potato_Late blight	89.62	95.00	92.23	100
Potato_Healthy	97.03	97.03	97.03	101
Tomato_Bacterial spot	99.53	99.07	99.30	214
Tomato_Early blight	95.92	98.33	97.11	239
Tomato_Late blight	95.71	93.70	94.69	238
Tomato_Leaf Mold	99.57	97.88	98.72	236
Tomato_Septoria leaf spot	98.10	93.64	95.81	220
Tomato_Spider mites Two-spotted spider mite	94.27	98.17	96.18	218
Tomato_Target Spot	98.67	95.28	96.94	233
Tomato_Yellow Leaf Curl Virus	97.99	99.59	98.79	245
Tomato_Mosaic virus	99.14	100.00	99.57	231
Tomato_Healthy	98.37	100.00	99.18	242
accuracy			**97.55**	3386
macro avg	97.40	97.45	97.41	3386
weighted avg	97.58	97.55	97.55	3386

Table 5.4: Précision, recall et F1-score par classe du modèle Sugeno avec PSO appliqué à CNN, CNN+SVM et CNN+Random Forest

5.2.3 CNN et CNN +KNN et CNN +Random Forest

Dans cette partie, nous allons décrire les performances obtenues en appliquant l'intégrale de Sugeno avec PSO sur trois modèles, à savoir les modèles CNN avec comme classifieur les couches entièrement connectées, le modèle CNN+KNN et le modèle CNN+Random Forest, comme montré dans la figure 5.5. Le tableau 5.5 résume les paramètres de l'optimisation par essaim de particules utilisés pour cette combinaison, à savoir les valeurs des positions initiales, la taille de l'essaim , la dimension de l'espace de recherche, l'intervalle de recherche, les valeurs des coefficients a_1 et a_2, l'inertie w, le nombre d'itérations, et le temps de recherche. Nous incluons également l'exactitude de chaque modèle avant l'application de l'intégrale ainsi que l'exactitude obtenue après avoir appliqué l'intégrale de Sugeno avec PSO, et le temps d'inférence d'une image.

Paramètres de PSO pour (CNN , CNN + KNN et CNN + Random Forest)						
Valeur des positions initiales	Taille de l'essaim	Dimension de l'espace de recherche	L'intervalle de recherche	Valeurs des coefficients a_1, a_2 et inertie w	Nombre d'itérations	Temps de recherche
[0.45,0.5,0.6]	5	3	[0,1]	a_1=1, a_2=2 , w=0.5	10	1h 29 min
Sugeno intégrale avec PSO pour (CNN , CNN + KNN et CNN + Random Forest)						
Descripteur	Classifieur	Exactitude des modèles (%)	Mesures floues optimales obtenues avec PSO	Exactitude obtenue avec Choquet et PSO (%)	Temps d'inférence d'une seule image	
CNN	Couche entièrement connectée	94.42	0.21	**97.31**	0.5 s	
CNN	KNN	95.30	0.93			
CNN	Random Forest	95.42	1			

Table 5.5: Paramètres de modèle 3 de l'intégrale de sugeno avec PSO adoptée

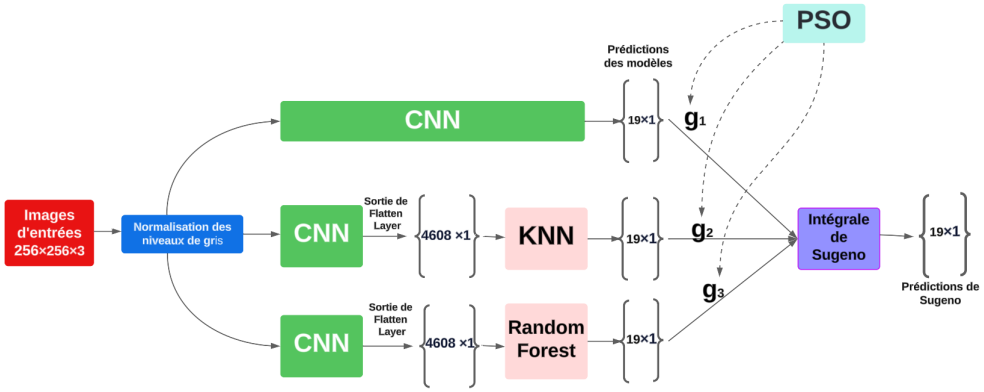

Figure 5.5: Sugeno avec PSO appliqué à CNN, CNN+KNN et CNN+Random Forest.

D'apès le tableau5.5 On constate que cette intégrale a permis une amélioration de 1,9% par rapport au meilleur des modèles disponibles, ce qui est bon.
En analysant la matrice de confusion 5.6, on observe que parmi les 3386 images testées, 91 sont mal classées. Par exemple :

- 13 images de la classe *Tomato-target-spot* sont incorrectement classées comme *Tomato-spider-mites-two-spotted-spider-mite*.
- 13 images de la classe *Tomato-Late-blight* sont incorrectement classées comme *Potato-late-blight*.
- 5 images de la classe *Tomato-Spectoria-Leaf-spot* sont incorrectement classées comme *Tomato-Early-blight*

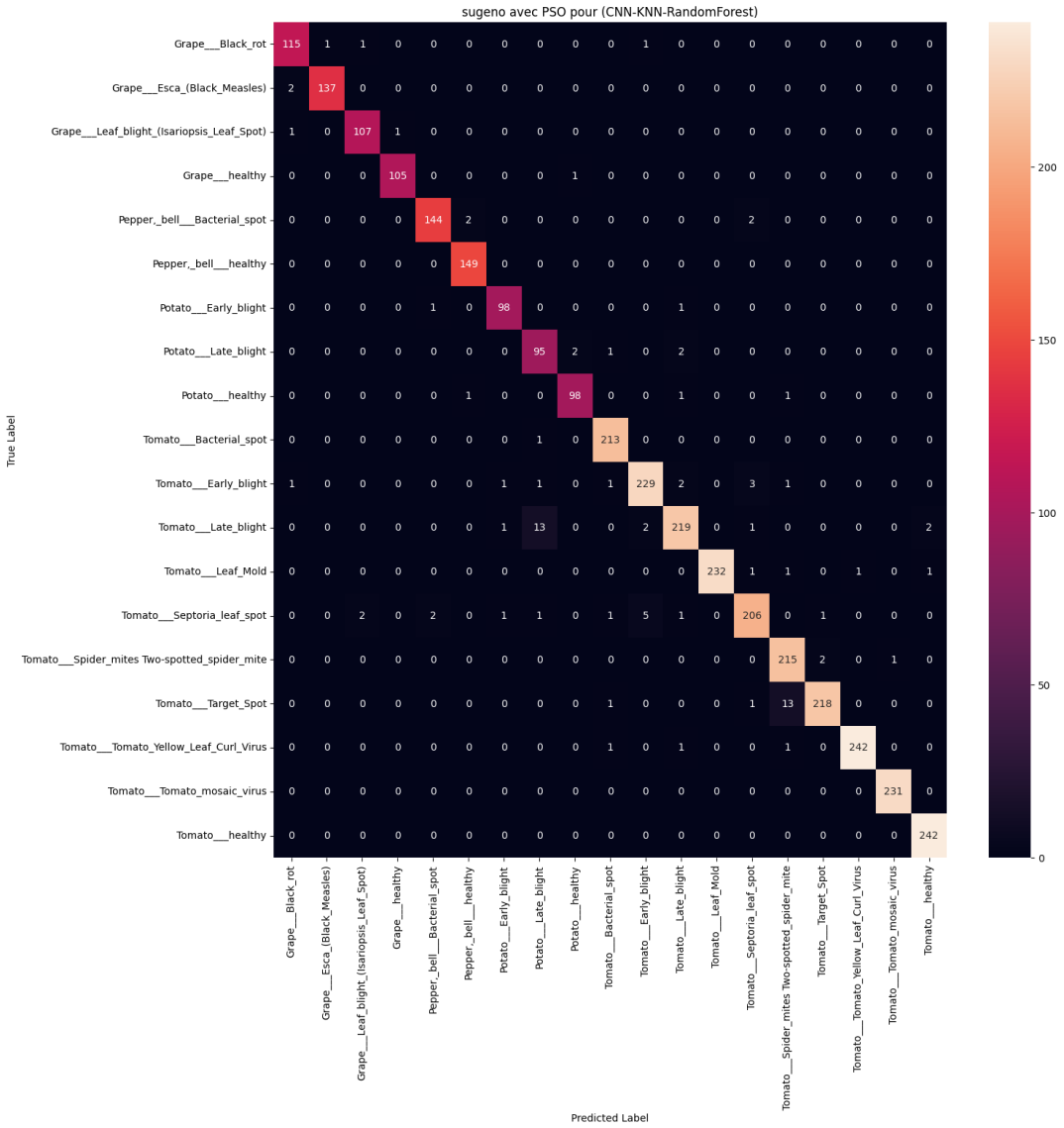

Figure 5.6: Matrice de confusion du modèle Sugeno avec PSO appliqué à CNN, CNN+KNN et CNN+Random Forest

Cette observation est soutenue par les valeurs de précision et de recall fournies dans le tableau 5.6. Une précision élevée indique que le modèle réduit au minimum le nombre de faux positifs, tandis qu'un recall élevé signifie que le modèle maximise le nombre de vrais positifs. Pour une évaluation plus complète de la performance du modèle, nous examinons également le score F1, plus il est élevé, plus le modèle est considéré comme performant.

	Précision (%)	Recall (%)	F1-score (%)	Nombre d'images
Grape_Black rot	96.64	97.46	97.05	118
Grape_Esca (Black Measles)	99.28	98.56	98.92	139
Grape_Leaf blight (Isariopsis Leaf Spot)	97.27	98.17	97.72	109
Grape_Healthy	99.06	99.06	99.06	106
Pepper bell_Bacterial spot	97.96	97.30	97.63	148
Pepper bell_Healthy	98.03	100.00	99.00	149
Potato_Early blight	97.03	98.00	97.51	100
Potato_Late blight	85.59	95.00	90.05	100
Potato_Healthy	97.03	97.03	97.03	101
Tomato_Bacterial spot	97.71	99.53	98.61	214
Tomato_Early blight	96.62	95.82	96.22	239
Tomato_Late blight	96.48	92.02	94.19	238
Tomato_Leaf Mold	100.00	98.31	99.15	236
Tomato_Septoria leaf spot	96.26	93.64	94.93	220
Tomato_Spider mites Two-spotted spider mite	92.67	98.62	95.56	218
Tomato_Target Spot	98.64	93.56	96.04	233
Tomato_Yellow Leaf Curl Virus	99.59	98.78	99.18	245
Tomato_Mosaic virus	99.57	100.00	99.78	231
Tomato_Healthy	98.78	100.00	99.38	242
accuracy			97.31	3386
macro avg	97.06	97.41	97.31	3386
weighted avg	97.37	97.31	97.32	3386

Table 5.6: Précision, recall et F1-score par classe du modèle Sugeno avec PSO appliqué à CNN, CNN+KNN et CNN+Random Forest

5.2.4 CNN+SVM et CNN +KNN et CNN +Random Forest

Dans cette partie, nous allons décrire les performances obtenues en appliquant l'intégrale de Sugeno avec PSO sur trois modèles, à savoir les modèles CNN+SVM, le modèle CNN+KNN et le modèle CNN+Random Forest, comme montré dans la figure 5.7. Le tableau 5.7 résume les paramètres de l'optimisation par essaim de particules utilisés pour cette combinaison, à savoir les valeurs des positions initiales, la taille de l'essaim, la dimension de l'espace de recherche, l'intervalle de recherche, les valeurs des coefficients a_1 et a_2, l'inertie w, le nombre d'itérations, et le temps de recherche. Nous incluons également l'exactitude de chaque modèle avant l'application de l'intégrale ainsi que l'exactitude obtenue après avoir appliqué l'intégrale de Sugeno avec PSO, et le temps d'inférence d'une image.

Paramètres de PSO pour (CNN + SVM, CNN + KNN et CNN + Random Forest)						
Valeur des positions initiales	Taille de l'essaim	Dimension de l'espace de recherche	L'intervalle de recherche	Valeurs des coefficients a_1, a_2 et inertie w	Nombre d'itérations	Temps de recherche
[0.45,0.5,0.6]	5	3	[0,1]	a_1=1, a_2=2 , w=0.5	10	1h 23 min
Sugeno intégrale avec PSO pour (CNN + SVM, CNN + KNN et CNN + Random Forest)						
Descripteur	Classifieur	Exactitude des modèles (%)	Mesures floues optimales obtenues avec PSO	Exactitude obtenue avec Choquet et PSO (%)	Temps d'inférence d'une seule image	
CNN	SVM	95.75	1	97.78	0.5 s	
CNN	KNN	95.30	0.94			
CNN	Random Forest	95.42	0.56			

Table 5.7: Paramètres de modèle 4 de l'intégrale de sugeno avec PSO adoptée

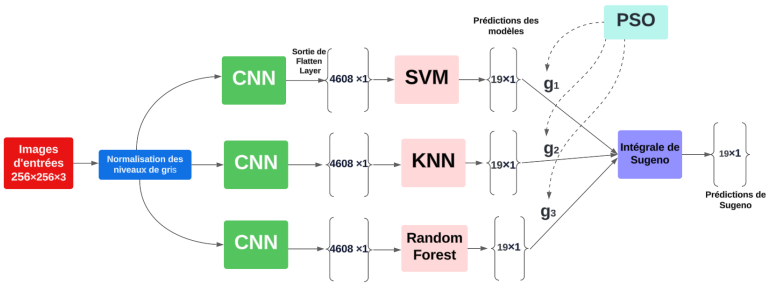

Figure 5.7: Sugeno avec PSO appliqué à CNN+SVM, CNN+KNN et CNN+Random Forest.

D'apès le tableau5.7 On constate que cette intégrale a permis une amélioration de 2.03% par rapport au meilleur des modèles disponibles, ce qui est très satisfaisant.
En analysant la matrice de confusion 5.8, on observe que parmi les 3386 images testées, 75 sont mal classées. Par exemple :

- 6 images de la classe *Tomato-target-spot* sont incorrectement classées comme *Tomato-spider-mites-two-spotted-spider-mite.*
- 6 images de la classe *Tomato-Late-blight* sont incorrectement classées comme *Potato-late-blight.*

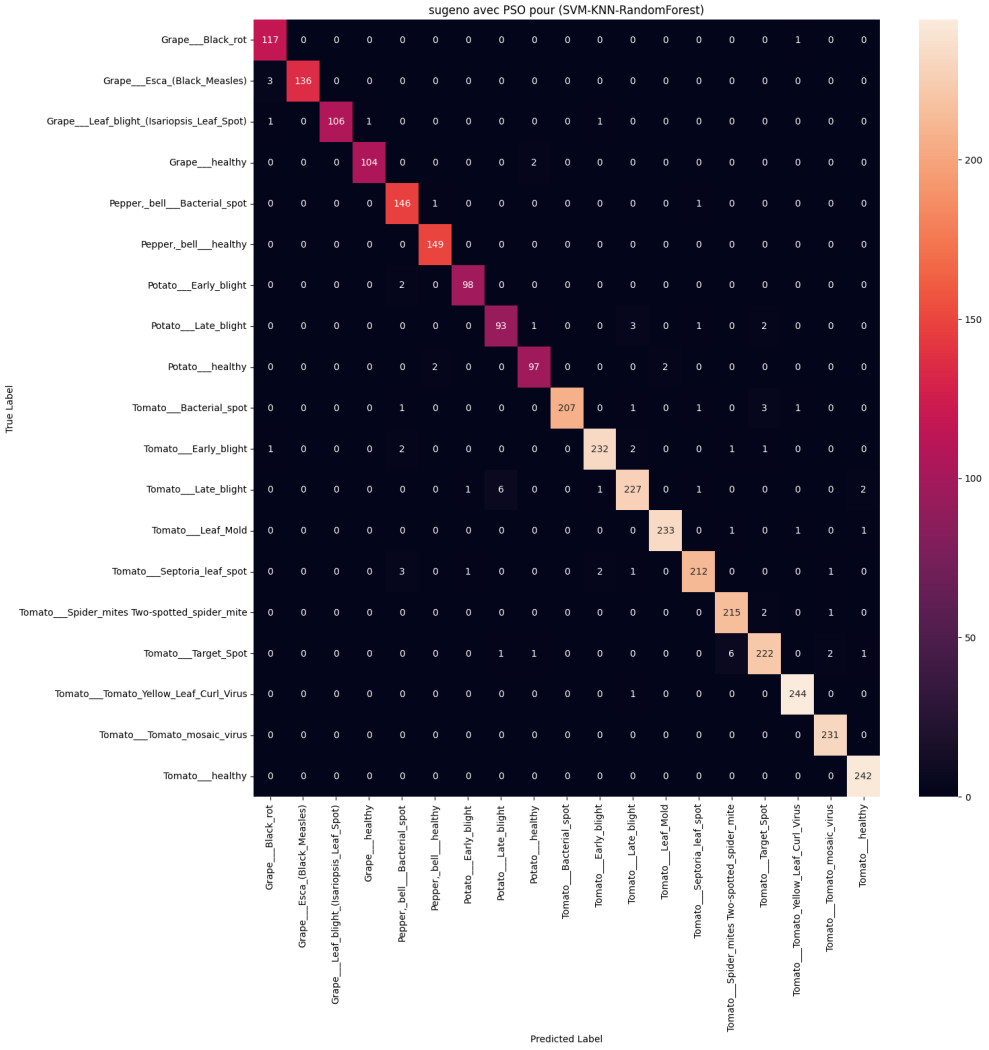

Figure 5.8: Matrice de confusion du modèle Sugeno avec PSO appliqué à CNN+SVM, CNN+KNN et CNN+Random Forest

Cette observation est soutenue par les valeurs de précision et de recall fournies dans le tableau 5.8. Une précision élevée indique que le modèle réduit au minimum le nombre de faux positifs, tandis qu'un recall élevé signifie que le modèle maximise le nombre de vrais positifs. Pour une évaluation plus complète de la performance du modèle, nous examinons également le score F1, plus il est élevé, plus le modèle est considéré comme performant.

	Précision (%)	Recall (%)	F1-score (%)	Nombre d'images
Grape_Black rot	95.90	99.15	97.50	118
Grape_Esca (Black Measles)	100.00	97.84	98.91	139
Grape_Leaf blight (Isariopsis Leaf Spot)	100.00	97.25	98.60	109
Grape_Healthy	99.05	98.11	98.58	106
Pepper bell_Bacterial spot	94.81	98.65	96.69	148
Pepper bell_Healthy	98.03	100.00	99.00	149
Potato_Early blight	98.00	98.00	98.00	100
Potato_Late blight	93.00	93.00	93.00	100
Potato_Healthy	96.04	96.04	96.04	101
Tomato_Bacterial spot	100.00	96.73	98.34	214
Tomato_Early blight	98.31	97.07	97.68	239
Tomato_Late blight	96.60	95.38	95.98	238
Tomato_Leaf Mold	99.15	98.73	98.94	236
Tomato_Septoria leaf spot	98.15	96.36	97.25	220
Tomato_Spider mites Two-spotted spider mite	96.41	98.62	97.51	218
Tomato_Target Spot	96.52	95.28	95.90	233
Tomato_Yellow Leaf Curl Virus	96.52	99.59	99.19	245
Tomato_Mosaic virus	98.30	100.00	99.14	231
Tomato_Healthy	98.37	100.00	99.18	242
accuracy			97.78	3386
macro avg	97.65	97.67	97.65	3386
weighted avg	97.80	97.78	97.78	3386

Table 5.8: Précision, recall et F1-score par classe du modèle Sugeno avec PSO appliqué à CNN+SVM, CNN+KNN et CNN+Random Forest

5.2.5 CNN, CNN +SVM, CNN +KNN,CNN +Random Forest

Dans cette partie, nous allons décrire les performances obtenues en appliquant l'intégrale de Sugeno avec PSO sur quatre modèles, à savoir les modèles CNN avec comme classifieur les couches entièrement connectées, le modèle CNN+SVM et le modèle CNN+KNN et CNN + Random Forest, comme montré dans la figure 5.9. Le tableau 5.9 résume les paramètres de l'optimisation par essaim de particules utilisés pour cette combinaison, à savoir les valeurs des positions initiales, la taille de l'essaim, la dimension de l'espace de recherche, l'intervalle de recherche, les valeurs des coefficients a_1 et a_2, l'inertie w, le nombre d'itérations, et le temps de recherche. Nous incluons également l'exactitude de chaque modèle avant l'application de l'intégrale ainsi que l'exactitude obtenue après avoir appliqué l'intégrale de Sugeno avec PSO, et le temps d'inférence d'une image.

Paramètres de PSO pour (CNN,CNN + SVM, CNN + KNN et CNN + Random Forest)						
Valeur des positions initiales	Taille de l'essaim	Dimension de l'espace de recherche	L'intervalle de recherche	Valeurs des coefficients a_1, a_2 et inertie w	Nombre d'itérations	Temps de recherche
[0.65,0.5,0.4,0.7]	5	4	[0,1]	a_1=1, a_2=2 , w=0.5	10	1h 47 min
Sugeno intégrale avec PSO pour (CNN ,CNN + SVM, CNN + KNN et CNN + Random Forest)						
Descripteur	**Classifieur**	**Exactitude des modèles (%)**	**Mesures floues optimales obtenues avec PSO**	**Exactitude obtenue avec Choquet et PSO (%)**	**Temps d'inférence d'une seule image**	
CNN	Couche entièrement connectée	94.42	0.30	97.64	0.7 s	
CNN	SVM	95.75	0.88			
CNN	KNN	95.30	0.77			
CNN	Random Forest	95.42	0.79			

Table 5.9: Paramètres de modèle 5 de l'intégrale de sugeno avec PSO adoptée

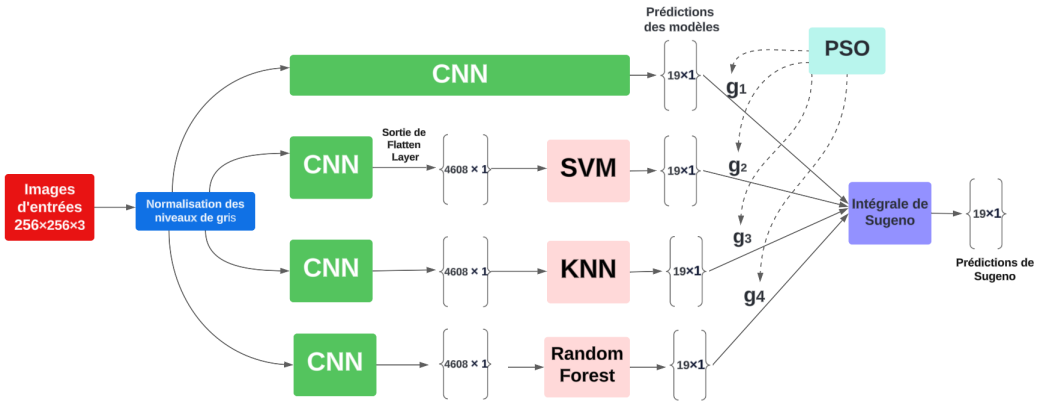

Figure 5.9: Sugeno avec PSO appliqué à CNN, CNN+SVM, CNN+KNN et CNN+Random Forest

D'après le tableau 5.9 On remarque que la dimension de l'espace de recherche égale à 4 car pour cette combinaison, nous disposons de quatre modèles. Il est donc nécessaire de déterminer la mesure floue à attribuer à chaque modèle. On constate aussi que cette intégrale a permis une amélioration de 1,9% par rapport au meilleur des modèles disponibles, ce qui est bon.
En analysant la matrice de confusion 5.10, on observe que parmi les 3386 images testées, 80 sont mal classées. Par exemple :

- 9 images de la classe *Tomato-target-spot* sont incorrectement classées comme *Tomato-spider-mites-two-spotted-spider-mite*.
- 11 images de la classe *Tomato-Late-blight* sont incorrectement classées comme *Potato-late-blight*.

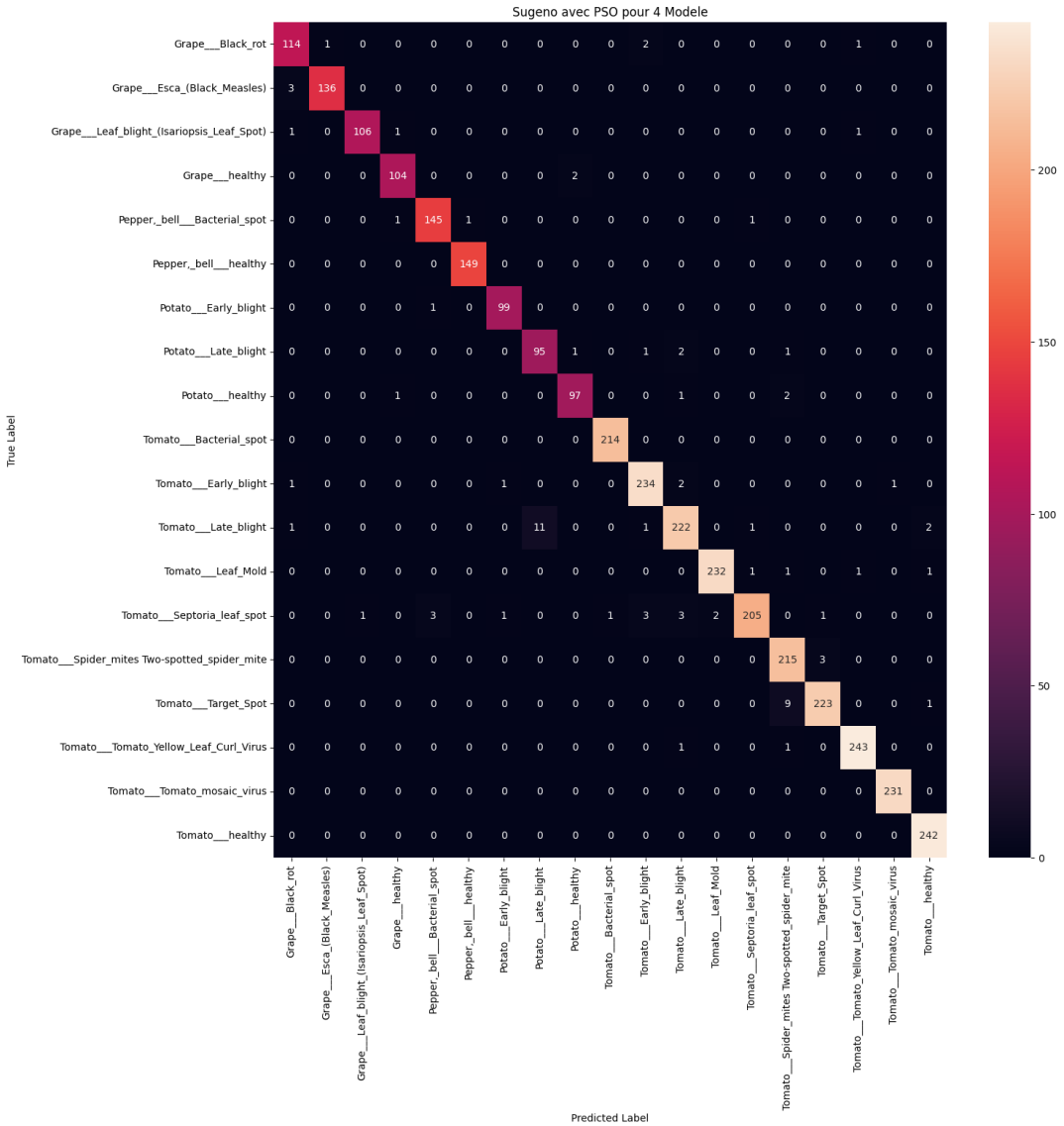

Figure 5.10: Matrice de confusion du modèle Sugeno avec PSO appliqué à CNN, CNN+SVM, CNN+KNN et CNN+Random Forest

Cette observation est soutenue par les valeurs de précision et de recall fournies dans le tableau 5.10. Une précision élevée indique que le modèle réduit au minimum le nombre de faux positifs, tandis qu'un recall élevé signifie que le modèle maximise le nombre de vrais positifs. Pour une évaluation plus complète de la performance du modèle, nous examinons également le score F1, plus il est élevé, plus le modèle est considéré comme performant.

	Précision (%)	Recall (%)	F1-score (%)	Nombre d'images
Grape_Black rot	95.00	96.61	95.80	118
Grape_Esca (Black Measles)	99.27	97.84	98.55	139
Grape_Leaf blight (Isariopsis Leaf Spot)	99.07	97.25	98.15	109
Grape_Healthy	97.20	98.11	97.65	106
Pepper bell_Bacterial spot	97.32	97.97	97.64	148
Pepper bell_Healthy	99.33	100.00	99.67	149
Potato_Early blight	98.02	99.00	98.51	100
Potato_Late blight	89.62	95.00	92.23	100
Potato_Healthy	97.00	96.04	96.52	101
Tomato_Bacterial spot	99.53	100.00	99.77	214
Tomato_Early blight	97.10	97.91	97.50	239
Tomato_Late blight	96.10	93.28	94.67	238
Tomato_Leaf Mold	99.15	98.31	98.72	236
Tomato_Septoria leaf spot	98.56	93.18	95.79	220
Tomato_Spider mites Two-spotted spider mite	93.89	98.62	96.20	218
Tomato_Target Spot	98.24	95.71	96.96	233
Tomato_Yellow Leaf Curl Virus	98.78	99.18	98.98	245
Tomato_Mosaic virus	99.57	100.00	99.78	231
Tomato_Healthy	98.37	100.00	99.18	242
accuracy			**97.64**	3386
macro avg	97.43	97.58	97.49	3386
weighted avg	97.67	97.64	97.64	3386

Table 5.10: Précision, recall et F1-score par classe du modèle Sugeno avec PSO appliqué à CNN,CNN+SVM, CNN+KNN et CNN+Random Forest

5.3 Les combinaisons de l'intégrale de Choquet avec PSO

Dans cette partie on va décrire les résultat obtenu on applicant l'intégrale de Choquet avec l'optimisation par essaim de particule pour les différents combianison des modèles qu'on dispose

5.3.1 CNN et CNN + SVM et CNN +KNN

Dans cette partie, nous allons décrire les performances obtenues en appliquant l'intégrale de Choquet avec PSO sur trois modèles, à savoir les modèles CNN avec comme classifieur les couches entièrement connectées, le modèle CNN+SVM et le modèle CNN+KNN, comme montré dans la figure 5.11. Le tableau 5.11 résume les paramètres de l'optimisation par essaim de particules utilisés pour cette combinaison, à savoir les valeurs des positions initiales, la taille de l'essaim, la dimension de l'espace de recherche, l'intervalle de recherche, les valeurs des coefficients a_1 et a_2, l'inertie w, le nombre d'itérations, et le temps de recherche. Nous incluons également l'exactitude de chaque modèle avant l'application de l'intégrale ainsi que l'exactitude obtenue après l'application de l'intégrale de Choquet avec PSO, et le temps d'inférence.

Paramètres de PSO pour (CNN , CNN + SVM et CNN + KNN)						
Valeur des positions initiales	Taille de l'essaim	Dimension de l'espace de recherche	L'intervalle de recherche	Valeurs des coefficients a_1, a_2 et inertie w	Nombre d'itérations	Temps de recherche
[0.65,0.5,0.4]	5	3	[0,1]	a_1=1, a_2=2 , w =0.5	10	2h 14 min
Choquet intégrale avec PSO pour (CNN , CNN + SVM et CNN + KNN)						
Descripteur	Classifieur	Exactitude des modèles (%)	Mesures floues optimales obtenues avec PSO	Exactitude obtenue avec Choquet et PSO (%)	Temps d'inférence d'une seule image	
CNN	Couche entièrement connectée	94.42	0.67	**98.05**	0.6 s	
CNN	SVM	95.75	0.73			
CNN	KNN	95.30	0.30			

Table 5.11: Paramètres de modèle 1 de l'intégrale de Choquet avec PSO adoptée

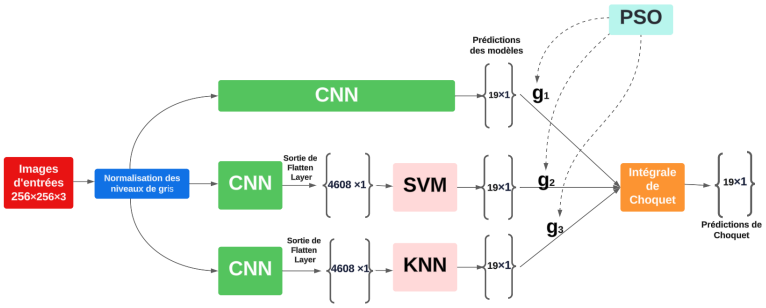

Figure 5.11: Choquet avec PSO appliqué à CNN, CNN+SVM, CNN+KNN

D'après le tableau 5.11 On remarque que le temps de recherche pour cette optimisation est très élevé (2h 14 min), due au fait que l'entraînement de cette optimisation a été effectué en utilisant le CPU de Google Colaboratory.
La dimension de l'espace de recherche égale à 3 car dans cette combinaison, nous disposons de trois modèles. Il est donc nécessaire de déterminer la mesure floue à attribuer à chaque modèle.
On constate que cette intégrale a permis une amélioration de 2.3% par rapport au meilleur des modèles disponibles, ce qui est très satisfaisant.
En analysant la matrice de confusion 5.12, on observe que parmi les 3386 images testées, 66 sont mal classées. Par exemple :

- 7 images de la classe *Tomato-target-spot* sont incorrectement classées comme *Tomato-spider-mites-two-spotted-spider-mite*.
- 5 images de la classe *Tomato-Late-blight* sont incorrectement classées comme *Potato-late-blight*.

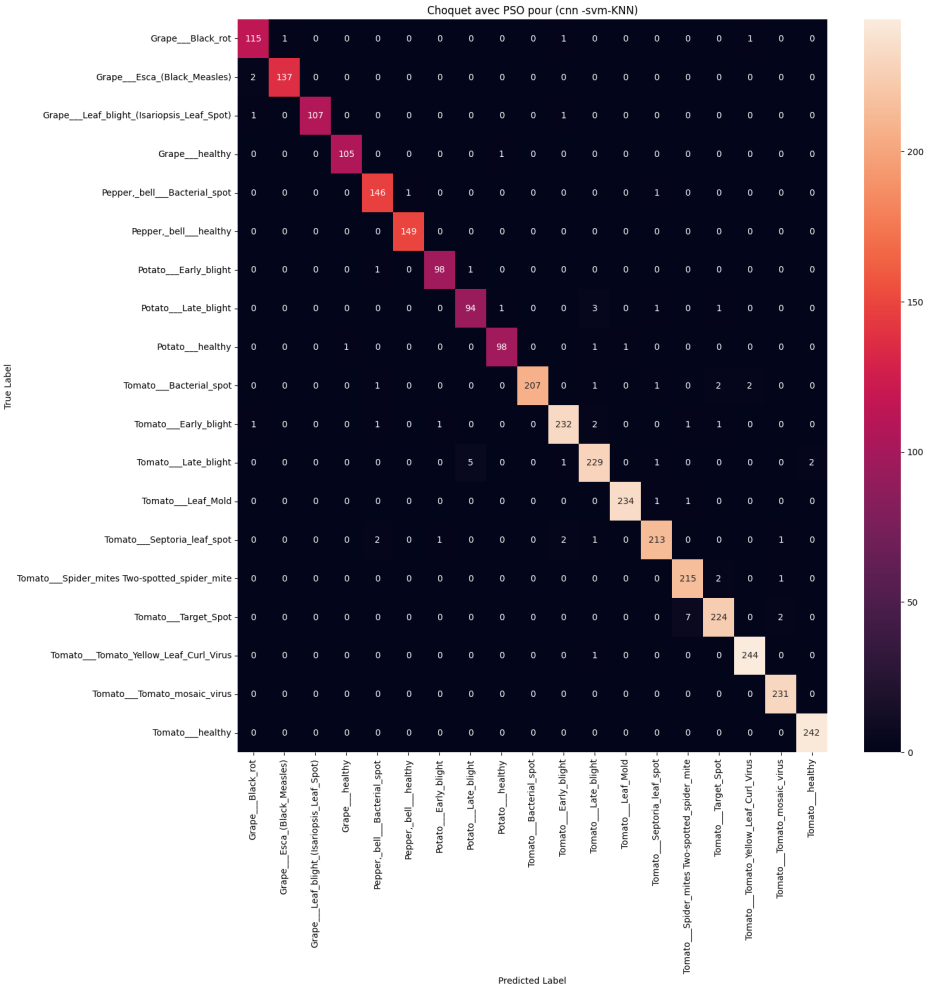

Figure 5.12: Matrice de confusion du modèle Choquet avec PSO appliqué à CNN, CNN+SVM, CNN+KNN

Cette observation est soutenue par les valeurs de précision et de recall fournies dans le tableau 5.12. Une précision élevée indique que le modèle réduit au minimum le nombre de faux positifs, tandis qu'un recall élevé signifie que le modèle maximise le nombre de vrais positifs. Pour une évaluation plus complète de la performance du modèle, nous examinons également le score F1, plus il est élevé, plus le modèle est considéré comme performant.

	Précision (%)	Recall (%)	F1-score (%)	Nombre d'images
Grape_Black rot	96.64	97.46	97.05	118
Grape_Esca (Black Measles)	99.28	98.56	98.92	139
Grape_Leaf blight (Isariopsis Leaf Spot)	100.00	98.17	99.07	109
Grape_Healthy	99.06	99.06	99.06	106
Pepper bell_Bacterial spot	96.69	98.65	97.66	148
Pepper bell_Healthy	99.33	100.00	99.67	149
Potato_Early blight	98.00	98.00	98.00	100
Potato_Late blight	94.00	94.00	94.00	100
Potato_Healthy	98.00	97.03	97.51	101
Tomato_Bacterial spot	100.00	96.73	98.34	214
Tomato_Early blight	97.89	97.07	97.48	239
Tomato_Late blight	96.22	96.22	96.22	238
Tomato_Leaf Mold	99.57	99.15	99.36	236
Tomato_Septoria leaf spot	97.71	96.82	97.26	220
Tomato_Spider mites Two-spotted spider mite	95.98	98.62	97.29	218
Tomato_Target Spot	97.39	96.14	96.76	233
Tomato_Yellow Leaf Curl Virus	98.79	99.59	99.19	245
Tomato_Mosaic virus	98.30	100.00	99.14	231
Tomato_Healthy	99.18	100.00	99.59	242
accuracy			98.05	3386
macro avg	98.00	97.96	97.98	3386
weighted avg	98.06	98.05	98.05	3386

Table 5.12: Précision, recall et F1-score par classe du modèle Choquet avec PSO appliqué à CNN, CNN+SVM,CNN+KNN

5.3.2 CNN et CNN + SVM et CNN +Random Forest

Dans cette partie, nous allons décrire les performances obtenues en appliquant l'intégrale de Choquet avec PSO sur trois modèles, à savoir les modèles CNN avec comme classifieur les couches entièrement connectées, le modèle CNN+SVM et le modèle CNN+Random Forest, comme montré dans la figure 5.13. Le tableau 5.13 résume les paramètres de l'optimisation par essaim de particules utilisés pour cette combinaison, à savoir les valeurs des positions initiales, la taille de l'essaim, la dimension de l'espace de recherche, l'intervalle de recherche, les valeurs des coefficients a_1 et a_2, l'inertie w, le nombre d'itérations, et le temps de recherche. Nous incluons également l'exactitude de chaque modèle avant l'application de l'intégrale ainsi que l'exactitude obtenue après avoir appliqué l'intégrale de Choquet avec PSO, et le temps d'inférence d'une image.

Paramètres de PSO pour (CNN , CNN + svm et CNN + Random Forest)						
Valeur des positions initiales	Taille de l'essaim	Dimension de l'espace de recherche	L'intervalle de recherche	Valeurs des coefficients a_1, a_2 et inertie w	Nombre d'itérations	Temps de recherche
[0.65,0.5,0.4]	5	3	[0,1]	a_1=1, a_2=2 , w=0.5	10	2h 18 min
Choquet intégrale avec PSO pour (CNN , CNN + SVM et CNN + Random Forest)						
Descripteur	Classifieur	Exactitude des modèles (%)	Mesures floues optimales obtenues avec PSO	Exactitude obtenue avec Choquet et PSO (%)	Temps d'inférence d'une seule image	
CNN	Couche entièrement connectée	94.42	0.14	98.08	0.6 s	
CNN	SVM	95.75	0.25			
CNN	Random Forest	95.42	0.25			

Table 5.13: Paramètres de modèle 2 de l'intégrale de Choquet avec PSO adoptée

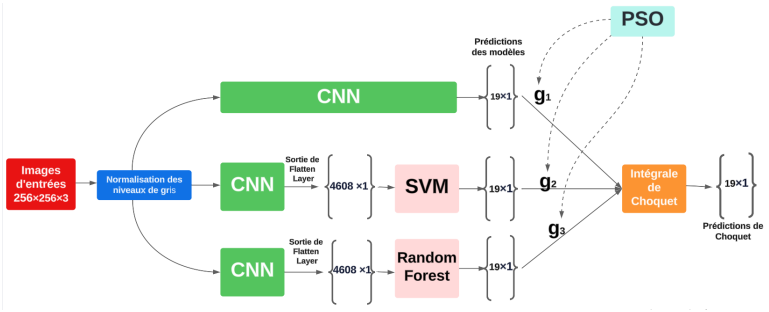

Figure 5.13: Choquet avec PSO appliqué à CNN, CNN+SVM,CNN+Random Forest.

D'après le tableau 5.13 On constate que cette intégrale a permis une amélioration de 2.33% par rapport au meilleur des modèles disponibles, ce qui est très satisfaisant.

En analysant la matrice de confusion 5.14, on observe que parmi les 3386 images testées, 65 sont mal classées. Par exemple :

- 11 images de la classe *Tomato-target-spot* sont incorrectement classées comme *Tomato-spider-mites-two-spotted-spider-mite*.
- 6 images de la classe *Tomato-Late-blight* sont incorrectement classées comme *Potato-late-blight*.

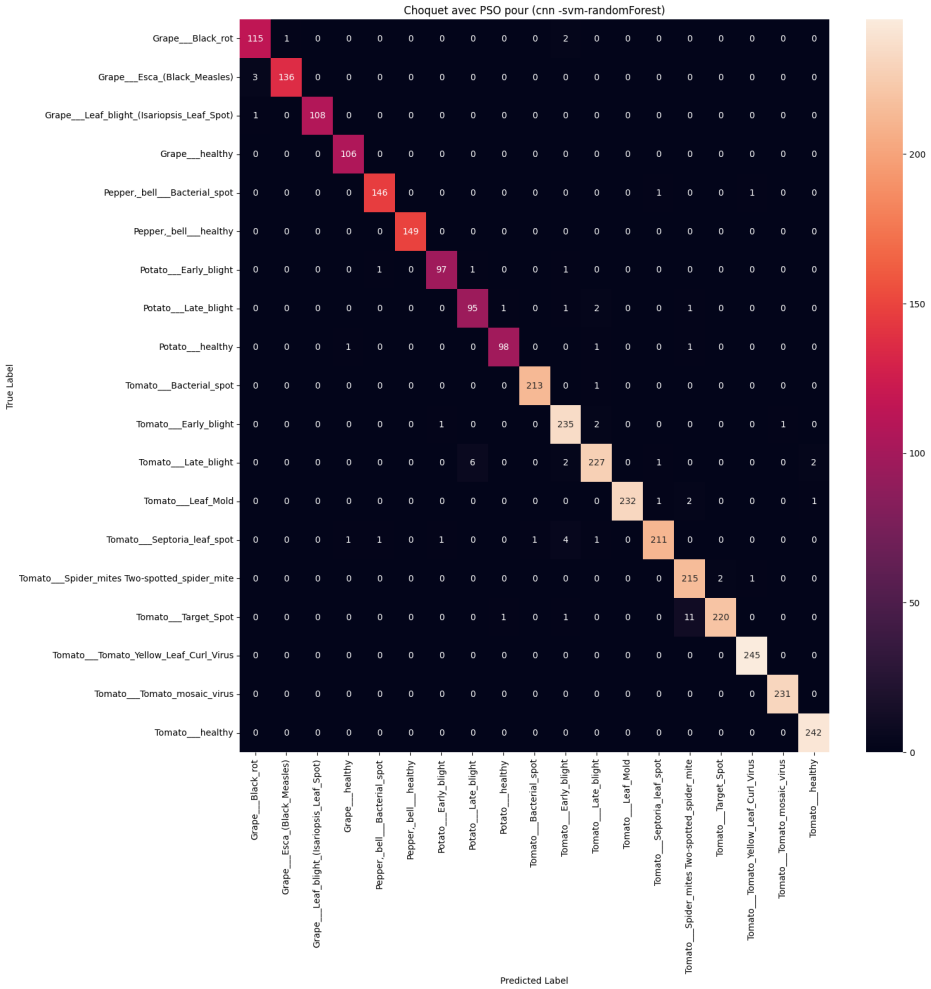

Figure 5.14: Matrice de confusion du modèle Choquet avec PSO appliqué à CNN, CNN+SVM, CNN+Random Forest

Cette observation est soutenue par les valeurs de précision et de recall fournies dans le tableau 5.14. Une précision élevée indique que le modèle réduit au minimum le nombre de faux positifs, tandis qu'un recall élevé signifie que le modèle maximise le nombre de vrais positifs. Pour une évaluation plus complète de la performance du modèle, nous examinons également le score F1, plus il est élevé, plus le modèle est considéré comme performant.

	Précision (%)	Recall (%)	F1-score (%)	Nombre d'images
Grape_Black rot	96.64	97.46	97.05	118
Grape_Esca (Black Measles)	99.27	97.84	98.55	139
Grape_Leaf blight (Isariopsis Leaf Spot)	100.00	99.08	99.54	109
Grape_Healthy	98.15	100.00	99.07	106
Pepper bell_Bacterial spot	98.65	98.65	98.65	148
Pepper bell_Healthy	100.00	100.00	100.00	149
Potato_Early blight	97.98	97.00	97.49	100
Potato_Late blight	93.14	95.00	94.06	100
Potato_Healthy	98.00	97.03	97.51	101
Tomato_Bacterial spot	99.53	99.53	99.53	214
Tomato_Early blight	95.53	98.33	96.91	239
Tomato_Late blight	97.01	95.38	96.19	238
Tomato_Leaf Mold	100.00	98.31	99.15	236
Tomato_Septoria leaf spot	98.60	95.91	97.24	220
Tomato_Spider mites Two-spotted spider mite	93.48	98.62	95.98	218
Tomato_Target Spot	99.10	94.42	96.70	233
Tomato_Yellow Leaf Curl Virus	99.19	100.00	99.59	245
Tomato_Mosaic virus	99.57	100.00	99.78	231
Tomato_Healthy	98.78	100.00	99.38	242
accuracy			98.08	3386
macro avg	98.03	98.03	98.02	3386
weighted avg	98.10	98.08	98.08	3386

Table 5.14: Précision, recall et F1-score par classe du modèle Choquet avec PSO appliqué à CNN, CNN+SVM,CNN+Random Forest

5.3.3 CNN et CNN +KNN et CNN +Random Forest

Dans cette partie, nous allons décrire les performances obtenues en appliquant l'intégrale de Choquet avec PSO sur trois modèles, à savoir les modèles CNN avec comme classifieur les couches entièrement connectées, le modèle CNN+KNN et le modèle CNN+Random Forest, comme montré dans la figure 5.15. Le tableau 5.15 résume les paramètres de l'optimisation par essaim de particules utilisés pour cette combinaison, à savoir les valeurs des positions initiales, la taille de l'essaim, la dimension de l'espace de recherche, l'intervalle de recherche, les valeurs des coefficients a_1 et a_2, l'inertie w, le nombre d'itérations, et le temps de recherche. Nous incluons également l'exactitude de chaque modèle avant l'application de l'intégrale ainsi que l'exactitude obtenue après avoir appliqué l'intégrale de Choquet avec PSO, et le temps d'inférence d'une image.

Paramètres de PSO pour (CNN , CNN + KNN et CNN + Random Forest)						
Valeur des positions initiales	Taille de l'essaim	Dimension de l'espace de recherche	L'intervalle de recherche	Valeurs des coefficients a_1, a_2 et inertie w	Nombre d'itérations	Temps de recherche
[0.25,0.3,0.4]	5	3	[0,1]	a_1=1, a_2=2 , w=0.5	10	2h 12 min
Choquet intégrale avec PSO pour (CNN , CNN + KNN et CNN + Random Forest)						
Descripteur	**Classifieur**	**Exactitude des modèles (%)**	**Mesures floues optimales obtenues avec PSO**	**Exactitude obtenue avec Choquet et PSO (%)**	**Temps d'inférence d'une seule image**	
CNN	Couche entièrement connectée	94.42	0.12	97.87	0.6 s	
CNN	KNN	95.30	0.42			
CNN	Random Forest	95.42	0.31			

Table 5.15: Paramètres de modèle 3 de l'intégrale de Choquet avec PSO adoptée

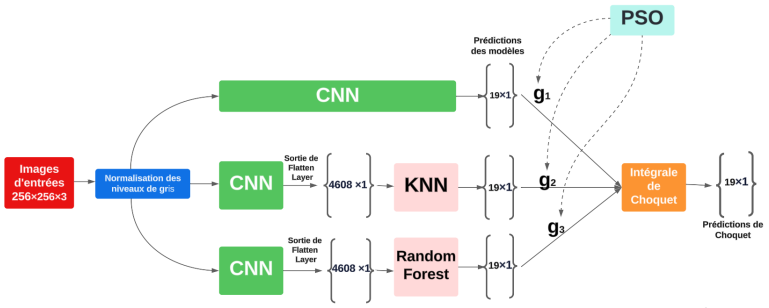

Figure 5.15: Choquet avec PSO appliqué à CNN, CNN+KNN,CNN+Random Forest.

D'après le tableau 5.15 On constate que cette intégrale a permis une amélioration de 2.45% par rapport au meilleur des modèles disponibles, ce qui est très satisfaisant.

En analysant la matrice de confusion 5.16, on observe que parmi les 3386 images testées, 72 sont mal classées. Par exemple :

- 11 images de la classe *Tomato-target-spot* sont incorrectement classées comme *Tomato-spider-mites-two-spotted-spider-mite*.
- 9 images de la classe *Tomato-Late-blight* sont incorrectement classées comme *Potato-late-blight*.

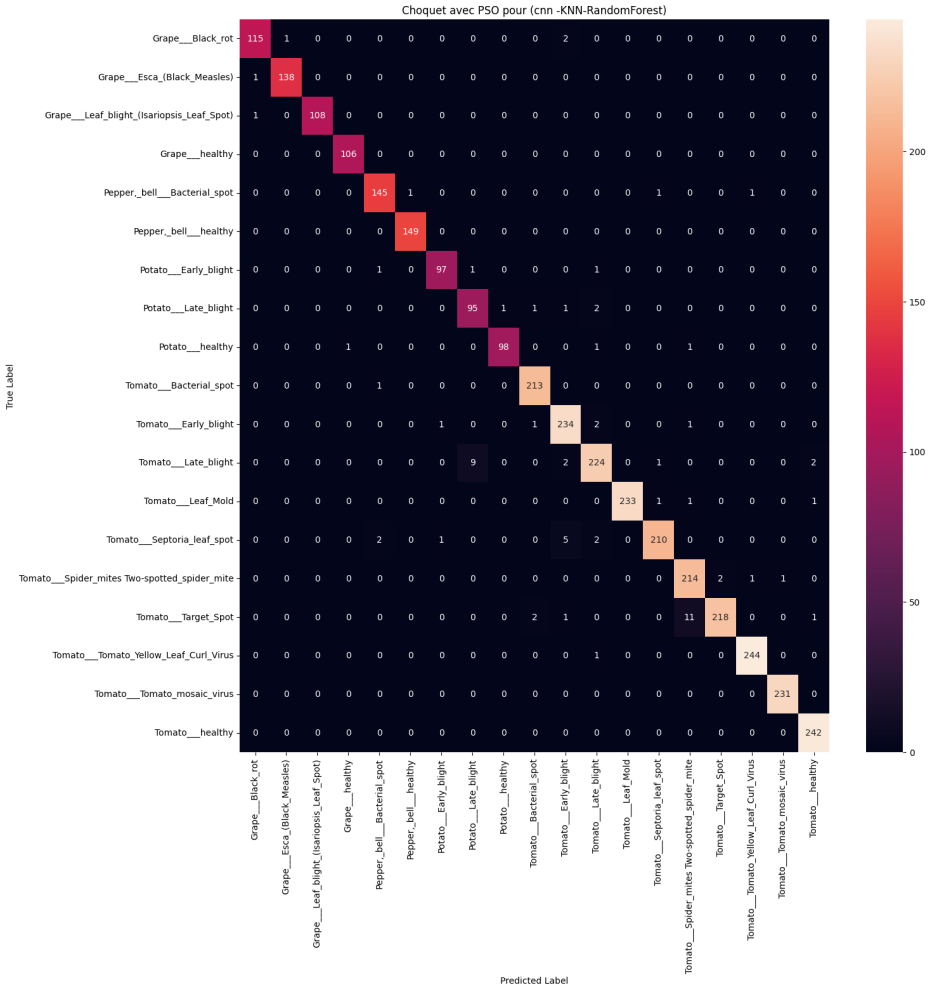

Figure 5.16: Matrice de confusion du modèle Choquet avec PSO appliqué à CNN, CNN+KNN, CNN+Random Forest

Cette observation est soutenue par les valeurs de précision et de recall fournies dans le tableau 5.16 Une précision élevée indique que le modèle réduit au minimum le nombre de faux positifs, tandis qu'un recall élevé signifie que le modèle maximise le nombre de vrais positifs. Pour une évaluation plus complète de la performance du modèle, nous examinons également le score F1 , plus il est élevé, plus le modèle est considéré comme performant.

	Précision (%)	Recall (%)	F1-score (%)	Nombre d'images
Grape_Black rot	98.28	97.46	97.87	118
Grape_Esca (Black Measles)	99.28	99.28	99.28	139
Grape_Leaf blight (Isariopsis Leaf Spot)	100.00	99.08	99.54	109
Grape_Healthy	99.07	100.00	99.53	106
Pepper bell_Bacterial spot	97.32	97.97	97.64	148
Pepper bell_Healthy	99.33	100.00	99.67	149
Potato_Early blight	97.98	97.00	97.49	100
Potato_Late blight	90.48	95.00	92.68	100
Potato_Healthy	98.99	97.03	98.00	101
Tomato_Bacterial spot	98.56	99.53	98.84	214
Tomato_Early blight	95.51	97.91	96.69	239
Tomato_Late blight	96.14	94.12	95.12	238
Tomato_Leaf Mold	100.00	98.73	99.36	236
Tomato_Septoria leaf spot	98.59	95.45	97.00	220
Tomato_Spider mites Two-spotted spider mite	93.86	98.17	95.96	218
Tomato_Target Spot	99.09	93.56	96.25	233
Tomato_Yellow Leaf Curl Virus	99.19	99.59	99.39	245
Tomato_Mosaic virus	99.57	100.00	99.78	231
Tomato_Healthy	98.37	100.00	99.18	242
accuracy			**97.87**	3386
macro avg	97.85	97.89	97.86	3386
weighted avg	97.91	97.87	97.87	3386

Table 5.16: Précision, recall et F1-score par classe du modèle Choquet avec PSO appliqué à CNN, CNN+KNN,CNN+Random Forest

5.3.4 CNN + SVM et CNN +KNN et CNN +Random Forest

Dans cette partie, nous allons décrire les performances obtenues en appliquant l'intégrale de Choquet avec PSO sur trois modèles, à savoir les modèles CNN+SVM, le modèle CNN+KNN et le modèle CNN+Random Forest, comme montré dans la figure 5.17. Le tableau 5.17 résume les paramètres de l'optimisation par essaim de particules utilisés pour cette combinaison, à savoir les valeurs des positions initiales, la taille de l'essaim, la dimension de l'espace de recherche, l'intervalle de recherche, les valeurs des coefficients a_1 et a_2, l'inertie w, le nombre d'itérations, et le temps de recherche. Nous incluons également l'exactitude de chaque modèle avant l'application de l'intégrale ainsi que l'exactitude obtenue après avoir appliqué l'intégrale de Choquet avec PSO, et le temps d'inférence d'une image.

Paramètres de PSO pour (CNN + SVM, CNN + KNN et CNN + Random Forest)						
Valeur des positions initiales	Taille de l'essaim	Dimension de l'espace de recherche	L'intervalle de recherche	Valeurs des coefficients a_1, a_2 et inertie w	Nombre d'itérations	Temps de recherche
[0.65,0.5,0.4]	5	3	[0,1]	a_1=1, a_2 =2, w =0.5	10	2h 6 min
Choquet intégrale avec PSO pour (CNN + SVM, CNN + KNN et CNN + Random Forest)						
Descripteur	**Classifieur**	**Exactitude des modèles (%)**	**Mesures floues optimales obtenues avec PSO**	**Exactitude obtenue avec Choquet et PSO (%)**	**Temps d'inférence d'une seule image**	
CNN	SVM	95.75	0.71	**98.02**	0.6 s	
CNN	KNN	95.30	0.44			
CNN	Random Forest	95.42	0.65			

Table 5.17: Paramètres de modèle 4 de l'intégrale de Choquet avec PSO adoptée

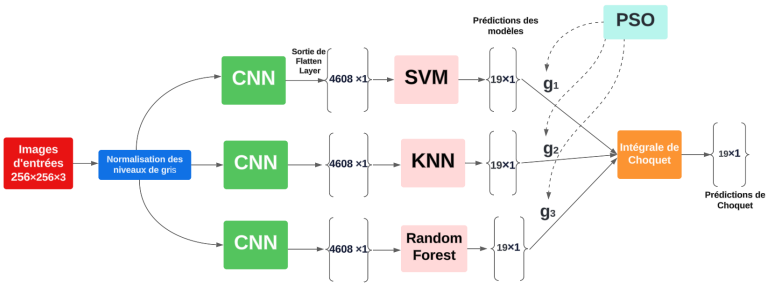

Figure 5.17: Choquet avec PSO appliqué à CNN+SVM, CNN+KNN,CNN+Random Forest

D'après le tableau 5.17 On constate que cette intégrale a permis une amélioration de 2.3% par rapport au meilleur des modèles disponibles, ce qui est très satisfaisant.
En analysant la matrice de confusion 5.18, on observe que parmi les 3386 images testées, 67 sont mal classées. Par exemple :

- 11 images de la classe *Tomato-target-spot* sont incorrectement classées comme *Tomato-spider-mites-two-spotted-spider-mite*.
- 9 images de la classe *Tomato-Late-blight* sont incorrectement classées comme *Potato-late-blight*.

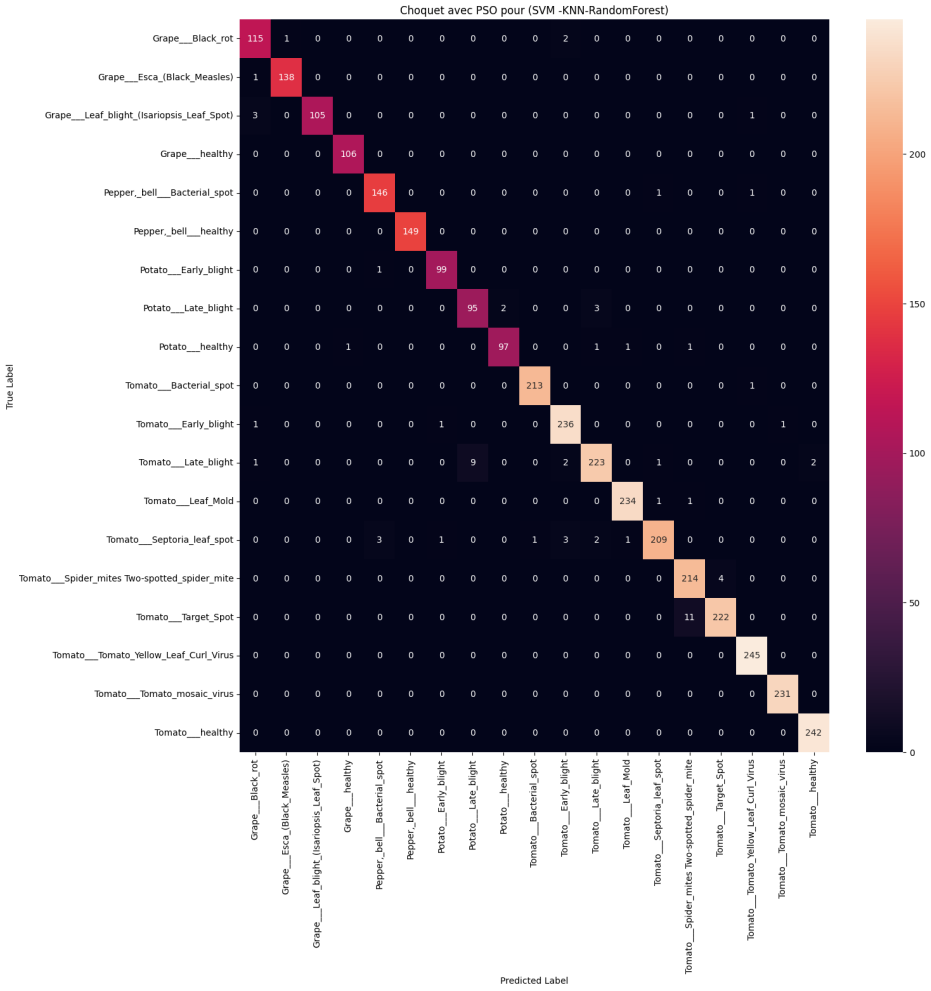

Figure 5.18: Matrice de confusion du modèle Choquet avec PSO appliqué à CNN+SVM, CNN+KNN , CNN+Random Forest

Cette observation est soutenue par les valeurs de précision et de recall fournies dans le tableau 5.18 Une précision élevée indique que le modèle réduit au minimum le nombre de faux positifs, tandis qu'un recall élevé signifie que le modèle maximise le nombre de vrais positifs. Pour une évaluation plus complète de la performance du modèle, nous examinons également le score F1, plus il est élevé, plus le modèle est considéré comme performant.

	Précision (%)	Recall (%)	F1-score (%)	Nombre d'images
Grape_Black rot	95.04	97.46	96.23	118
Grape_Esca (Black Measles)	99.28	99.28	99.28	139
Grape_Leaf blight (Isariopsis Leaf Spot)	100.00	96.33	98.13	109
Grape_Healthy	99.07	100.00	99.53	106
Pepper bell_Bacterial spot	97.33	98.65	97.99	148
Pepper bell_Healthy	100.00	100.00	100.00	149
Potato_Early blight	98.02	99.00	98.51	100
Potato_Late blight	91.35	95.00	93.14	100
Potato_Healthy	97.98	96.04	97.00	101
Tomato_Bacterial spot	99.53	99.53	99.53	214
Tomato_Early blight	97.12	98.74	97.93	239
Tomato_Late blight	97.38	93.70	95.50	238
Tomato_Leaf Mold	99.15	99.15	99.15	236
Tomato_Septoria leaf spot	98.58	95.00	96.76	220
Tomato_Spider mites Two-spotted spider mite	94.27	98.17	96.18	218
Tomato_Target Spot	98.23	95.28	96.73	233
Tomato_Yellow Leaf Curl Virus	98.79	100.00	99.39	245
Tomato_Mosaic virus	99.57	100.00	99.78	231
Tomato_Healthy	99.18	100.00	99.59	242
accuracy			98.02	3386
macro avg	97.89	97.96	97.91	3386
weighted avg	98.04	98.02	98.02	3386

Table 5.18: Précision, recall et F1-score par classe du modèle Choquet avec PSO appliqué à CNN+SVM, CNN+KNN,CNN+Random Forest

5.3.5 CNN et CNN + SVM et CNN +KNN et CNN +Random Forest

Dans cette partie, nous allons décrire les performances obtenues en appliquant l'intégrale de Choquet avec PSO sur quatre modèles, à savoir les modèles CNN avec comme classifieur les couches entièrement connectées, le modèle CNN+SVM et le modèle CNN+KNN et le modèles CNN +Random Forest, comme montré dans la figure 5.19. Le tableau 5.19 résume les paramètres de l'optimisation par essaim de particules utilisés pour cette combinaison, à savoir les valeurs des positions initiales, la taille de l'essaim, la dimension de l'espace de recherche, l'intervalle de recherche, les valeurs des coefficients a_1 et a_2, l'inertie w, le nombre d'itérations, et le temps de recherche. Nous incluons également l'exactitude de chaque modèle avant l'application de l'intégrale ainsi que l'exactitude obtenue après avoir appliqué l'intégrale de Choquet avec PSO, et le temps d'inférence d'une image.

Paramètre de PSO pour (CNN, CNN + SVM, CNN + KNN et CNN + Random Forest)						
Valeur des positions initiales	Taille de l'essaim	Dimension de l'espace de recherche	L'intervalle de recherche	Valeurs des coefficients a_1, a_2 et inertie w	Nombre d'itérations	Temps de recherche
[0.65,0.5,0.4,0.7]	5	4	[0,1]	a_1=1, a_2=2 , w =0.5	10	2h 24 min
Choquet intégrale avec PSO pour (CNN, CNN + SVM, CNN + KNN et CNN + Random Forest)						
Descripteur	Classifieur	Exactitude des modèles (%)	Mesures floues optimales obtenues avec PSO	Exactitude obtenue avec Choquet et PSO (%)	Temps d'inférence d'une seule image	
CNN	Couche entièrement connectée	94.42	0.63	98.38	0.8	
CNN	SVM	95.75	0.72			
CNN	KNN	95.30	0.67			
CNN	Random Forest	95.42	1			

Table 5.19: Paramètres de Modèle 5 de l'intégrale de Choquet avec PSO adoptée

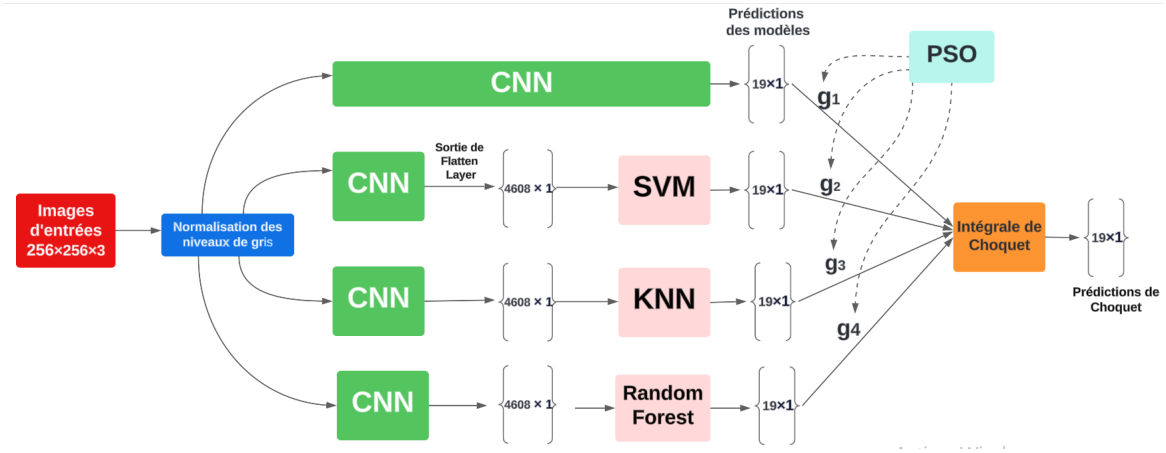

Figure 5.19: Choquet avec PSO appliqué à CNN, CNN+SVM, CNN+KNN,CNN+Random Forest

D'après le tableau 5.19 On remarque que le temps de recherche pour cette optimisation est très élevé(2h 24 min), en raison du fait que l'entraînement de cette optimisation a été effectué en utilisant le CPU de Google Colaboratory.
Nous avons choisi une dimension de l'espace de recherche égale à 4, car pour cette combinaison, nous disposons de quatre modèles. Il est donc nécessaire de déterminer la mesure floue à attribuer à chaque modèle.
On constate que cette intégrale a permis une amélioration de 2.7% par rapport au meilleur des modèles disponibles, ce qui est très satisfaisant.
En analysant la matrice de confusion 5.20, on observe que parmi les 3386 images testées, 55 sont mal classées. Par exemple :

- 10 images de la classe *Tomato-target-spot* sont incorrectement classées comme *Tomato-spider-mites-two-spotted-spider-mite*.
- 7 images de la classe *Tomato-Late-blight* sont incorrectement classées comme *Potato-late-blight*.

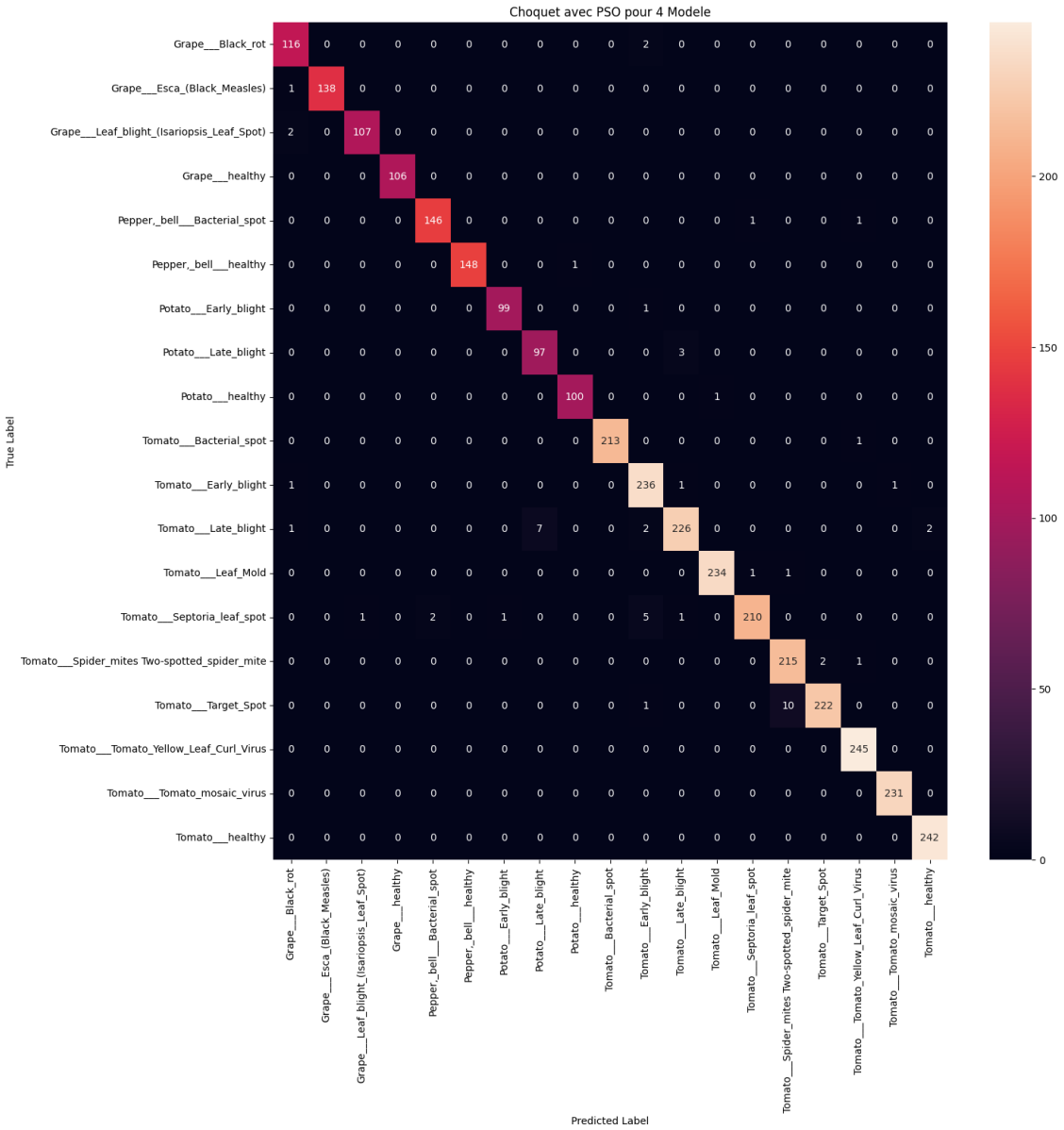

Figure 5.20: Matrice de confusion du modèle Choquet avec PSO appliqué à CNN, CNN+SVM, CNN+KNN , CNN+Random Forest

Cette observation est soutenue par les valeurs de précision et de recall fournies dans le tableau 5.20. Une précision élevée indique que le modèle réduit au minimum le nombre de faux positifs, tandis qu'un recall élevé signifie que le modèle maximise le nombre de vrais positifs. Pour une évaluation plus complète de la performance du modèle, nous examinons également le score F1, plus il est élevé, plus le modèle est considéré comme performant.

	Précision (%)	Recall (%)	F1-score (%)	Nombre d'images
Grape_Black rot	95.87	9831	97.07	118
Grape_Esca (Black Measles)	100.00	99.28	99.64	139
Grape_Leaf blight (Isariopsis Leaf Spot)	99.07	98.17	98.62	109
Grape_Healthy	100.00	100.00	100.00	106
Pepper bell_Bacterial spot	98.65	98.65	98.65	148
Pepper bell_Healthy	100.00	99.33	99.66	149
Potato_Early blight	99.00	99.00	99.00	100
Potato_Late blight	93.27	97.00	95.10	100
Potato_Healthy	99.01	99.01	99.01	101
Tomato_Bacterial spot	100.00	99.53	99.77	214
Tomato_Early blight	95.55	98.74	97.12	239
Tomato_Late blight	97.84	94.96	96.38	238
Tomato_Leaf Mold	99.57	99.15	99.36	236
Tomato_Septoria leaf spot	99.06	95.45	97.22	220
Tomato_Spider mites Two-spotted spider mite	95.13	98.62	96.85	218
Tomato_Target Spot	99.11	95.28	97.16	233
Tomato_Yellow Leaf Curl Virus	98.97	100.00	99.39	245
Tomato_Mosaic virus	99.57	100.00	99.78	231
Tomato_Healthy	99.18	100.00	98.59	242
accuracy			98.38	3386
macro avg	98.35	98.45	98.39	3386
weighted avg	98.40	98.38	98.38	3386

Table 5.20: Précision, recall et F1-score par classe du modèle Choquet avec PSO appliqué à CNN, CNN+SVM,CNN+KNN, CNN+Random Forest

5.4 Récapitulatif et discussion des résultats

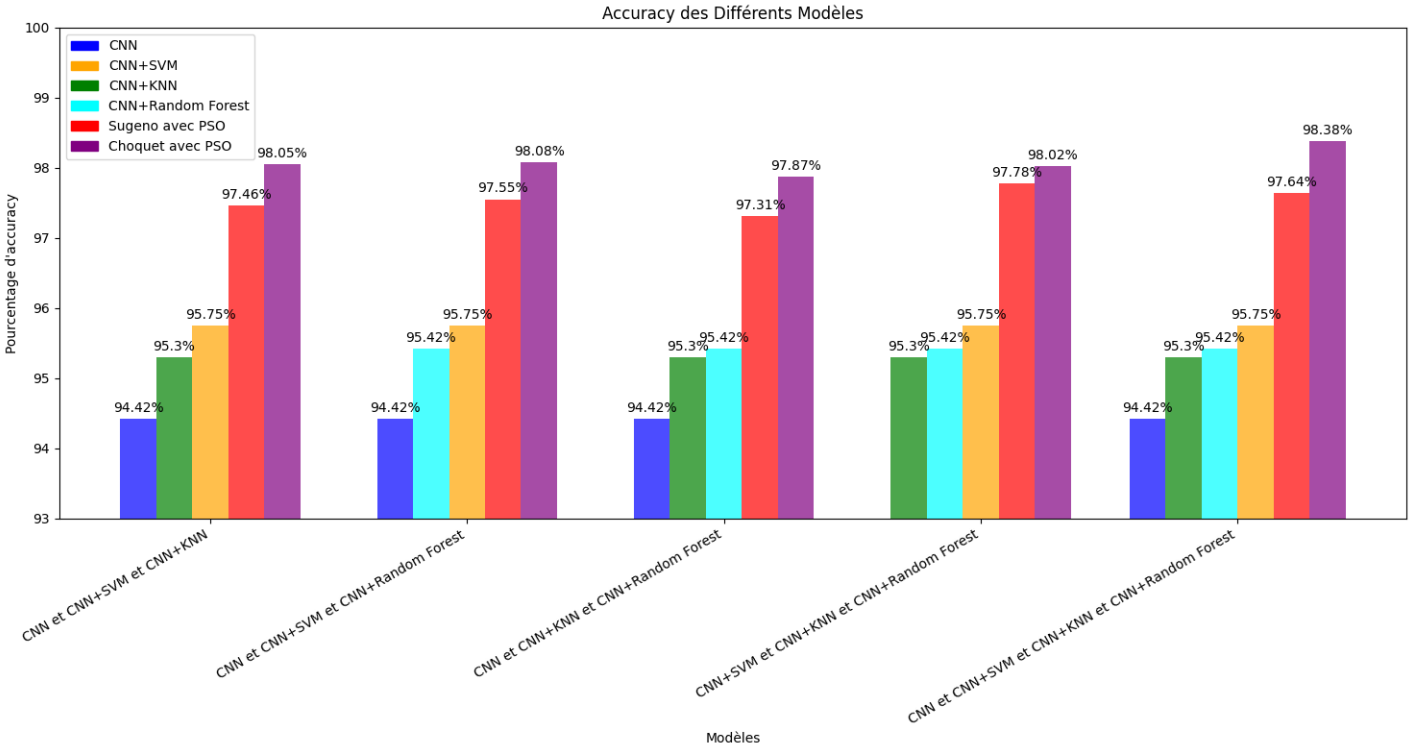

Figure 5.21: Accuracy des différents modèles

Les combinaisons des modèles sur lesquelles l'intégrale et PSO sont appliqués	Exactitude obtenue après l'application de l'intégrale de Sugeno avec PSO (%)	Exactitude obtenue après l'application de l'intégrale de Choquet avec PSO (%)
CNN et CNN+SVM et CNN+KNN	97.46	98.05
CNN et CNN+SVM et CNN+Random Forest	97.55	98.08
CNN et CNN+KNN et CNN+Random Forest	97.31	97.87
CNN+SVM et CNN+KNN et CNN+Random Forest	**97.78**	98.02
CNN et CNN+SVM et CNN+KNN et CNN+Random Forest	97.64	**98.38**

Table 5.21: Accuracy des différents modèles après l'application des intégrales

D'après la figure 5.21 et le tableau 5.21, on constate que l'intégrale de Choquet combinée avec PSO entraîne des améliorations allant jusqu'à 2.7%. La meilleure exactitude obtenue (98,38%) résulte de son application sur les quatre modèles développés dans le protocole 2.
En revanche, l'intégrale de Sugeno avec PSO entraîne des améliorations allant jusqu'à 2.03%, la meilleure exactitude (97,78%) étant obtenue par son application sur trois modèles hybrides (CNN+SVM, CNN+KNN, CNN+RandomForest).
En conclusion, notre étude montre que l'intégrale de Choquet surpasse celle de Sugeno en termes de performance, offrant les meilleurs résultats et cela est dû au fait que l'intégrale de Choquet prend en compte les interactions entre les modèles.

Le modèle le plus performant est celui résultant de l'application de l'intégrale de Choquet avec PSO sur quatre modèles : CNN, CNN+SVM, CNN+KNN et CNN+Random Forest, atteignant une exactitude de 98,38%. Ce modèle présente une amélioration de 2.7% par rapport au meilleur modèle développé avec le protocole 2. En analysant la matrice de confusion (figure 5.20) de ce modèle, on constate que parmi les 3386 images, 55 sont mal classées. Nous avons corrigé la classification de 100 images par rapport au modèle du protocole 2, ce qui est très satisfaisant. Cependant, des erreurs de classification persistent entre certaines classes :

- Parmi les 238 images constituant la classe *Tomato-Late-blight*, 7 sont mal classées dans la classe *Potato-Late-blight*. Cela s'explique par le fait que la maladie est la même, mais le type de plante est différent, rendant le diagnostic des symptômes très difficile, comme illustré dans la figure 5.22.
- Parmi les 233 images constituant la classe *Tomato-Target-spot*, 10 sont mal classées dans la classe *Tomato-Spider-mites-Two-spotted-spider-mite*. Les symptômes des deux classes se ressemblent, rendant le diagnostic très difficile entre ces deux classes, comme illustré dans la figure 5.22.

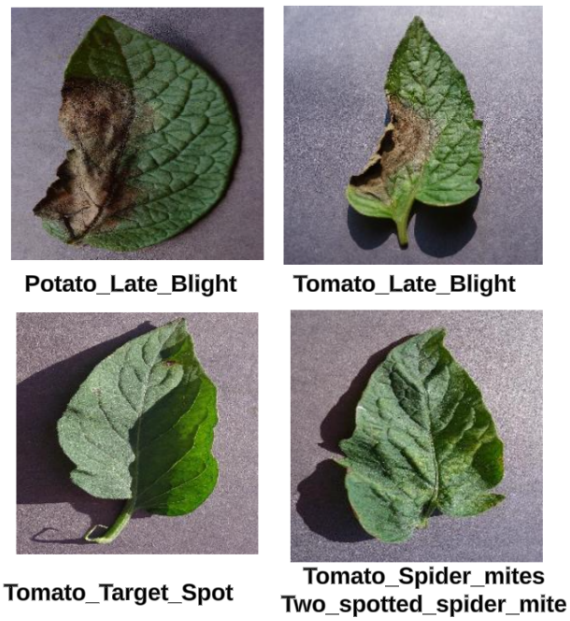

Figure 5.22: Les classes mal classées par notre modèle le plus performant

5.5 Conclusion

A travers ce chapitre, nous avons présenté les méthodes que nous avons utilisé pour améliorer les performances de classification, à savoir les intégrales floues (Sugeno et Choquet) combinées avec l'optimisation par essaim de particules (PSO). Par la suite , nous exposons les résultats obtenus après l'application de cette nouvelle méthode d'amélioration des performances.

En terminant ce chapitre, nous pouvons tirer deux conclusions majeures :

- L'optimisation par essaim de particules est une technique très utile pour déterminer de manière optimale les mesures floues à utiliser dans nos intégrales floues.
- L'intégrale de Sugeno améliore les performances, mais celle de Choquet les améliore encore davantage.

CONCLUSION GÉNÉRALE

Les maladies des plantes constituent une menace persistante pour la qualité et la quantité de la production agricole mondiale, entraînant ainsi des pertes économiques considérables. Cela incite les équipes de recherche du monde entier à intensifier leurs efforts pour lutter contre ces maladies.

Dans ce mémoire, nous avons présenté notre contribution à ce combat. Nous avons commencé par mettre en lumière les travaux les plus récents sur la classification des maladies des plantes en utilisant l'ensemble de données **PlantVillage**. Nous avons décidé de travailler sur un sous-ensemble de quatre types de cultures comprenant 19 classes de plantes, dont 15 maladies. Ce sous-ensemble est extrait de la base de données **PlantVillage** disponible sur Kaggle, et nous avons utilisé des techniques de traitement d'image, d'apprentissage automatique et des outils mathématiques.

Notre approche a consisté à utiliser deux types d'intégrales floues (Choquet et Sugeno) et l'optimisation par essaim de particules pour déterminer les mesures floues optimales attribuées aux quatre modèles que nous avons développés pour la classification des 19 classes simultanément. L'intégrale de Choquet avec PSO appliquée à nos quatre modèles a donné les meilleurs résultats (98,38%).

Ce travail démontre la puissance des intégrales floues combinées avec PSO pour la détermination des mesures floues optimales de manière générale, et de l'intégrale de Choquet avec PSO en particulier, en obtenant des résultats extrêmement encourageants.

Bien entendu, ce travail, qui représente une aide à la classification des maladies des plantes, n'est pas une finalité en soi et ne constitue que la première pierre dans un long combat contre ces maladies. Parmi les perspectives éventuelles pour l'évolution de ce projet, on peut citer :

- L'extension de ce travail à d'autres types de cultures proposés par la base PlantVillage ou d'autres bases publiques, pour confirmer la robustesse des systèmes développés.
- L'implémentation de notre meilleur système obtenu dans une application mobile pour faciliter la tâche de traitement aux utilisateurs grâce à la caméra présente dans le smartphone, sans nécessiter une connexion Internet.
- La mise en place d'une base de données des plantes algériennes permettant l'étude de ces maladies et leur combat à l'échelle nationale.

BIBLIOGRAPHIE

[1] Veysel Aslantas Hasan Ulutaş. Design of efficient methods for the detection of tomato leaf disease utilizing proposed ensemble cnn model. Electronics .2023.

[2] A.Y. Rossman. The impact of invasive fungi on agricultural ecosystems in the united states. the Journal Biological Invasions, Volume 11, No 1, 97–107 .(2009).

[3] James G. Horsfall and Ellis B. Cowling. Plant disease: An advanced treatise. Volume 14. Book.1979.

[4] Dimitrios Moshou Simon Pearson Konstantinos G. Liakos, Patrizia Busato and Dionysis Bochits. Machine learning in agriculture: A review. 2018.

[5] A. Arora S. Baranwal, S. Khandelwal. Deep learning convolutional neural network for apple leaves disease detection. SSRN Electron. J. (2019),.

[6] Mohammad Husain Mohammad Nadeem Ahmed Arshad Ali Parveen Badoni Ashish Gupta, Deepak Gupta. A pso-cnn-based approach for enhancing precision in plant leaf disease detection and classification. Informatica 47 (2023) 173–182.

[7] Gulshan Saleem Muhammad Usman Younus Sadia Anwar Muhammad Rizwan Anjum Nisar Ahmad, Hafz Muhammad Shahzad Asif. Leaf image-based plant disease identifcation using color and texture features. 2021.

[8] Brajesh Kumar Vibhor Kumar Vishnoi, Krishan Kumar. Plant disease detection using computational intelligence and image processing. Journal of Plant Diseases and Protection .2020.

[9] Neha Singh Sumit Chaudhary Madhu Kirola, Kapil Joshi. Plants diseases prediction framework: A image-based system using deep learning. 2022.

[10] L Campbell Z Guo MN Rouse SD Silwal S Tolos B Van Allen Karen A Garrett PD Esker, Adam H Sparks. Ecology and epidemiology in r :disease forecasting and validation. 2016.

[11] Systèmes de prévision des maladies végétales. [https://lc.cx/mB9wgA].

[12] Boyle R Sonka M, Hlavac V. Image pre-processing. Image processing, analysis and machine vision, Springer, Boston, MA, chap 4:56–111.(1993).

[13] Dr. Pushpender Sarao Miss Tejswini S. Danwadkar. Tomato plant leaf disease detection using convolutional neural network. International Journal on Recent and Innovation Trends in Computing and Communication ISSN: 2321-8169 Volume: 11 Issue: 9 .2023.

[14] Jaya Sil Tanmoy Bera, Ankur Das and Asit K. Das. A survey on rice plant disease identification using image processing and data mining techniques. Emerging Technologies in Data Mining and Information Security, Advances in Intelligent Systems and Computing 814 .2019.

[15] Radha Kokare Yogesh Dandawate. An automated approach for classification of plant diseases towards development of futuristic decision support system in indian perspective. International Conference on Advances in Computing, Communications and Informatics (ICACCI) .2015.

[16] Gulshan Saleem Muhammad Usman Younus Sadia Anwar Muhammad Rizwan Anjum Nisar Ahmad, Hafz Muhammad Shahzad Asif. Leaf image-based plant disease identifcation using color and texture features. under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2021.

[17] Bhatia PK Kumar G. A detailed review of feature extraction in image processing systems. In: International conference on advanced computing and communication technologies, ACCT,(2014).

[18] Jain S Chouhan SS, Singh UP. Applications of computer vision in plant pathology: a survey. Arch Comput Methods Eng .(2019).

[19] EHassenian A Mokhtar U, El-Bendary N. Svm-based detection of tomato leaves diseases. 2015.

[20] Karibasappa KG Desai SD Deshapande AS, Giraddi SG. Fungal disease detection in maize leaves using haar wavelet features. Springer, Singapore . (2019).

[21] Huang W You Z Zhang S, Wang H. Plant diseased leaf segmentation and recognition by fusion of superpixel, k-means and phog. Optik Int J Light Electron Opt 157:866–872.(2018).

[22] Iqbal Z Faisal M Ullah MI Younus M Sharif M, Attique M. Detection and classifcation of citrus diseases in agriculture based on optimized weighted segmentation and feature selection. Comput Electron Agric 150, 2018.

[23] Sudha VP. Feature selection techniques for the classification of leaf diseases in turmeric. Int J Comput Trends Technol 43(3):138–142 . 2017.

[24] Gurjar A Gulhane M. Detection of diseases on cotton leaves and its possible diagnosis. Int J Image 5:590–598 .(2011).

[25] Hu B Li MX Zhang YC, Mao HP. Features selection of cotton disease leaves image based on fuzzy feature selection techniques. In: Proceedings of the 2007 international conference on wavelet analysis and pattern recognition, ICWAPR '07 1:124–129.(2008).

[26] Hemalatha M Revathi P. Cotton leaf spot diseases detection utilizing feature selection with skew divergence method. (2014).

[27] Plantvillage dataset/kaggle. [https://urlz.fr/qPlT].

[28] Youcef Chibani Adel Hafiane Amine Mezenner, Hassiba Nemmour. Tomato plant leaf disease classification based on cnn features and support vector machines. 2022.

[29] Chinna Gopi Simhadri Hari Kishan Kondaveeti. Automatic recognition of rice leaf diseases using transfer learning. 2023.

[30] Md. Arif Hossain Toukir Ahammed Md Shakirul Islam 1Nahid Hasan Ashik, Rakibul Islam. A study on the performance of hybrid approach for image classification using cnns and svm for plant disease detection. International Journal of Innovative Science and Research Technology ISSN No:-2456-2165 .2023.

[31] Sowmiya B Saminathan K and Chithra Devi M. Multiclass classification of paddy leaf diseases using random forest classifier. 2023.

[32] NZ Jhanjhi Sarfraz Nawaz Brohi R. Sujatha, Jyotir Moy Chatterjee. Performance of deep learning vs machine learning in plant leaf disease detection. 2021.

[33] H.Mallika K.Indira. Classifcation of plant leaf disease using deep learning. 2024.

[34] Md ABDUR Rahman. Deep learning approaches for tomato plant disease detection. 2020.

[35] Prof Sahil Verma.... Rahul Sharma, Amar Singh. Plant disease diagnosis and image classification using deep learning. 2021.

[36] Kavish mojhoa. Detecting plant disease using cnn and knn. University of Mauritius .2023.

[37] Aman Shakya Bijaya Kumar Hatuwal and Basanta Joshi. Plant leaf disease recognition using random forest, knn, svm and cnn. 2021.

[38] Motahar Reza Tasleem Sultanal. Identification of potato leaf diseases using hybrid convolution neural network with support vector machine. 2023.

[39] Gururaj Vinayakumar Ravi Geetabai, HukkeriB Soundarya. Classification of various plant leaf disease using pretrained convolutional neural network on imagenet. The Open Agriculture Journal .2024.

[40] Emmanuel ESTEVES Directeur RD Mobilité web. L'intelligence artificielle dans le monde numÉrique. 2020.

[41] Intelligence artificielle. [https://urlr.me/jKf4g].

[42] Réseaux de neurones. [https://urlr.me/4gh1F].

[43] Riccardo Molinari... Cristoforo Decaro, Giovanni Battista Montanari. Machine learning approach for prediction of hematic parameters in hemodialysis patients. EEE Journal of Translational Engineering in Health and Medicine .2019.

[44] P. Dollar K. He, G. Gkioxari and R. Girshick. Mask r-cnn. In 2017 IEEE International Conference on Computer Vision (ICCV), pages 2980–2988, Oct 2017.

[45] D. H. Hubel and T. N. Wiesel. Receptive fields and functional architecture of monkey striate cortex. Journal of Physiology (London), 195:215–243, 1968.

[46] K. Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. Biological Cybernetics, 36(4):193–202, Apr 1980.

[47] A. Sufian F. Sultana and P. Dutta. Advancements in image classification using convolutional neural network. In 2018 Fourth International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN), pages 122–129, Nov 2018.

[48] Andronicus A. Akinyelu Laith Abualigah Habiba N. Ngugi, Absalom E. Ezugwu. Revolutionizing crop disease detection with computational deep learning: a comprehensive review. 2024.

[49] Hang Zhang. A review of convolutional neural network development in computer vision. School of Computer Science and Technology, Henan Polytechnic University, Jiaozuo, Henan 454000, P R China .2022.

[50] Fully connected layers in convolutional neural networks. [https://urlz.fr/qOZG].

[51] Andrew Zisserman Karen Simonyan. Very deep convolutional networks for large-scale image recognition. Visual Geometry Group, Department of Engineering Science, University of Oxford .2015.

[52] Xuming Han Muhammet Deveci Milan Parmar Xia Zhao, Limin Wang. A review of convolutional neural networks in computer vision. 2024.

[53] Riries Rulaningtyas Wahyudi Setiawan, Moh.Imam Utoyo. Classification of neovascularization using convolutional neural network model. Informatics Department, University of Trunojoyo Madura, Bangkalan, Indonesia .2018.

[54] Daniel A.Abaye Ernest Yeboah Boateng, Joseph Otoo. Basic tenets of classification algorithms k-nearest-neighbor, support vector machine, random forest and neural network: A review. Journal of Data Analysis and Information Processing .2020.

[55] Elisha B Hubert K. Support vector machines (svm): Explaining svmanditsapplication in regression tasks for sales forecasting. 2023.

[56] K-nearest neighbor(knn) algorithm. [https://www.ibm.com/topics/knn].

[57] Qu'est-ce que l'algorithme des k plus proches voisins ? [https://urlz.fr/qNAL].

[58] What is the random forest algorithm? [https://urlz.fr/qMuz].

[59] M.Sugeno. Theory of fuzzy and its applications. PhD thesis,Tokyo Institute of technology .1974.

[60] Derek T. Anderson Anthony J.Pinar, Timothy C. Havens Muhammad Aminul Islam. Visualization and learning of the choquet integral with limited training data. Conference: FUZZ-IEEE .2017.

[61] Youcef Chibani Nesrine Bouadjenek, Hassiba Nemmour. Fuzzy integrals for combining multiple svm and histogram features for writer's gender prediction. IET Biometrics .2017.

[62] Hoel LE CAPITAINE. Opérateurs d'agrégation pour la mesure de similarité application à l'ambiguïté en reconnaissance de formes. Université de La Rochelle .2009.

[63] Chia-Ying Hsieh Chun-Shu Wei Jian-Xue Huang, Ya-Lin Huang. Fuzzy integral with particle swarm optimization for cnn-based motor-imagery eeg classification. 2021.

[64] Pierre Morizet Yaya Sylla, Adama Coulibaly. Towards deep learning: A comprehensive overview on pso with machine learning. Proceedings of the 4th International Conference on Statistics: Theory and Applications (ICSTA'22) .2022.

[65] Daniela Elena Popescu M. Kalpana Chowdary Andrew J., Jennifer Eunice R. and Jude Hemanth D. Deep learning-based leaf disease detection in crops using images for agricultural applications. *Agronomy*, 12:2395, 2022.

[66] Marcel Salathé David P. Hughes. An open access repository of images on plant health to enable the development of mobile disease diagnostics. arXiv preprint arXiv:1511.08060, 2015.

[67] Qu'est-ce que python ? [https://www.teradata.fr/glossary/what-is-python].

[68] What is scikit-learn? [https://urlz.fr/qCBX].

[69] Qu'est-ce que keras ? [https://urlz.fr/qCF7].

[70] Tensorflow qu'est-ce que c'est? [https://blent.ai/blog/a/tensorflow-deep-learning-python].

[71] Qu'est-ce que colaboratory? [https://urlz.fr/qCOi].